A Hydrodynamic and Thermodynamic Simulation of Droplet Impacts on Hot Surfaces

by

Dalton James Eric Harvie

A thesis submitted to the University of Sydney in fulfilment of the requirements for the degree of Doctor of Philosophy

Department of Mechanical and Mechatronic Engineering
The University of Sydney

August 1999

© D. Harvie
I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other university or institution.

Dalton Harvie

August 1999
Acknowledgments

I am indebted first and foremost to Dr David Fletcher for his friendship, guidance and encouragement during the term of this research. Without David’s extensive knowledge of, and enthusiasm for, boiling heat transfer and computational fluid dynamics, this thesis would not have been possible. I am also heavily indebted to Assoc. Prof. John Kent, particularly for his support during the first years of the degree.

I would also like to thank the staff at Mechanical Engineering, in particular Hasso Nibbe, Greg Elder and Paul Mear for their help and friendship during the degree, and also my closest friends (appearing in strictly alphabetical order) Tim Birkett, Geoffrey Davis, Chris Jenkins, Shane Tevlin and Ross West.

Special thanks is extended to Marie-Clarie Muir, who during the last few years has had the pleasure of enduring many a monologue on such fascinating topics as the advantages of \LaTeX{} through to the demise of the modern motorcycle. Her friendship has been a constant source of inspiration.

Finally I would like to thank my close family for their love and encouragement, in particular my parents, who have been extremely supportive throughout all the years of my extended university education.
Abstract

In this thesis a computational code, termed BOUNCE, has been developed to simulate the behaviour of volatile liquid droplets impacting with a hot solid surface in the film boiling regime. BOUNCE is comprised of a Volume of Fluid (VOF) code, used to model fluid dynamics within the volume of the droplet, and a separate one dimensional code, used to model fluid behaviour within the vapour layer existing between the droplet and solid surface.

Two new VOF advection schemes have been developed as part of this study. Both schemes use a piecewise linear technique for reconstructing fluid free surfaces, coupled to a fully multidimensional integration technique for calculating cell boundary fluxes. The Defined Donating Region (DDR) scheme is shown to be a computationally efficient scheme, particularly suited to complex flow problems where stability of the fluid free surfaces is paramount. The Stream scheme, based on a completely new computational approach, is shown to be slightly more accurate than the DDR scheme, but also to be more computationally expensive.

The vapour layer code employed in BOUNCE uses a one dimensional parabolic viscous flow assumption, and solves heat transfer equations within the liquid, solid and vapour phases. A velocity and temperature molecular interface treatment is included at both the liquid-vapour and solid-vapour interfaces, and an air volume fraction transport equation is solved within the
Comparisons of simulated and actual droplet impacts have shown that BOUNCE is able to accurately simulate the hydrodynamic and thermodynamic behaviour of saturated droplet film boiling impacts. It has also been shown that BOUNCE is able to predict the behaviour of significantly subcooled n-Heptane film boiling impacts. However, the ability of BOUNCE to predict significantly subcooled water impacts has not been validated, as experimental data of film boiling water impacts does not appear to be presently available.
Table of Contents

1 Introduction 1
  1.1 Droplet Impact Applications 1
  1.2 The Thermodynamic Regimes of Droplet Impact 3
  1.3 The Hydrodynamic Regimes of Droplet Impact 7
  1.4 Scope and Structure of the Thesis 8

2 Literature Review 10
  2.1 Steady State Droplet Film Boiling 10
  2.2 Film Boiling Droplet Impact Experimental Studies 12
    2.2.1 Saturated Droplet Impact Experimental Studies 12
    2.2.2 Subcooled Droplet Impact Experimental Studies 15
  2.3 Film Boiling Droplet Impact Analytical Studies 23

3 The Cylindrical Droplet Impact Model 25
  3.1 The Buyevich, Mankevich and Polotsky Droplet Impact Model 25
  3.2 Validation of the BOING Code 30
  3.3 Performance of the Buyevich, Mankevich and Polotsky Impact Model 33

4 The Internal Droplet Dynamics Algorithm 38
  4.1 The Main Features of the BOUNCE VOF Code 38
4.2 Volume of Fluid Codes ........................................... 39
4.3 Iterative Implicit Structure of the BOUNCE VOF Algorithm 40
   4.3.1 Basic Code Structure ..................................... 40
   4.3.2 The Main Subroutines used by BOUNCE .............. 40
   4.3.3 Implicit Convergence Criterion ......................... 43
   4.3.4 Variable Timestep ....................................... 45
   4.3.5 Failed Iterations ....................................... 46
4.4 PPE-ICCG solution ............................................. 46
   4.4.1 Pressure Poisson Equation .............................. 46
   4.4.2 Variable Placement and Computational Mesh Geometry 48
   4.4.3 Momentum Advection and Viscous Stresses .......... 50
   4.4.4 Numerical Solution .................................... 52
   4.4.5 Boundary Conditions .................................. 54
   4.4.6 Incomplete Cholesky Conjugate Gradient PPE Matrix
       Inversion ....................................................... 58
4.5 Free Surface Forces .......................................... 58
   4.5.1 Surface Tension .......................................... 58
   4.5.2 The Viscous Vapour Layer Surface Force ............ 73
   4.5.3 Implementation of the Surface Volume Forces in the
       PPE Equation ................................................. 75
5 The Defined Donating Region VOF Advection Algorithm 77
   5.1 VOF Advection ............................................... 77
   5.2 The Hirt-Nichols VOF advection Algorithm ........... 81
       5.2.1 Translation Test - The Box Problem .............. 81
       5.2.2 The Advection Model ................................ 82
       5.2.3 Stability Criterion ................................. 85
       5.2.4 The Computational Procedure ....................... 85
5.2.5 VOF Conservation ........................................... 86
5.2.6 VOF Diffusion ............................................. 89
5.3 The Defined Donating Region Advection Algorithm ....... 93
   5.3.1 Defined Donating Regions ............................ 94
   5.3.2 Free Surface Reconstruction ......................... 105
   5.3.3 Integration of the VOF Fluxes ....................... 117
   5.3.4 The Code Structure .................................. 119
   5.3.5 Translation Test - The Box Problem ............... 121

6 The Stream VOF Convection Algorithm .......................... 124
   6.1 The Basic Method ....................................... 124
   6.2 Defining the Velocity Field ............................ 125
   6.3 The Fluid Boundary Flux Calculation .................. 127
      6.3.1 Approximate Integration Method ................ 129
      6.3.2 Fluid Particle Trajectory Calculation .......... 132
   6.4 Algorithm Accuracy ..................................... 137
      6.4.1 Fluid Volume Conservation ....................... 137
      6.4.2 Fluid and Void Wisp Generation ................. 139
   6.5 Numerical Procedure .................................... 141
   6.6 Translation Test - The Box Problem ................... 143
   6.7 Cylindrical Coordinates ................................ 145

7 VOF Advection Tests — A Comparison of VOF Advection ..... 147
   Algorithm Accuracy ......................................... 147
      7.1 The Zalesak Slotted Disk Test ....................... 147
      7.2 Non-Uniform Vorticity Tests ....................... 152
         7.2.1 Single Vortex ................................... 152
         7.2.2 A Deformation Field Test ..................... 160
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.2.1</td>
<td>The Height Calculation Method</td>
<td>204</td>
</tr>
<tr>
<td>9.2.2</td>
<td>The Accuracy and Stability of the Vapour Layer Height Calculation</td>
<td>207</td>
</tr>
<tr>
<td>9.2.3</td>
<td>The Velocity of the Underside of the Droplet</td>
<td>209</td>
</tr>
<tr>
<td>9.3</td>
<td>Vaporisation Velocity Calculation</td>
<td>209</td>
</tr>
<tr>
<td>9.3.1</td>
<td>Solid Phase Heat Transfer</td>
<td>210</td>
</tr>
<tr>
<td>9.3.2</td>
<td>Liquid Phase Heat Transfer</td>
<td>218</td>
</tr>
<tr>
<td>9.3.3</td>
<td>Solid-Vapour Interface</td>
<td>226</td>
</tr>
<tr>
<td>9.3.4</td>
<td>Liquid-Vapour Interface</td>
<td>227</td>
</tr>
<tr>
<td>9.3.5</td>
<td>Heat Transfer Simplifications</td>
<td>230</td>
</tr>
<tr>
<td>9.3.6</td>
<td>Vaporisation Velocity Solution Procedure</td>
<td>231</td>
</tr>
<tr>
<td>9.4</td>
<td>Vapour Layer Flow Solution</td>
<td>236</td>
</tr>
<tr>
<td>9.4.1</td>
<td>Linear Pressure Relationship</td>
<td>237</td>
</tr>
<tr>
<td>9.4.2</td>
<td>Pressure Boundary Conditions</td>
<td>243</td>
</tr>
<tr>
<td>9.4.3</td>
<td>Vapour Layer Flow Solution Technique</td>
<td>247</td>
</tr>
<tr>
<td>9.5</td>
<td>Air Volume Fraction Solution</td>
<td>248</td>
</tr>
<tr>
<td>9.5.1</td>
<td>Linear Air Volume Fraction Relationship</td>
<td>248</td>
</tr>
<tr>
<td>9.5.2</td>
<td>Air Volume Fraction Boundary Conditions</td>
<td>250</td>
</tr>
<tr>
<td>9.5.3</td>
<td>Accuracy Analysis</td>
<td>251</td>
</tr>
<tr>
<td>9.5.4</td>
<td>Air Volume Fraction Solution Technique</td>
<td>253</td>
</tr>
<tr>
<td>10</td>
<td>Droplet Impact Simulations — Saturated Droplets</td>
<td>255</td>
</tr>
<tr>
<td>10.1</td>
<td>The Wachters and Westerling Impacts</td>
<td>255</td>
</tr>
<tr>
<td>10.1.1</td>
<td>The We = 15 Wachters and Westerling Impact</td>
<td>256</td>
</tr>
<tr>
<td>10.1.2</td>
<td>The We = 74 Wachters and Westerling Impact</td>
<td>275</td>
</tr>
<tr>
<td>10.1.3</td>
<td>The We = 184 Wachters and Westerling Impact</td>
<td>282</td>
</tr>
<tr>
<td>10.2</td>
<td>The Groendes and Mesler Impact</td>
<td>289</td>
</tr>
<tr>
<td>10.2.1</td>
<td>Hydrodynamic Behaviour</td>
<td>289</td>
</tr>
</tbody>
</table>
10.2.2 Thermodynamic Behaviour ............................................. 294

11 Droplet Impact Simulations — Subcooled Droplets .......................... 303

11.1 Subcooled n-Heptane Droplet Impacts .................................... 303

11.1.1 Hydrodynamic Behaviour ............................................. 304

11.1.2 Thermodynamic Behaviour ............................................. 310

11.1.3 A Comparison Between the Hydrodynamic Behaviour

of Saturated and Subcooled Impacts ............................................. 319

11.1.4 The BOUNCE Code and the Different Impact Boiling

Regimes ................................................................. 320

11.1.5 Vapour Layer Destabilisation ........................................... 324

11.2 Subcooled Water Droplet Impacts ........................................... 332

11.2.1 The Qiao and Chandra Water Impacts .................................. 332

11.2.2 The Chen and Hsu Impact ................................................ 339

12 Conclusions and Future Work .................................................. 345

A Kinetic Theory Analysis of Vapour Region Boundaries ......................... 363

A.1 Molecular Flow Regimes ..................................................... 363

A.2 A Molecular Treatment of the Solid-Vapour Interface ....................... 365

A.2.1 Interface Slip Velocity ..................................................... 365

A.2.2 Interface Temperature Discontinuity ...................................... 368

A.3 A Molecular Treatment of the Liquid-Vapour Interface ...................... 372

A.3.1 Interface Mass Balance ..................................................... 372

A.3.2 Interface Slip Velocity ..................................................... 375

A.3.3 Interface Temperature Discontinuity ...................................... 376

A.4 Vapour Mixture Mean Free Path ............................................. 380
B The Thermal Energy Transport Equation Applied to a VOF Simulation

C Material and Computational Properties used in the Droplet Impact Simulations

C.1 Liquid Properties
C.1.1 Water
C.1.2 n-Heptane

C.2 Vapour Properties
C.2.1 Wachters and Westerling Impacts — Water
C.2.2 Groendes and Mesler Impacts — Water
C.2.3 Qiao and Chandra and Chandra and Avedisian Impacts — n-Heptane and Air
C.2.4 Qiao and Chandra Impacts — Water and Air
C.2.5 Chen and Hsu Impact — Water and Air

C.3 Solid Properties
C.3.1 Wachters and Westerling Impacts — Gold
C.3.2 Groendes and Mesler Impacts — Quartz
C.3.3 Qiao and Chandra and Chandra and Avedisian Impacts — Stainless Steel
C.3.4 Chen and Hsu Impacts — Inconel 600

C.4 Gas Molecular Properties
C.5 Computational Properties

D Viscous Vapour Layer Code Integrity Calculations
D.1 Pressure Application Test
D.2 Pressure Generation Test
D.3 Air Volume Fraction Test
### List of Figures

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Three normal gravity n-Heptane impacts conducted at the different initial solid temperatures of 24°C, 90°C and 104°C</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>Three normal gravity n-Heptane impacts conducted at the different initial solid temperatures of 160°C, 195°C and 205°C</td>
<td>6</td>
</tr>
<tr>
<td>3.1</td>
<td>The geometry and main variables used to describe the Buyevich and Mankevich cylindrical droplet</td>
<td>27</td>
</tr>
<tr>
<td>3.2</td>
<td>Sample calculations presented by Buyevich &amp; Mankevich [1995]</td>
<td>31</td>
</tr>
<tr>
<td>3.3</td>
<td>Two droplet impacts calculated by the BOING code. Shown in this figure is the non-dimensional radius of the droplet</td>
<td>32</td>
</tr>
<tr>
<td>3.4</td>
<td>Two droplet impacts calculated by the BOING code. Shown in this figure is the non-dimensional height of the vapour layer</td>
<td>33</td>
</tr>
<tr>
<td>3.5</td>
<td>The We = 15 impact performed by Wachters and Westerling</td>
<td>34</td>
</tr>
<tr>
<td>3.6</td>
<td>The droplet height and radius during the We = 15 Wachters and Westerling water impact</td>
<td>35</td>
</tr>
<tr>
<td>3.7</td>
<td>The vapour layer height, $h$, as calculated using the Buyevich, Mankevich and Polotsky droplet impact model for the We = 15 Wachters and Westerling impact</td>
<td>37</td>
</tr>
<tr>
<td>4.1</td>
<td>A schematic showing the basic code structure of BOUNCE</td>
<td>41</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>5.13</td>
<td>An example fluid configuration used to illustrate the grid dependence of the Pilliod error function.</td>
<td></td>
</tr>
<tr>
<td>5.14</td>
<td>Comparison of the BOUNCE and Pilliod error functions for the example shown in Figure 5.13.</td>
<td></td>
</tr>
<tr>
<td>5.15</td>
<td>Interfaces are classified into eight different types in BOUNCE, depending on the location and orientation of the interface.</td>
<td></td>
</tr>
<tr>
<td>5.16</td>
<td>Steps used to determine the fluid interface type within each cell.</td>
<td></td>
</tr>
<tr>
<td>5.17</td>
<td>Example surface reconstructions showing $F_{\text{min}}$ (A) and $F_{\text{max}}$ (B) for a negative gradient interface closer to vertical than horizontal. Also shown are the location of the left and right points for an $itype = 4$ reconstruction with the fluid positioned to the left of the interface (C).</td>
<td></td>
</tr>
<tr>
<td>5.18</td>
<td>Example flux calculation over a right donating boundary.</td>
<td></td>
</tr>
<tr>
<td>5.19</td>
<td>Schematic of the BOUNCE subroutine $VFGRAD$.</td>
<td></td>
</tr>
<tr>
<td>5.20</td>
<td>Test case calculated using the BOUNCE Defined Donating Region algorithm.</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Example in cell velocity fields, represented by fluid streamlines.</td>
<td></td>
</tr>
<tr>
<td>6.2</td>
<td>Example channel used to calculate the fluid flux over the right boundary of cell $i, j$.</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>Angles are calculated to the left and right points of the free surface interface in order to determine whether the point lies in a fluid or void region.</td>
<td></td>
</tr>
</tbody>
</table>
6.4 A worst case scenario for border fluid flux calculation. The fluid free surface is orientated parallel with the integration channels, and the free surface is located close to a characteristic streamline.

6.5 Schematic showing the algorithm used to locally redistribute non-real $F$ values.

6.6 Schematic of subroutine $VFSTREAM$, used to advect the VOF function using the Stream method.

6.7 Schematic showing function $TFLUID$. This function is used by subroutine $VFSTREAM$ to calculate $t_{fluid}$ for each cell when integrating boundary fluxes.

6.8 Test case calculated using the BOUNCE Stream algorithm. Here $n_{channel} = 10$ and $F_{wisp} = 0.1$.

6.9 Test case calculated using the BOUNCE Stream algorithm. Here $n_{channel} = 100$ and $F_{wisp} = 0.01$.

7.1 Zalesak slotted disk rotation test performed with the H-N, DDR and Stream VOF advection algorithms.

7.2 Single vortex flow field deforming a cylinder. Marker particle positions shown at indicated times approximate exact solution.

7.3 Single vortex VOF advection test calculated with the Defined Donating Region scheme using the Ripple method for free surface gradient calculation.

7.4 Single vortex VOF advection test calculated with the Defined Donating Region scheme using the error minimisation method for free surface gradient calculation.
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5</td>
<td>Single vortex VOF advection test calculated with the Stream scheme using the error minimisation method for free surface gradient calculation. Here $n_{\text{channel}} = 100$, and $F_{\text{wisp}} = 0.01$.</td>
</tr>
<tr>
<td>7.6</td>
<td>Time reversed single vortex VOF advection test.</td>
</tr>
<tr>
<td>7.7</td>
<td>Deformation field deforming a cylinder. Marker particle positions shown at indicated times approximate exact solution.</td>
</tr>
<tr>
<td>7.8</td>
<td>Time reversed single vortex VOF advection test.</td>
</tr>
<tr>
<td>7.9</td>
<td>Deformation field VOF advection test calculated with the Stream scheme.</td>
</tr>
<tr>
<td>7.10</td>
<td>Example donating regions defined using the DDR (A), Rider-Kothe (B) Puckett, Almgren, Bell, Marcus &amp; Rider (C) and Stream (D) schemes.</td>
</tr>
<tr>
<td>8.1</td>
<td>The basic variables used in the viscous vapour layer analysis.</td>
</tr>
<tr>
<td>8.2</td>
<td>A cylinder of radius $r'$, contained within the vapour layer, is used to perform continuity calculations.</td>
</tr>
<tr>
<td>8.3</td>
<td>Liquid, vapour mixture and solid temperatures at radius $r$ used in the calculation of the vaporisation velocity, $\dot{w}$.</td>
</tr>
<tr>
<td>9.1</td>
<td>The vapour layer flow mesh and associated variable positioning.</td>
</tr>
<tr>
<td>9.2</td>
<td>Schematic showing the procedure used by subroutine VA-PORLAYER to solve vapour layer flow.</td>
</tr>
<tr>
<td>9.3</td>
<td>Schematic showing the procedure used by subroutine VA-PORSUB to calculate vapour layer pressures.</td>
</tr>
<tr>
<td>9.4</td>
<td>An example droplet used to demonstrate vapour layer height calculations.</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>9.5</td>
<td>The temperature distribution within the solid phase is represented on a variable mesh size grid.</td>
</tr>
<tr>
<td>9.6</td>
<td>The temperature distribution within the liquid phase is represented on a variable mesh size grid.</td>
</tr>
<tr>
<td>9.7</td>
<td>Control volume contained within subcell ( o ), used for the air volume fraction stability analysis.</td>
</tr>
<tr>
<td>10.1</td>
<td>Simulated droplet impact replicating the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.2</td>
<td>Height of vapour layer at the centreline of the droplet, ( \delta_2 ), and the minimum vapour layer height, ( \delta_{\text{min}} ), during the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.3</td>
<td>Maximum droplet radius and height during the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.4</td>
<td>Maximum Knudsen number, ( \text{Kn} ), within the vapour layer during the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.5</td>
<td>Average droplet vertical velocity during the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.6</td>
<td>Vapour layer pressure at the axis of the droplet during the Wachters and Westerling ( \text{We} = 15 ) impact simulation.</td>
</tr>
<tr>
<td>10.7</td>
<td>Temperatures at the liquid-vapour and solid-vapour interfaces predicted during the Wachters and Westerling ( \text{We} = 15 ) impact.</td>
</tr>
<tr>
<td>10.8</td>
<td>Total energy transported during the simulated Wachters and Westerling ( \text{We} = 15 ) droplet impact.</td>
</tr>
<tr>
<td>10.9</td>
<td>Local vapour layer centreline energy fluxes during the simulated Wachters and Westerling ( \text{We} = 15 ) droplet impact.</td>
</tr>
</tbody>
</table>
10.10 Simulated droplet impact replicating the Wachters and Westerling $We = 15$ impact. The computational cell size used in this calculation was twice the size of the cell size used in the previous calculations.

10.11 Simulated droplet impact replicating the Wachters and Westerling $We = 15$ impact. In this calculation the temperature of the solid was assumed constant, and the vapour layer molecular interface treatment was neglected.

10.12 The $We = 74$ impact performed by Wachters and Westerling.

10.13 Simulated droplet impact replicating the Wachters and Westerling $We = 74$ impact.

10.14 Maximum radius and height of the experimental and simulated Wachters and Westerling $We = 74$ impact.

10.15 The maximum Knudsen number calculated during the simulated Wachters and Westerling $We = 74$ impact.

10.16 The $We = 184$ impact performed by Wachters and Westerling.

10.17 Simulated droplet impact replicating the Wachters and Westerling $We = 184$ impact.

10.18 The maximum Knudsen number calculated during the simulated impact of the Wachters and Westerling $We = 184$ impact.

10.19 The Groendes and Mesler water droplet impact. This figure displays frames from the first 20 ms of the impact.

10.20 The Groendes and Mesler water droplet impact. This figure is a continuation of Figure 10.19.

10.21 The Groendes and Mesler droplet impact simulated by the BOUNCE code.
10.22 Vapour layer heights below the simulated Groendes and Mesler droplet impact. ........................................... 295
10.23 The maximum Knudsen number calculated during the simulated the Groendes and Mesler impact. ............... 296
10.24 The solid surface temperature measured beneath the Groendes and Mesler subcooled water droplet impact. ....... 297
10.25 Vapour layer interface temperatures at the centre of the computational domain during the simulated Groendes and Mesler droplet impact. .................................................. 298
10.26 Total energy transported during the simulated Groendes and Mesler droplet impact. ....................................... 300
10.27 Local vapour layer centreline energy fluxes during the simulated Groendes and Mesler droplet impact. ................... 301
11.1 Three low gravity n-Heptane impacts conducted at different initial solid temperatures. .................................. 305
11.2 Simulated droplet impact replicating the Qiao and Chandra n-Heptane low-gravity experiment. The initial solid temperature was 210°C. ......................................................... 306
11.3 Simulated droplet impact replicating the Chandra and Ave-disian n-Heptane normal gravity experiment. The initial solid temperature was 205°C. ......................................................... 309
11.4 Temperatures at the liquid-vapour and solid-vapour interfaces predicted during the low gravity Qian and Chandra impact. The initial solid temperature in this simulation was 300°C. ......................................................... 311
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.5</td>
<td>The solid surface temperature drop under a low gravity n-Heptane impact. The initial surface temperature for this simulation was 300°C.</td>
<td>313</td>
</tr>
<tr>
<td>11.6</td>
<td>Volume fraction of air within the vapour layer beneath the axis of the droplet, as calculated for the low gravity Qiao and Chandra n-Heptane impact with the initial solid temperature of 300°C.</td>
<td>315</td>
</tr>
<tr>
<td>11.7</td>
<td>Local vapour layer centreline energy fluxes during the simulated low gravity Qiao and Chandra impact. The initial surface temperature for this impact was 300°C.</td>
<td>317</td>
</tr>
<tr>
<td>11.8</td>
<td>Total energy transported during the simulated low gravity Qiao and Chandra impact. The initial surface temperature for this impact was 300°C.</td>
<td>318</td>
</tr>
<tr>
<td>11.9</td>
<td>The maximum solid surface temperature drop occurring during n-Heptane impacts as a function of the initial temperature of the solid.</td>
<td>321</td>
</tr>
<tr>
<td>11.10</td>
<td>The minimum height of the vapour layer during a low gravity n-Heptane impact. The initial solid temperature for this simulation was 200°C.</td>
<td>326</td>
</tr>
<tr>
<td>11.11</td>
<td>The maximum Knudsen numbers beneath four low gravity n-Heptane impacts as a function of time.</td>
<td>327</td>
</tr>
<tr>
<td>11.12</td>
<td>Three low gravity water impacts conducted at different initial solid temperatures.</td>
<td>333</td>
</tr>
<tr>
<td>11.13</td>
<td>Simulated droplet impact replicating the Qiao and Chandra water low-gravity experiment. The initial solid temperature was 320°C.</td>
<td>334</td>
</tr>
</tbody>
</table>
11.14 The maximum Knudsen number within the vapour layer during a low gravity water impact. The initial solid temperature for this simulation was 320°C.

11.15 The minimum height of the vapour layer during a low gravity water impact. The initial solid temperature for this simulation was 320°C.

11.16 The maximum solid surface temperature drop occurring during subcooled water impacts as a function of the initial temperature of the solid.

11.17 Transient solid surface superheat and heat flux for the 80 K subcooled Chen and Hsu water droplet.

11.18 The centreline solid surface temperature predicted by BOUNCE during the Chen and Hsu subcooled water impact.

11.19 The centreline energy flux from the solid as predicted by BOUNCE during the Chen and Hsu subcooled water impact.

11.20 The maximum Knudsen number within the vapour layer during the simulated Chen and Hsu subcooled water droplet impact.

A.1 Tangential velocities at a solid-vapour interface.

A.2 Temperatures at a solid-vapour interface.

A.3 Molecular evaporation and condensation rates at a liquid-vapour interface.

A.4 Control volume situated at a liquid-vapour interface.
| D.1 | Pressure application test case showing the height of the vapour layer when a cylinder of fluid is accelerated by a constant vapour layer pressure. Both analytical and BOUNCE solutions are shown. | 396 |
| D.2 | Non-dimensional vapour layer flowrates and pressures calculated by BOUNCE, and analytically. | 398 |
| D.3 | Analytical and BOUNCE solutions to the air volume fraction test case. The volume fraction shown was evaluated at the centreline of the computational domain. | 400 |
| D.4 | Non-dimensional solid surface temperature calculated by BOUNCE and analytically for the solid phase conduction test. | 401 |
| D.5 | Non-dimensional liquid surface temperature calculated by BOUNCE and analytically for the liquid phase conduction test. | 403 |
| D.6 | Temperatures calculated within the liquid phase during the convection test. | 404 |
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Pressures $P_{av}$ calculated within a stationary cylinder of radius $R$.</td>
<td>72</td>
</tr>
<tr>
<td>7.1</td>
<td>Calculation times for the Zalesak slotted disk problem.</td>
<td>151</td>
</tr>
<tr>
<td>C.1</td>
<td>Liquid water properties at 373 K.</td>
<td>385</td>
</tr>
<tr>
<td>C.2</td>
<td>Liquid water properties at 298 K.</td>
<td>385</td>
</tr>
<tr>
<td>C.3</td>
<td>Saturation constants for water.</td>
<td>386</td>
</tr>
<tr>
<td>C.4</td>
<td>Liquid n-Heptane properties at 298 K.</td>
<td>386</td>
</tr>
<tr>
<td>C.5</td>
<td>Saturation constants for n-Heptane.</td>
<td>387</td>
</tr>
<tr>
<td>C.6</td>
<td>Steam properties at 520 K.</td>
<td>387</td>
</tr>
<tr>
<td>C.7</td>
<td>Steam properties at 550 K.</td>
<td>388</td>
</tr>
<tr>
<td>C.8</td>
<td>Gaseous n-Heptane properties at 447 K.</td>
<td>388</td>
</tr>
<tr>
<td>C.9</td>
<td>Properties for air at 447 K.</td>
<td>389</td>
</tr>
<tr>
<td>C.10</td>
<td>Steam properties at 473 K.</td>
<td>389</td>
</tr>
<tr>
<td>C.11</td>
<td>Properties for air at 473 K.</td>
<td>389</td>
</tr>
<tr>
<td>C.12</td>
<td>Steam properties at 573 K.</td>
<td>390</td>
</tr>
<tr>
<td>C.13</td>
<td>Properties for air at 573 K.</td>
<td>390</td>
</tr>
<tr>
<td>C.14</td>
<td>Gold properties at 673 K.</td>
<td>390</td>
</tr>
<tr>
<td>C.15</td>
<td>Quartz properties at 735 K.</td>
<td>391</td>
</tr>
<tr>
<td>C.16</td>
<td>Stainless steel properties at 523 K.</td>
<td>392</td>
</tr>
</tbody>
</table>
C.17 Inconel 600 properties at 773 K. . . . . . . . . . . . . . . . 392
C.18 Molecular gas properties. . . . . . . . . . . . . . . . . . . . 393
C.19 Computational properties used in integrating the PPE equa-
tion. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 394
C.20 The solid and liquid phase temperature grid specifications
used in the droplet impact simulations. . . . . . . . . . . . . 394
# Nomenclature

**Arabic Symbols**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>area</td>
<td>$[m^2]$</td>
</tr>
<tr>
<td>$\bar{c}$</td>
<td>mean molecular speed of the gas</td>
<td>$[m/s]$</td>
</tr>
<tr>
<td>$c_p$</td>
<td>specific heat at constant pressure</td>
<td>$[J/kg.K]$</td>
</tr>
<tr>
<td>$c_v$</td>
<td>specific heat at constant volume</td>
<td>$[J/kg.K]$</td>
</tr>
<tr>
<td>$d$</td>
<td>droplet diameter</td>
<td>$[m]$</td>
</tr>
<tr>
<td>$e$</td>
<td>energy</td>
<td>$[J]$</td>
</tr>
<tr>
<td>$E$</td>
<td>kinetic energy (cylindrical model)</td>
<td>$[J]$</td>
</tr>
<tr>
<td>$E$</td>
<td>error</td>
<td></td>
</tr>
<tr>
<td>$f$</td>
<td>force generated by vapour layer (cylindrical model)</td>
<td>$[N]$</td>
</tr>
<tr>
<td>$F$</td>
<td>volume of fluid (VOF) function</td>
<td></td>
</tr>
<tr>
<td>$F$</td>
<td>body force vector</td>
<td>$[N/kg]$</td>
</tr>
<tr>
<td>$g$</td>
<td>gravitation vector</td>
<td>$[m/s^2]$</td>
</tr>
<tr>
<td>$Gr$</td>
<td>Grashof number</td>
<td></td>
</tr>
<tr>
<td>$h$</td>
<td>height of vapour layer (cylindrical model)</td>
<td>$[m]$</td>
</tr>
<tr>
<td>$H$</td>
<td>enthalpy</td>
<td>$[J/kg]$</td>
</tr>
<tr>
<td>$i$</td>
<td>horizontal or radial unit vector</td>
<td></td>
</tr>
<tr>
<td>$j$</td>
<td>vertical or axial unit vector</td>
<td></td>
</tr>
<tr>
<td>$J$</td>
<td>mass flux</td>
<td>$[kg/s]$</td>
</tr>
<tr>
<td>$k$</td>
<td>thermal conductivity</td>
<td>$[W/m.K]$</td>
</tr>
</tbody>
</table>
Kn  Knudsen number

l  droplet half height (cylindrical model) [m]

M  molecular mass [kg/kmol]

Ma  Marangoni number

n  free surface interface normal vector [1/m]

Na  Avogadro constant [1/kmol]

P  Pressure [Pa]

q  energy flux [W/m²]

r  radial displacement [m]

R  Universal Gas constant [J/kmol.K]

R  radius of vapour layer [m]

Ra  surface roughness [m]

Re  Reynolds number

S  vapour mixture volume flowrate [m³/s]

t  time [s]

T  time period [s]

T  temperature [K]

u  horizontal or radial velocity [m/s]

U  surface energy (cylindrical model) [J]

v  axial or vertical velocity [m/s]

V  volume [m³]

V  velocity vector [m/s]

w  droplet underside vaporisation velocity [m/s]

We  Weber number

x  radial or horizontal displacement [m]

x  dimensionless radius (cylindrical model)

X  horizontal or radial dimension [m]
$y$  vertical or axial displacement [m]
$y$  vapour volume fraction
$Y$  vertical or axial dimension [m]
$z$  axial displacement [m]

Greek Symbols

$\alpha$  thermal diffusivity [m$^2$/s]
$\alpha$  momentum advection scheme coefficient
$\gamma$  ratio of specific heats
$\delta$  vapour layer height [m]
$\varepsilon$  small parameter
$\eta$  dimensionless vapour layer height (cylindrical model)
$\theta$  angle [rad]
$\kappa$  curvature [1/m]
$\lambda$  mean free path [m]
$\mu$  absolute viscosity [N.s/m$^2$]
$\nu$  kinematic viscosity [m$^2$/s]
$\xi$  Cartesian or cylindrical coordinate coefficient
$\rho$  density [kg/m$^3$]
$\sigma$  surface tension [N/m]
$\sigma$  molecular collision diameter [m]
$\sigma_{S,B}$  Stefan-Boltzmann constant [W/m$^2$.K$^4$]
$\sigma_t$  thermal accommodation coefficient
$\sigma_v$  specular reflection coefficient
$\tau$  dimensionless time (cylindrical model)
$\tau$  stress [N/m$^2$]
$\tau$  viscous stress tensor [N/m$^2$]
\( \phi \) ratio of mass fluxes
\( \Psi \) stream function \([\text{m}^2/\text{s}]\)

Superscripts
\( n \) timestep index

Subscripts
\( a \) air
\( a \) average
\( A \) accepting
\( \text{atmos} \) atmospheric
\( B \) bottom
\( c \) condensing
\( D \) donating
\( e \) evaporating
\( f \) liquid
\( g \) vapour
\( i \) initial
\( i \) horizontal or radial VOF cell index
\( j \) vertical or axial VOF cell index
\( k \) axial temperature cell index
\( l \) axial temperature cell index
\( l \) liquid
\( L \) left
\( m \) vapour mixture
\( \text{max} \) maximum
\( \text{min} \) minimum
\( o \) vapour layer subcell index
\( o \) nominal value (cylindrical model)
\( R \) right
\( s \) solid
\( \text{sat} \) saturation
\( sv \) surface volume
\( T \) top
\( v \) vapour
\( vl \) vapour layer
\( x \) horizontal or radial component
\( y \) vertical or axial component
Chapter 1

Introduction

While the interaction between liquid sprays and hot solid objects occurs in a wide variety of industrial, domestic and environmental applications, our understanding of the mechanisms involved in the process are far from complete. Indeed, current methods of estimating the heat transfer and fluid dynamics of sprays impacting hot surfaces are largely empirically derived. Of fundamental importance to such processes is the hydrodynamic and thermodynamic behaviour of individual droplets which impact a solid surface. It is the simulation of this phenomenon that constitutes the topic of this thesis.

1.1 Droplet Impact Applications

Liquid sprays are used as a means of cooling materials in a wide variety of industrial applications. In the continuous casting of iron billets and blooms, liquid sprays extract up to 60% of the heat generated by the solidifying material [Mizikar 1970]. Thus, predicting the rate of heat removal by the spray is essential to creating the correct metallurgical properties of the product. Industrial sprays are also used for cooling turbine blades [McGinnis III
& Holman 1969], in a wide variety of heat exchanger designs [Incropera & DeWitt 1996] and even as an emergency cooling mechanism in severe nuclear reactor accidents [Collier & Hewitt 1987].

Fire extinguishment is another common application for liquid sprays. Water sprayed on a solid fuel fire is effective as an extinguishment method as the liquid removes heat from the fuel upon impact, thus cooling the burning material and slowing the production of volatiles [Rasbash 1986, Mike, Tinker & diMarzo 1997]. An added feature of water application as an extinguishment device is that the water vapour produced acts to dilute the oxygen available in the vicinity of the combusting material, acting to further slow the rate of combustion [Rasbash 1986]. Unfortunately, our understanding of the behaviour of droplets when impacting a burning solid is poor, which limits our ability to optimise the extinguishment of fires [Rasbash 1986, Harvie, Novozhilov, Kent & Fletcher 1999].

In some applications, prediction of the dynamical behaviour of the impacting liquid is more important than calculating the heat transfer resulting from droplet solid collisions. A commonly encountered domestic or industrial scenario is the hazardous explosive boiling of discrete droplets of water, oil or other liquids on a hot surface. An ability to predict the hydrodynamics of such incidents would be advantageous to human safety.

A similar scenario is the spilling of liquid nitrogen in a space module. Under low gravity conditions, the nitrogen would be repulsed from solid surfaces, causing the potentially hazardous situation of the module filling with small droplets of cold liquid [Qiao & Chandra 1996]. Again, an ability to predict the behaviour of such an incident allows the consequences to be studied.

The rate of evaporation of a droplet impacting a solid surface may also
be of interest. In a direct injection diesel engine, fuel is injected into the combustion chamber as a fine spray, a large proportion of which may impact the piston or adjacent wall of the combustion chamber [Arcoumanis & Chang 1993, Naber & Reitz 1988]. Under such conditions, it is advantageous for the fuel to vaporise rapidly, allowing fast and complete combustion of the fuel. The optimisation of such a process requires a thorough understanding of droplet impact behaviour.

Finally, the study of droplet impact behaviour on hot surfaces addresses some fundamental issues which are relevant to other solid-liquid heat transfer phenomenon. Questions relating to the possible contact between liquid and solid phases during high temperature boiling [Bradfield 1966], and the initiation of potentially hazardous vapour explosions [Inoue, Ganguli & Bankoff 1981, Corradini 1983, Fletcher & Theofanous 1997], have yet to be fully answered. The study of individual droplet impacts may well provide the necessary experimental data and simulation tools for this to be accomplished.

1.2 The Thermodynamic Regimes of Droplet Impact

When a droplet impacts a hot surface, the thermodynamic behaviour of the droplet is largely governed by the temperature of the solid and the initial temperature of the liquid. The different thermodynamic regimes that a droplet may experience can be roughly classified into four types, examples of which are shown in Figures 1.1 and 1.2.

When the temperature of the solid surface is only marginally higher than the saturation temperature of the liquid at atmospheric pressure, an impacting droplet will spread out over the solid, and single phase heat transfer will
Figure 1.1: Three normal gravity n-Heptane impacts conducted at the different initial solid temperatures of 24°C, 90°C and 104°C. The saturation temperature for n-Heptane at atmospheric pressure is 98.4°C. (Taken from Chandra & Avedisian [1991].)
result. During single phase heat transfer, no vapour is generated within the body of the liquid. Instead, liquid slowly vaporises from the exposed upper surface of the droplet. A droplet undergoing this mode of heat transfer is shown in Figure 1.1c.

At slightly higher solid temperatures, nucleate boiling impacts occur. During a nucleate boiling impact, the droplet again spreads out over the surface of the solid, however, after a short waiting period, vapour bubbles form, or nucleate, at the solid-liquid interface. Once generated, the bubbles move upward through the droplet, exiting through the upper surface of the droplet and causing a crater type appearance at the top surface of the liquid. Example nucleate boiling impacts are shown in Figure 1.2(a) and (b). Note that in Figure 1.2(b), the droplet appears to leave the solid after the initial impact.

At still higher solid temperatures, transition boiling impacts occur. During a transition boiling impact, the droplet contacts the surface of the solid, but the generation of vapour at the solid-liquid interface quickly acts to push the droplet away again. Conduction of heat through the vapour film separating the liquid and solid phases then commences, vaporising liquid at the lower surface of the droplet. The vapour forms a viscous layer which supports the droplet, however, during a transition impact, occasional contacts between the solid and liquid phases may continue to occur.

Film boiling impacts occur at the highest solid temperatures. During a film boiling impact, the generation of vapour at the lower surface of the droplet is so large that no contact between the solid and liquid phases occurs. Thus, the droplet is at all times separated from the impact surface by a layer of vapour, vaporised from the lower surface of the droplet. In general, the higher the initial temperature of the solid, and the lower the initial subcooling
Figure 1.2: Three normal gravity n-Heptane impacts conducted at the different initial solid temperatures of 160°C, 195°C and 205°C. (Taken from Chandra & Avedisian [1991].)
of the droplet, the more likely an impact will be of the film, rather than transition type.

Unlike nucleate and transition boiling impacts, film boiling impacts generally result in low heat transfer rates from the solid and low vapour production rates, as the vapour layer present during the impact acts to insulate the two colliding mediums. The example impact shown in Figure 1.2 (c) may be either a transition or film boiling impact — it is difficult to determine from photographs alone whether direct solid-liquid contact has occurred.

The focus of this study is impact droplet boiling regimes rather than stationary boiling regimes. Stationary droplet boiling regimes refer to the type of regime existing throughout the lifetime of a droplet, after the droplet has been placed gently on the surface of the solid. Impact boiling regimes refer to the type of regime occurring during the impact period, when the droplet collides with the solid surface with a finite velocity.

1.3 The Hydrodynamic Regimes of Droplet Impact

The Weber number is a ratio between the surface tension and inertial forces existing within a droplet. In droplet impact applications it is defined as

\[ We = \frac{\rho dv^2}{\sigma}, \]

where \( \rho \) is the density of the fluid, \( d \) is the diameter of the droplet, \( v \) is the initial velocity of the droplet normal to the impact surface, and \( \sigma \) is the surface tension of the fluid.

As shown by Wachters & Westerling [1966], the Weber number can be interpreted as a non-dimensional droplet impact velocity. Different ranges
of the Weber number correlate to different hydrodynamic impact regimes [Wachters & Westerling 1966].

For an initial Weber number of less than 30, droplets impact and expand over the surface, before recollecting and rebounding as one discrete mass of fluid. For Weber numbers between 30 and 80, droplets impact and expand over the surface as the lower velocity droplets do, but during the recollection and rebound stage, separate into a number of smaller, or satellite droplets. When droplets impact the solid with a Weber number greater than 80, droplet disintegration is observed during the initial expansion phase, and often only a small percentage of the initial droplet volume recollects after the impact into one major fluid body.

When the impact velocity of a droplet is of the same order as the speed of sound in the droplet liquid, the droplet dynamics can no longer be assumed to be incompressible, and the behaviour of the droplet becomes significantly different to the hydrodynamical regimes outlined above.

1.4 Scope and Structure of the Thesis

The objective of this thesis is to simulate droplet impacts which occur in the film boiling regime, and whose internal dynamical behaviour can be modelled using an incompressible fluid assumption. To accomplish this objective, a computational fluid dynamics code, termed BOUNCE, has been written. The code is composed of a Volume of Fluid (VOF) internal droplet dynamics algorithm, coupled to a separate one dimensional fluid dynamics algorithm for predicting the movement of vapour within the viscous vapour layer.

The thesis is structured in four main parts. The first part, composed of Chapters 2 and 3, reviews the experimental and computational efforts
that have taken place in this field to date. The second part, composed of Chapters 4, 5, 6 and 7, introduces the VOF internal dynamics code, outlines two new VOF advection algorithms which have been developed as part of this study, and concludes with a performance comparison of the different VOF advection methods that are currently in use.

The third part of the thesis is concerned with the viscous vapour layer code, used to model the fluid behaviour between the impacting droplet and solid surface. Chapter 8 outlines the assumptions used in this fluid flow region, while Chapter 9 details how these assumptions were implemented in a computational algorithm.

The fourth section validates the BOUNCE code by comparing impact simulations performed by the code against impacts documented in experimental studies. Chapter 10 is concerned with saturated droplet impacts, while Chapter 11 is concerned with impacts where the temperature of the droplet is initially subcooled. Chapter 11 also examines the range of impacts over which the BOUNCE code is applicable.

Finally, in Chapter 12, conclusions arising from the thesis are drawn, and areas for possible future research are outlined.
Chapter 2

Literature Review

In this chapter a review of the studies that have been conducted to date on the subject of film boiling droplet impacts is presented.

2.1 Steady State Droplet Film Boiling

The process of a stationary droplet film boiling on a solid surface is known as the Leidenfrost phenomenon. The process was named after Johann Gottlob Leidenfrost, who observed the film boiling of droplets on a teaspoon heated to a ‘red-hot’ temperature [Leidenfrost 1756, Wares 1963, Gottfried, Lee & Bell 1966]. An excellent review of the history of the Leidenfrost phenomenon is given by Gottfried et al. [1966].

While the present study is concerned with impacting droplets, a brief review of the works accomplished with Leidenfrost droplets to date is useful, as many of the processes involved in Leidenfrost boiling share similarities with the studied impact film boiling.

The Leidenfrost temperature is defined as the solid temperature which gives the longest time for a film boiling droplet to completely evaporate.
[Gottfried et al. 1966]. The temperature gives an indication of the boundary between the stationary transition and film boiling regimes.

Experimental investigations into the Leidenfrost phenomenon have been largely concerned with evaluating the Leidenfrost temperature, determining the time required by a film boiling droplet to completely vaporise, and determining the heat transfer rate from the solid to the liquid during the lifetime of a droplet. Examples of these studies include those by Cumo, Farelllo & Ferrari [1969], Temple-Pediani [1969-70], Emmerson & Snoek [1978], Michiyoshi & Makino [1978], Testa & Nicotra [1986] and Mosdorf & Poniewski [1987].

In a notable study by Baumeister & Simon [1973], semi-empirical correlations were found to predict the Leidenfrost temperature for a wide variety of conditions. Data used in developing the correlations included a large variety of different droplet liquids, solids, liquid and solid temperatures, and solid surface conditions. It was found that the Leidenfrost temperature is strongly dependent on the temperature and thermal properties of the solid surface, and also on the degree of subcooling of the liquid. Also, roughness and fouling of the solid surface were found to increase the Leidenfrost temperature.

Several analytical models have been developed to predict stationary droplet film boiling. The early study by Gottfried et al. [1966], applicable to small droplets, used the assumption that the shape of the droplet was spherical. Gottfried et al. [1966] also assumed that heat transfer to the liquid occurred by conduction through the viscous vapour layer and radiation from the solid to the lower surface of the droplet. Radiation exchange from the top surface of the droplet to the surroundings was included. The comparison between evaporation times calculated by the model and experimental results performed in the study was good.

At approximately the same time, Wachters, Bonne & van Nouhuis [1966]
presented an analysis of the evaporation rate of larger stationary droplets. In this study the lower surface of the droplet was assumed to be either flat, or slightly concave upwards, and heat transfer to the droplet was assumed to consist of one dimensional conduction through the vapour layer. The comparison of calculated evaporation rates against the presented experimental results was good, however, the authors stressed the importance of droplet oscillations, which occurred during the vaporisation process, on the droplet evaporation rate.

More recent studies [Sen & Law 1984, Nguyen & Avedisian 1987, Zhang & Gogos 1991] have concentrated on refining the small droplet model of Gottfried et al. [1966], by including more complex solutions to the vapour flow surrounding the evaporating droplets.

\section*{2.2 Film Boiling Droplet Impact Experimental Studies}

The number of experimental studies that have been performed on film boiling droplet impacts is large. They can be loosely classified by whether the initial temperature of the droplet is saturated or subcooled. Also, some studies tend to concentrate on droplet impact dynamics, while others on droplet impact heat transfer.

\subsection*{2.2.1 Saturated Droplet Impact Experimental Studies}

One of the first experimental investigations into the dynamics and heat transfer during film boiling droplet impacts was carried out by Wachters & Westerling [1966]. In this study, water droplets, initially at the saturation tempera-
ture corresponding to atmospheric pressure, impacted upon a highly polished gold surface.

Two series of experiments were performed in the Wachters & Westerling [1966] paper. In the first, the amount of heat transferred from the solid to the liquid was measured as a function of initial droplet velocity and initial solid temperature. In the second series of experiments, photographs were taken of individual droplet impacts, in order to determine the effect of initial droplet velocity and initial solid temperature on subsequent droplet dynamical behaviour.

Gold was chosen as the impact material in these experiments because it has a high thermal conductivity, and as a result, can be approximated as isothermal. The diameter of the droplets used was fixed at 2.3 mm. In the heat transfer series of experiments, the atmosphere in which the impacts were performed was composed solely of water vapour, however, for the droplet dynamics experiments, the composition of the atmosphere was not specified.

Wachters & Westerling [1966] showed from the heat transfer experiments that the proportion of droplet volume vaporised during each impact was generally very low. For the approximately 2 mm diameter droplets used in the study, the proportion measured was always less than 1%. Wachters & Westerling [1966] also predicted that film boiling impacts, that is impacts during which there is no direct solid-liquid contact, only occurred above an initial solid temperature of approximately 370°C in their experiments.

From the droplet dynamics series of experiments, Wachters & Westerling [1966] showed that there are at least three different regimes of film boiling droplet dynamical behaviour, depending on the initial Weber number of the impact. These regimes were discussed in Section 1.3 of the previous chapter. Another important dynamical observation made by Wachters & Westerling
[1966] was that the residence time of the droplet on the solid was roughly equal to the first-order vibration period of a freely oscillating droplet as given by Lord Rayleigh [1879]. Individual droplet dynamical experiments from the Wachters & Westerling [1966] study are used for the purposes of code validation in Chapters 3 and 10 of this thesis.

In a subsequent study, Wachters, Smulders, Vermeulen & Kleiweg [1966] extended the heat transfer work of Wachters & Westerling [1966] to mist droplets with an initial diameter of around 60 µm. It was again found that the total amount of heat transferred to droplets impacting the solid in the film boiling regime was low, and that in the cases examined, the proportion of initial droplet volume vaporised during each impact was always less than 1%. It was also observed in this study that the heat transfer rate to impacting droplets increased as the level of surface roughness of the impact material increased.

Another study to consider droplets initially heated to the saturation temperature was that by Groendes & Mesler [1982]. In this work, droplets having a large initial diameter of 4.7 mm were impacted on a surface whose initial temperature was 462°C. The dynamics of droplet impingement were recorded using a series of high speed photographs, and the surface temperature of the solid beneath the droplet during the impact was also measured.

In the Groendes & Mesler [1982] study, the droplets impacted the solid surface after falling a distance of 45 mm. Using the assumption that they experienced negligible air resistance during this brief free-fall period, the velocity of the droplets on impacting the solid was 0.94 m/s, corresponding to an impact Weber number of 68. The atmosphere through which the droplets fell and impacted the solid was composed solely of steam.

The solid material used for these impact experiments was quartz, so cho-
sen because it has a low thermal diffusivity. The lower the thermal diffusivity, the larger the temperature drop at the surface of the solid for a given heat flux. Thus, quartz was chosen to provide the most easily measured temperature drop at the surface of the solid during the impact.

The temperature of the solid surface was measured using a four lead, platinum film, resistance thermometer, applied in a Pi shape to the top of the quartz surface. The method of application and operation of this device is detailed in the droplet impact study by Groendes & Mesler [1982], and also in the earlier paper on boiling heat transfer by Foltz & Mesler [1970]. The effective area of the thermometer used in the droplet impact study was $0.25 \times 1 \text{ mm}^2$.

It was found by Groendes & Mesler [1982] that the solid surface temperature drop during a single impact was only of the order $20^\circ \text{C}$. Groendes & Mesler [1982] were also able to show that the solid surface temperature under an impacting droplet displayed large oscillations, suggesting that the height of the vapour layer beneath an impacting droplet is oscillatory. An impact presented by Groendes & Mesler [1982] is compared against a BOUNCE simulation of the same impact in Chapter 10.

2.2.2 Subcooled Droplet Impact Experimental Studies

A far greater number of experiments have been conducted using droplets whose temperature is significantly below the saturation temperature corresponding to atmospheric pressure. Indeed, in some studies, the authors have not specified the initial droplet temperature, thus not appreciating the dependence impact dynamics and heat transfer have on this initial condition.

McGinnis III & Holman [1969] conducted one of the first investigations into the heat transfer during subcooled droplet impacts. In their study, water,
ethanol and acetone droplets, having initial diameters within the range 2.5 to 3.4 mm, were impacted upon a polished copper surface. The impact velocities were high enough that the droplets were observed to ‘splatter’ on impact, this term defined in the study as meaning the droplets disintegrated on impact without significantly wetting the solid surface. The initial surface temperature of the solid was varied within the range 120°C to 420°C, and the liquid was initially at room temperature.

McGinnis III & Holman [1969] found that the maximum heat transfer to the impacting droplets occurred at an initial surface temperature in excess of the saturation temperature of the liquid of approximately 150°C for all three liquids studied. Also, by using a non-dimensional heat transfer rate devised using a similitude analysis, the authors were able to plot the heat transfer rate per droplet against the initial solid temperature in excess of the saturation temperature, and the data lie approximately on one curve. However, no attempt was made to vary the initial temperature of the droplets or to include the effects of droplet subcooling in the similitude analysis. Also, given the high impact velocities used and the low initial droplet temperatures, it is likely that direct solid-liquid contact occurred in all of the experiments conducted in this study.

In a subsequent study, Holman, Jenkins & Sullivan [1972] expanded on the results of McGinnis III & Holman [1969] by including droplets of initial diameter down to 0.5 mm in their experiments, and by using two new liquids in the experiments, Freon-11 and Freon-113. Again, however, no attempt was made to vary the initial temperature of the droplets, or examine the effect of this on subsequent heat transfer rates.

A study by Pedersen [1970] at approximately the same time investigated both the heat transfer and dynamical behaviour of subcooled droplets. The
diameters of these droplets ranged from 0.2 to 0.4 mm, and the impact velocities from 2 to 10 m/s. The initial temperature of the droplets was 22°C, giving an initial subcooling of 78°C.

It was found by Pedersen [1970] that within the ‘non-wetting’ regime, the heat transfer effectiveness of each impact, defined as the ratio of energy consumed during the impact to the total amount of energy required to vaporise the droplet, varied between approximately 3% and 25%. It was also found that the heat transfer effectiveness increased with droplet impact velocity, and that within the non-wetting regime, the effectiveness was only weakly dependent on the impact surface temperature.

A limited number of photographs of droplets during the impingement process were also taken by Pedersen [1970], and show clearly the difference in behaviour of impacts which are supposed to be in the wetting and non-wetting regimes. Unfortunately however, the time from impact at which these photographs were taken was not measured, so that they are not suitable for code validation purposes.

Seki, Kawamura & Sanokawa [1978] were the first investigators to attempt temperature measurements of the solid surface under the droplet during an impact. In these experiments, the surface temperature was measured using a thin film thermometer evaporated onto a stainless steel cylinder. The initial droplet temperature in these experiments was 20°C, the initial solid temperature varied from 130°C to 414°C, and experiments were conducted using both water and ethanol droplets.

It was found by Seki et al. [1978] that during the highest temperature impacts with water droplets, where non-wetting behaviour was assumed to occur, the surface temperature dropped substantially after the droplet initially impacted the solid. For example, during a water droplet impact con-
ducted with an initial solid temperature of 414°C, the temperature of the solid surface dropped to below 200°C during the first millisecond. Surface temperature drops measured under similar ethanol droplet impacts however, were substantially smaller.

A study by Makino & Michiyoshi [1984] attempted to measure the contact period between the solid and liquid phases when droplets impacted upon four different metal surfaces. The initial diameters of the water droplets in this experiment varied between 2.5 and 4.5 mm, and the liquid was initially at room temperature. The impact velocity of the droplets was not specified.

The solid-liquid contact period was estimated by measuring the resistance between the impacting droplet and the solid surface, by measuring the surface temperature under the solid, and by photographing the droplet impact and estimating the contact period from the resulting images. Makino & Michiyoshi [1984] found that in general the contact period decreased with increasing initial surface temperature, but that even when the initial surface temperature was above 300°C, an appreciable contact period of around 1 ms occurred after initial impact.

The experimental heat transfer study by Inada, Miyasaka & Nichida [1985] is particularly notable as the authors were able to vary the initial subcooling of the water droplets. In this experiment, droplets of initial diameter 4.0 mm fell from a height of 20 mm onto a polished copper surface. Heat transfer measurements were made using a series of thermocouples mounted into the depth of the impact material, and an inverse conduction analysis was applied to the resulting temperatures to calculate the heat transfer rate caused by the impact of droplets.

It was found by Inada et al. [1985] that within the nucleate, transition and film boiling regimes, a change in subcooling from 2°C to 88°C could in-
crease the heat transfer during an impact by over an order of magnitude, and also significantly change the droplet impact dynamical behaviour. Unfortunately however, the study focused primarily on the transition rather than film boiling regime of droplet impacts.

In a subsequent investigation, Inada & Yang [1993] studied the miniaturisation of water droplets which occurs within the transition boiling regime. Miniaturisation refers to the explosive boiling of droplets which results in the formation of numerous smaller droplets. The study showed that miniaturisation, and thus transition boiling, could occur until at least initial solid temperatures of 360°C, if the magnitude of droplet subcooling was high.

Two notable experimental droplet impact studies relevant to this thesis are the studies by Chandra & Avedisian [1991] and Qiao & Chandra [1996]. Both studies share similar features.

In the Chandra & Avedisian [1991] study, n-Heptane was used for the impacting droplets, while in the Qiao & Chandra [1996] study, impacts were performed using both n-Heptane and water droplets. The fluid n-Heptane was chosen by the experimentalists because it is often used as a hydrocarbon fuel, and because the physical properties of the fluid are qualitatively similar to those of water.

The initial temperature of the fluid in both experiments was fixed at ambient room temperature. As the boiling temperature of n-Heptane at atmospheric pressure is 371.58 K (see Appendix C, Section C.1.2), the initial subcooling of the n-Heptane droplets in these experiments was approximately 75 K. For the water impacts performed in Qiao & Chandra [1996], the initial fluid subcooling was approximately the same.

The initial diameter of the n-Heptane droplets used in the Chandra & Avedisian [1991] and Qiao & Chandra [1996] studies was fixed at 1.5 mm,
while the initial diameter of the water droplets used in the Qiao & Chandra [1996] study was 2.0 mm. The impact velocity of the droplets in the Chandra & Avedisian [1991] study was fixed at 0.93 m/s, giving an initial Weber number of approximately 43, while the impact velocity of the n-Heptane droplets in the Qiao & Chandra [1996] study was fixed at 0.8 m/s, giving an initial Weber number of approximately 32. The initial velocity of the water droplets in the Qiao & Chandra [1996] study was also 0.8 m/s, giving an initial Weber number of 17.8.

The impact material used in all experiments was stainless steel, the surface of which was polished to a mirror finish. The initial surface temperature of the steel was varied between room temperature and 320°C.

Some of the experiments conducted in the Qiao & Chandra [1996] study were performed in a low gravity environment. This was facilitated by mounting the impact surface on a test bed, capable of free falling during the droplet impact. The magnitude of gravity acting during these low gravity impacts was calculated in the study to be less than 1% of the nominal value.

Solid surface temperature measurements were also made in the Qiao & Chandra [1996] study. These measurements were performed using a commercially available chromel-alumel thermocouple. Further details of the type and mounting of the thermocouple are given in Qiao & Chandra [1996], although the position relative to the droplet axis and area over which the thermocouple operated are not specified.

An interesting result from these papers was that while transition, or perhaps film boiling impacts occurred with n-Heptane above an initial solid temperature of approximately 200°C, similar impacts could not be replicated with water, even when the initial temperature of the solid was raised to the limit of the experimental equipment, namely 320°C. It was found that even
when this high initial surface temperature was used for the water experiments, nucleate boiling impacts were still occurring. This phenomenon was explained by Qiao & Chandra [1996] to be a result of the different surface tension values between the two liquids, which affects the ability of the liquids to nucleate at the solid interface.

A particularly nice feature of both the Chandra & Avedisian [1991] and Qiao & Chandra [1996] studies is the clarity of the droplet impact photographs produced. In the Chandra & Avedisian [1991] study, a full set of images is provided which clearly display the behaviour of an impacting droplet from the single phase heat transfer regime, through to the high transition or low film boiling heat transfer regime. Some of these images are used for the purposes of code validation in Chapter [11] and were used for illustration purposes in the previous chapter.

The experimental study by Chen & Hsu [1995] was motivated by the need to determine local heat transfer rates during different boiling regimes. In this study, subcooled water droplets of initial diameters varying between 2.8 and 4.8 mm impacted a metal surface whose initial temperature ranged between 150°C and 550°C.

The impact material used in the Chen & Hsu [1995] study was Inconel 600. Inconel is an alloy of nickel, developed for use in high temperature applications, such as boiler tubes. Solid surface temperature measurements were taken in the Chen & Hsu [1995] study using an extremely fine thermocouple, mounted just below the surface of the impact material. Further details of the thermocouple and the mounting technique are given in Chen & Hsu [1995] and Lee, Chen & Nelson [1985].

A record of the impact dynamics of the water droplet was not taken during the Chen & Hsu [1995] experiments, however, solid surface temperatures
during the impacts were recorded, and the local heat transfer rates were calculated by the authors from the same data. These rates, determined under individual impacting droplets, were found to vary from approximately $1 \times 10^6 \, \text{W/m}^2$ during low temperature single phase heat transfer impacts, up to approximately $2 \times 10^7 \, \text{W/m}^2$ during nucleate boiling impacts with an initial solid temperature of $500^\circ\text{C}$. An impact presented in Chen & Hsu [1995] is used in Chapter 11 for the purposes of code validation.

Other experimental studies on subcooled impacting droplets have been performed, but few have used initial solid temperatures higher than those used by Chen & Hsu [1995]. Xiong & Yuen [1991] presented experimental data for a wide variety of liquids impacting a stainless steel plate at initial temperatures up to $600^\circ\text{C}$, although in this study, little information about the impact behaviour of the droplets was given. Chandra & Aziz [1994] experimented with liquid nitrogen droplets, although the emphasis in this study was on the droplet lifetime rather than droplet impact behaviour. Still more dynamics and heat transfer experiments, similar to those described above, include those by Takano & Kobayasi [1988], Naber & Farrell [1993], Labeish [1994], Hatta, Fujimoto, Takuda, Kinoshita & Takahashi [1995], Bernardin, Stebbins & Mudawar [1997a] and Bernardin, Stebbins & Mudawar [1997b].

In an interesting variation on the standard droplet impact dynamical study, Yao & Cai [1988], Choi & Hong [1990] and Anders, Roth & Frohn [1993] all performed experiments where the initial droplet velocity was not directed in a normal direction to the impact surface. It was found that in the ‘non-wetting’ heat transfer regime, the initial tangential droplet velocity was often not affected by droplet and wall interactions.
2.3 Film Boiling Droplet Impact Analytical Studies

There have been very few attempts to model film boiling droplet impacts to date, however, the models of Buyevich, Mankevich & Polotsky [1986] and Inada & Yang [1994] are notable exceptions. These models assume that the droplet can be represented as a constant volume, variable radius cylinder of fluid, which is supported above the solid surface by a quasi-steady viscous vapour layer. The assumptions used in these models, and a detailed analysis of their performance is given in the next chapter.

As part of their analysis of water spray cooling, Bolle & Moureau [1976] proposed a semi-empirically based model to predict individual droplet heat transfer rates. The model assumed that each impact consisted of two phases; a conduction phase, where direct solid-liquid contact occurs, and a film boiling phase, where heat is conducted across a vapour layer generated between the solid and droplet. The dynamics of the droplet impingement were estimated using empirically derived correlations.

By the nature of the first heat transfer assumption used by Bolle & Moureau [1976], namely that direct contact between the solid and liquid occurs, the Bolle & Moureau [1976] model is not relevant to simulating the same range of impacts that is the concern of this thesis. However, Bolle & Moureau [1976] did make the noteworthy observation that for initially subcooled droplets, the amount of energy transferred to the droplet during an impact is generally less than the sensible energy required to raise the temperature of the droplet to the saturation value. Bolle & Moureau [1976] also observed that during transition boiling impacts, a substantially greater amount of energy is extracted from the solid when direct solid-liquid contact
occurs than when the two phases are separated by a vapour film.

Other models concerned with hot surface impacts include the semi-empirical transition boiling model of Toda [1972], and the analytical nucleate boiling model of Savic [1958].

As is discussed in the next chapter, impact models which do not simulate the geometrical deformation of impacting droplets are not capable of predicting the hydrodynamic or thermodynamic behaviour of such impacts. Consequently, the BOUNCE model developed in this study is based on a Volume of Fluid (VOF) fluid dynamics algorithm, which allows an almost continuous simulation of droplet deformation.

To the knowledge of the author, a VOF method has not been previously used to model two phase droplet dynamics. However, VOF methods have been successfully used to predict the dynamics of droplets impacting surfaces within the single phase heat transfer regime and below. In the studies by Fukai, Shiiba, Yamamoto, Miyatake, Poulakakos, Megaridis & Zhao [1995] and Pasandideh-Fard, Qiao, Chandra & Mostaghimi [1996], the effect of changing the liquid surface tension and solid-liquid contact angles on droplet impact dynamics was investigated. The study by Pasandideh-Fard, Bhola, Chandra & Mostaghimi [1998] used the VOF method to model the solidification of tin droplets impacting a steel plate. Bussmann, Chandra & Mostaghimi [1997] investigated the dynamics of cold surface droplet impingement on irregularly shaped surfaces, using a fully three dimensional VOF code.
Chapter 3

The Cylindrical Droplet Impact Model

In this chapter a description of the cylindrical hot surface droplet impact model, as developed by Buyevich et al. [1986], is given, followed by an analysis of the performance of this model in predicting the dynamics of real hot surface droplet impacts. The Buyevich et al. [1986] model, which is also detailed in Buyevich & Mankevich [1995], uses very similar assumptions to those employed by Inada & Yang [1994] in their study of film boiling droplet impacts. Consequently, conclusions drawn in this chapter with regards to the performance and validity of the Buyevich et al. [1986] model can also be applied to the Inada & Yang [1994] model.

3.1 The Buyevich, Mankevich and Polotsky Droplet Impact Model

The most significant dynamical assumption used in the Buyevich et al. [1986] impact model is that the droplet has the shape of a cylinder or disc, whose
radius and height can vary with time, but whose volume remains constant. This assumption considerably simplifies the hydrodynamic analysis of the droplet. Partial justification for this assumption is that during a significant proportion of the duration in which a droplet is impinging a hot solid surface, the droplet is flattened across the surface of the solid, and the shape of the droplet is approximately that of a disc [Wachters & Westerling 1966, Chandra & Avedisian 1991]. However, further discussion of the accuracy of this assumption is deferred to Section 3.3.

Other dynamical assumptions used in the Buyevich et al. [1986] model are that there is no viscous energy dissipation within the body of the droplet and that there is no gravity force acting on the droplet. Also, the reactive force resulting from liquid vaporisation on the underside of the droplet is neglected. Thus, the only force which supports and repels the droplet is the pressure resulting from the viscous flow of vapour between the droplet and solid surface.

Figure 3.1 shows the geometry of the Buyevich et al. droplet, and the variables used to describe the geometry and position of the droplet relative to the solid surface. The dynamics of the droplet are calculated in two dimensional cylindrical coordinates.

Using the above assumptions, an energy balance conducted over the volume of the droplet yields [Buyevich et al. 1986],

\[ dE + dE^* + dU = fh, \]

where \( f \) is the force acting upwards on the base of the droplet from the viscous vapour layer, and \( h \) is the height of the vapour layer existing between the droplet and solid surface.
Figure 3.1: The geometry and main variables used to describe the Buyevich and Mankevich cylindrical droplet.

The external kinetic energy of the droplet is given by

\[ E = \frac{\rho V}{2} \left( \frac{dz_o}{dt} \right)^2, \]  

(3.2)

where \( z_o \) is the height of the centre of the cylinder above the solid surface, \( V \) is the volume of the droplet and \( \rho \) is the density of the droplet fluid. The internal kinetic energy of the droplet is given by

\[ E^* = \frac{\rho V}{4} \left( 1 + \frac{8}{3} \frac{R_o^6}{R^6} \right) \left( \frac{dR}{dt} \right)^2, \]  

(3.3)

where \( R_o \) is the initial equilibrium radius of the droplet, given by

\[ R_o = \left( \frac{V}{2\pi} \right)^{\frac{1}{3}}, \]  

(3.4)

and the surface energy of the droplet is given as

\[ U = 2\pi\sigma \left( R^2 + \frac{R_o^3}{R} \right). \]  

(3.5)
Here $\sigma$ is the surface tension of the droplet liquid. Equations (3.2)–(3.5) are taken directly from Buyevich et al. [1986].

A second energy relation for the droplet is derived from Newton’s Second Law [Buyevich et al. 1986],

$$dE = f\,d\tilde{z}_o = f\,(dh + dl),$$

where $l$ is defined as half the height of the droplet.

Substituting equations (3.2)–(3.5) into equations (3.1) and (3.6) gives two equations relating the dynamics of the droplet to the force generated within the vapour layer [Buyevich et al. 1986],

$$\rho V \left[ \frac{d^2 h}{dt^2} - 2\frac{R_o^3}{R^3} \frac{d^2 R}{dt^2} + 6\frac{R_o^3}{R^4} \left( \frac{dR}{dt} \right)^2 \right] = f \quad (3.7)$$

and

$$\rho V \left[ \left( 1 + \frac{8}{3} \frac{R_o^6}{R^6} \right) \frac{d^2 R}{dt^2} - \frac{8}{3} \frac{R_o^6}{R^7} \left( \frac{dR}{dt} \right)^2 \right] + 8\pi \sigma \left( R - \frac{R_o^3}{R^2} \right) = 4\frac{R_o^3}{R^3} f. \quad (3.8)$$

In calculating the magnitude of the force generated within the vapour layer, a number of assumptions are utilised. Firstly, the vapour layer height is assumed to be small compared with the maximum radius of the vapour layer. This allows several simplifications to be made to the heat transfer and flow within the vapour layer.

As the temperature gradient across the height of the vapour layer is far larger than the gradient along the length of the layer, heat transfer between the solid and liquid phases is dominated by one dimensional conduction. In the Buyevich & Mankevich [1995] model, the temperature of the liquid and solid surfaces are fixed at their initial values, so the heat transfer rate across
the vapour layer \([W/m^2]\) is given by

\[ q = k \frac{\Delta T}{h}, \]  

(3.9)

where \(\Delta T\) is the difference in initial temperatures between the solid and liquid phases, and \(k\) is the thermal conductivity of the vapour within the vapour layer. In the Buyevich et al. [1986] model, the initial temperature of the liquid is assumed to be equal to the saturation temperature of the droplet liquid at atmospheric pressure, and the atmosphere surrounding the droplet is assumed to contain only the vapour phase of the droplet fluid.

A second assumption that can be made from assuming that the length of the vapour layer is far greater than the height of the vapour layer is that the fluid momentum equations within the layer reduce to the one dimensional laminar viscous flow equation [Buyevich et al. 1986],

\[ \frac{\partial P}{\partial r} = \mu \frac{\partial^2 u}{\partial z^2}, \]  

(3.10)

where \(P\) is the pressure within the vapour layer, \(u\) is the radial fluid velocity within the layer, and \(\mu\) is the absolute viscosity of the vapour within the layer.

Assuming that the level of surface roughness within the layer is small compared with the height of the vapour layer, and assuming that no heat is conducted into the body of the droplet during the impact, equations (3.9) and (3.10) can be integrated, combined and rearranged to yield the force acting on the base of the droplet as [Buyevich et al. 1986],

\[ f = \frac{3\pi}{2} \nu k \Delta T \frac{R^4}{H_{fg,c} \ h^4}. \]  

(3.11)

Here \(\nu\) is the kinematic viscosity of the vapour phase, and \(H_{fg,c}\) is the latent heat of vaporisation of the liquid, corrected to account for the sensible heat of the vapour within the vapour layer.
Other assumptions used in the development of equation (3.11) are that the flow within the layer is quasi-stationary, and that there is no molecular interface temperature or velocity treatment at the liquid-vapour or solid-vapour interfaces. Also, it is assumed that there is no heat exchange from the droplet at surfaces other than the base of the droplet, and that there is no heat exchange to the droplet from radiation [Buyevich et al. 1986].

3.2 Validation of the BOING Code

A computational code, termed the BOING code, was written to solve equations (3.7), (3.8) and (3.11), thus replicating the work presented in Buyevich et al. [1986] and Buyevich & Mankevich [1995]. To solve these equations simultaneously, a semi-implicit extrapolation algorithm [Press, Teukolsky, Vetterling & Flannery 1992] as developed by Bader & Deuflhard [1983] was employed, being an efficient algorithm for integrating such sets of stiff differential equations.

Figure 3.2 shows two impact results calculated by Buyevich & Mankevich [1995] using the cylindrical droplet model, and Figures 3.3 and 3.4 the results calculated by BOING for the same impacts. The impacts are identified by two non-dimensional parameters; the Weber number, defined in Buyevich et al. [1986] as

\[
\text{We} = \frac{\rho R_o v_o^2}{8\sigma},
\]

(3.12)

where \(v_o\) is the initial velocity of the droplet, and the non-dimensional small parameter [Buyevich et al. 1986]

\[
\varepsilon = \left(3 \frac{\nu k \Delta T}{4 \sigma H_{fg,c} R_o}\right)^{\frac{1}{4}}.
\]

(3.13)

The two impacts shown in Figures 3.2 and 3.4 both have the small parameter value of \(\varepsilon = 0.1\), and differing Weber numbers of 1 and 3.
The results in Figures 3.2, 3.3 are presented in terms of a non-dimensional impact time,

$$\tau = \frac{t}{L_t},$$  \hspace{1cm} (3.14)

a non-dimensional droplet radius,

$$x = \frac{R}{R_o},$$  \hspace{1cm} (3.15)
and a non-dimensional vapour layer height,

\[ \eta = \frac{h}{L_h}. \]  

(3.16)

The scaling factors in equations (3.14) and (3.16) are given by [Buyevich et al. 1986],

\[ L_t = \left( \frac{\rho \Delta T}{8\sigma} \right)^{\frac{1}{2}} \]  

(3.17)

and

\[ L_h = \left( \frac{3\nu k}{4\sigma L} \right)^{\frac{1}{2}} R_o^{\frac{3}{4}}, \]  

(3.18)

respectively.

Comparing the BOING results of Figures 3.3 and 3.4 with the results of Figure 3.2, it is clear that BOING is correctly replicating the results of the Buyevich & Mankevich [1995] study, although the latter results appear to have been smoothed over a non-dimensional time period of the order 0.5.
Figure 3.4: Two droplet impacts calculated by the BOING code. The impacts have the non-dimensional parameters $We = 1, 3$ and $\varepsilon = 0.1$. Shown in this figure is the non-dimensional height of the vapour layer.

3.3 Performance of the Buyevich, Mankevich and Polotsky Impact Model

In order to assess the performance of the Buyevich et al. [1986] impact model, a simulation of a well-documented actual impact is performed and compared against experimental results. The actual impact chosen for comparison is the $We = 15$ impact presented in Wachters & Westerling [1966], as discussed in Section 2.2.1 of the previous chapter. The initial droplet velocity for this particular impact was 0.63 m/s, and Figure 3.5 shows photographic images of the impact.

Figure 3.6 shows a comparison between the dimensions of the experimental and simulated droplets during the Wachters and Westerling $We = 15$
Figure 3.5: The We = 15 impact performed by Wachters and Westerling.
(Taken from Wachters & Westerling [1966].)
Figure 3.6: The droplet height and radius during the $We = 15$ Wachters and Westerling water impact. The symbols indicate the maximum radius and height measured from experimental images, and the continuous lines are the predicted cylindrical droplet results from the Buyevich, Mankevich and Polotsky impact model.

impact. The cylindrical droplet model predicts the oscillation period of the droplet with reasonable accuracy, but predicts the amplitude of these oscillations poorly. In particular, the model over-predicts the maximum droplet radius during the impact by approximately 10%, under-predicts the minimum droplet radius during the impact by approximately the same amount, and under-predicts the maximum droplet height during the impact by almost 40%.
One important difference between the experimental and simulated results of Figure 3.6 is that the simulated droplet dimensions deform prior to and faster than the experimental droplet dimensions after the initial impact. This is a result of the simulated single degree of freedom droplet not being able to replicate the continuous deformation of the underside of the real droplet. Figure 3.7 shows the vapour layer height under the We = 15 Wachters and Westerling impact, calculated using the Buyevich et al. [1986] model. Although this height was not directly measured in the experimental study, some important observations can be made by comparing these results with the photographic images of the Wachters and Westerling impact shown in Figure 3.5.

Firstly, the simulated results indicate that at 8 ms after initial impact, the height of the vapour layer, or distance between the solid surface and underside of the droplet, is approximately 1 mm. The experimental images of Figure 3.5 show that the droplet is in close proximity to the solid until at least 12 ms after the initial impact, so the cylindrical model is significantly under-predicting the residence time of the droplet near the solid surface.

Also, the calculated vapour layer height given in Figure 3.7 predicts that there are vapour layer height peaks during the impact process of approximately 0.4 mm at 1.2 ms, and 0.3 mm at 4.7 ms, after initial impact. Considering that the radius of the Wachters and Westerling droplet is 2.3 mm, a distance of 0.3 mm should be clearly visible in the photographic images of Figure 3.5. However, no such vapour layer peaks appear at the times predicted by the Buyevich et al. [1986] model, and as a result, it is clear that the cylindrical model is not simulating the behaviour of the vapour layer beneath the impacting droplet. Also, as heat transfer rates during an impact are critically dependent on the height and geometry of this vapour layer, it
is clear that attempts to model impacting droplet heat transfer rates using the cylindrical impact model would be unsuccessful.

It is for these reasons that further development work that was conducted on the cylindrical droplet impact model is not presented. Instead, we now focus on the development of an impact model which is better able to replicate the continuous deformation of a droplet when it impacts a solid surface, and consequently, is more accurate in predicting the vapour layer and heat transfer behaviour beneath such droplets.

Figure 3.7: The vapour layer height, $h$, as calculated using the Buyevich, Mankevich and Polotsky droplet impact model for the $We = 15$ Wachters and Westerling impact.
Chapter 4

The Internal Droplet Dynamics Algorithm

BOUNCE uses two coupled fluid dynamics codes to calculate impacting droplet dynamics. The first is a Volume of Fluid (VOF) code, used to calculate internal droplet dynamics. The second is a code developed in this study to calculate fluid flow within the thin viscous vapour layer existing between the lower surface of the droplet and the hot solid surface. The vapour layer code works within the VOF code — pressures calculated within the vapour code are fed back to the VOF code in calculating droplet dynamics.

4.1 The Main Features of the BOUNCE VOF Code

The focus of this, and the next three chapters, is the structure and features of the internal droplet VOF code. The VOF coding used in BOUNCE was originally based on the well-documented SOLA-VOF code of Nichols, Hirt & Hotchkiss [1980], however modifications made during the development of
BOUNCE have left all but a few subroutines bearing a resemblance to their SOLA-VOF counterparts.

Some features of BOUNCE are based on the more recent RIPPLE-VOF code of Kothe, Mjolsness & Torrey [1994]. These features include the Incomplete Cholesky-Conjugate Gradient (ICCG) solution method to the Pressure Poisson Equation (PPE), and the implementation of the Continuum Surface Force (CSF) model of Brackbill, Kothe & Zemach [1992].

Other features have been developed in this work, including two new VOF advection algorithms, an implicit treatment of surface location, momentum advection and viscous stresses, and coupling of the droplet solution to the new viscous vapour layer code.

In this chapter we present an overview of the BOUNCE VOF code, and detail the features of the code. The following Chapters 5, 6 and 7 are devoted to describing and analysing the new VOF advection algorithms which are used by BOUNCE, as developed in this study. Details of the viscous vapour layer code are left until Chapter 8.

### 4.2 Volume of Fluid Codes

The internal droplet code uses a Volume of Fluid (VOF) function to locate fluid and void regions [Nichols et al. 1980]. The VOF function is defined as unity within fluid and zero elsewhere. In a numerical simulation, where the computational domain is divided into a discrete number of cells, the cell centered VOF function is defined as

$$F = \frac{\text{volume of fluid in cell}}{\text{total volume of cell}}.$$  \hspace{1cm} (4.1)

Following this definition, a cell entirely full of fluid has $F = 1$, a cell completely void of fluid has $F = 0$ and a partially full cell has an $F$ value between...
zero and one.

In practice roundoff errors resulting from floating point arithmetic cause $F$ to fluctuate around unity in fluid regions, and fluctuate around zero in void regions. For this reason, the $F$ function is given an upper and lower limit within partially full cells of $1 - \varepsilon_F$ and $\varepsilon_F$, respectively [Nichols et al. 1980]. Here $\varepsilon_F$ is a small number, in BOUNCE $\varepsilon_F = 1.0 \times 10^{-6}$. Any changes in volume caused by forcing the VOF function to these limits during a calculation are recorded and are typically below a fraction of a percent of the total fluid volume [Nichols et al. 1980].

### 4.3 Iterative Implicit Structure of the BOUNCE VOF Algorithm

#### 4.3.1 Basic Code Structure

The basic structure of BOUNCE is depicted schematically in Figure 4.1. As shown, the code consists of two main loops, termed the outer and inner loops. The outer loop is used to advance the fluid calculation in time, while the inner loop is used to iteratively converge the implicit treatment of surface location, vapour layer calculation, momentum advection and viscous stresses.

#### 4.3.2 The Main Subroutines used by BOUNCE

**ORIENTATE** and **SURFTEN**

As shown in Figure 4.1, subroutines **ORIENTATE** and **SURFTEN** are the first subroutines to be called at the beginning of each timestep. These subroutines calculate the position of the droplet surface, specified in terms of characteristic vectors normal to the droplet surface, and the curvature of the
PROGRAM BOUNCE

Initialise constants and open files for saving solution

Start time cycle loop

Call subroutine ORIENTATE and SURFEN to calculate no inerctions to droplet s surface and calculate droplet s surface cu nature at time $t$

Start implicit iterative loop

Call subroutine VFGRAD or subroutineVF STREAM to calculate fluid VOF position at time $t+\Delta t$ based on most recently calculated fluid velocities

Call subroutine VAPORLAYER to determine conditions within viscous vapor layer at time $t+\Delta t$

Call subroutineVFSETUP to implement VOF function boundary conditions to time $t+\Delta t$

Call subroutine SURFACEFORCE to calculate surface tension body force and apply viscous vapor layer pressure to lower droplet surface

Call subroutine ITLIDE to calculate momentum advection terms and viscous diffusion terms using most recently calculated velocities

Call subroutine VAPORLAYER to determine initial conditions within viscous vapor layer

Call subroutineVFSEGUP to implement initial VOF function boundary conditions

Call subroutine PPEICCG to solve the Pressure Poisson Equation (PPE) thus finding the internal droplet pressures at time $t+\Delta t$

Call subroutineACCEL to find the internal droplet velocities at time $t+\Delta t$ using the PPE pressures

Loop through all VOF cells

Calculate maximum relative pressure change per cell

Implicit iterative converge rate criterion: Check if $|\text{error}| < \text{pepsi}$

Satisfied

Call subroutine DELTADJ to calculate the timestep duration for the next timestep

Was the timestep successful?

Yes

Advance time: $t = t + \Delta t$

Save solution if required

Is computation finished?

Yes

Update all variables to new time

No

Not Satisfied

Figure 4.1: A schematic showing the basic code structure of BOUNCE.
droplet surface. The vector normals are used by the subroutines $VFGRAD$ or $VFSTREAM$ in determining the free surface advection during each timestep, while the surface curvatures are used by subroutine $SURFACE$ to calculate the surface tension forces acting on the droplet over each timestep. As the surface curvature is calculated using the position of the droplet at the beginning of the timestep, surface tension calculations in BOUNCE are explicit. More details of the surface tension algorithm are given in Section 4.5.1.

$VFGRAD$ and $VFSTREAM$

Subroutines $VFGRAD$ or $VFSTREAM$ are the first to be called within the implicit loop. Either subroutine is used, depending on the method chosen to advect fluid free surfaces. Subroutine $VFGRAD$ uses the Defined Donating Region algorithm to advect free surfaces, while subroutine $VFSTREAM$ uses the Stream algorithm. Details of these algorithms are given in Chapters 5 and 6, respectively. The velocities used by the subroutines to determine cell boundary fluid fluxes during the timestep are the most recently calculated, and consequently represent end of timestep values. Thus, surface location is treated implicitly in BOUNCE.

$VAPORLAYER$

Subroutine $VAPORLAYER$, which calculates the pressure force acting on the droplet from the viscous vapour layer, uses the end of timestep droplet position calculated by $VFGRAD$ or $VFSTREAM$ to calculate the geometry of the vapour layer. Thus, vapour layer pressures, like the position of the droplet surface, are calculated implicitly. This is an important factor, as small variations in the position of the droplet, and consequently the height of the viscous vapour layer, can cause large variations in the pressure acting on
the lower surface of the droplet. Under such unstable conditions, an explicit code would require an impractically large number of timesteps to calculate a single droplet impact. It was primarily for this reason that an iterative implicit technique was developed in BOUNCE. As previously mentioned, details of the secondary fluids algorithm used by VAPORLAYER are given in Chapters 8 and 9.

\textit{SURFACEFORCE, ITILDE, VASSEMBLE, PPEICCG} and \textit{ACCEL}

Other subroutines called during the implicit loop are \textit{SURFACEFORCE}, \textit{ITILDE, VASSEMBLE, PPEICCG} and \textit{ACCEL}. Subroutine \textit{SURFACEFORCE} calculates the surface tension volume force terms using the surface curvature values calculated by \textit{SURFTEN}. Subroutine \textit{ITILDE} calculates the momentum advection and viscous stress terms used in solving the internal droplet flow Navier-Stokes equations. As fluid velocities used to calculate these terms correspond to end of timestep values, both momentum advection and viscous stresses are treated implicitly. Details of algorithms used by \textit{ITILDE} are given in Section 4.4.3. Subroutines \textit{VASSEMBLE, PPEICCG} and \textit{ACCEL} are responsible for integrating the Navier-Stokes equations to find end of timestep pressures and velocities. The technique used to accomplish this is outlined in Section 4.4.

4.3.3 Implicit Convergence Criterion

Convergence of the iterative implicit algorithm is assumed when variations in internal fluid pressures between successive iterations fall below a specified
level. Specifically, this is when,

\[
\left| \frac{P - P^*}{P_{\text{max}}} \right|_{\text{all fluid cells}} \leq \varepsilon_P,
\]

where asterisk values represent those calculated during the last implicit treatment loop iteration, and \( \varepsilon_P \) is a user set implicit pressure iteration convergence comparator. Typically, \( \varepsilon_P \) is set to \( 1 \times 10^{-2} \) for droplet impact calculations.

A disadvantage of iterating the pressure solution to achieve implicit convergence is that all variable values output from any algorithm used in the iterative calculations must be continuous over the range of possible input variable values for that algorithm. That is, an infinitely small variation in any input parameter to any algorithm cannot produce a discrete change in the output values of that algorithm. If such discrete changes in variables were to occur, then the functional link between pressures output after one iteration and pressures input to the next iteration would be broken, and in such a case, the implicit pressure solution could not converge.

One specific implication of this iterative convergence criteria is that the number of cells which contain fluid during any implicit iteration must remain constant during the iteration for convergence to be achieved. This is to ensure that large discrete changes in the pressure field solution do not occur as a result of perhaps only small changes in the location of the fluid free surface.

In BOUNCE, this is achieved by identifying any cells which after three implicit iterations, continue to alternate between containing only void, and containing a small proportion of fluid. Once identified, these cells are artificially filled during subsequent iterations to give a minimum VOF level of \( F = \varepsilon_F \). Thus, after three implicit iterations, the number of cells containing any fluid remains constant, and implicit convergence can be achieved.
4.3.4 Variable Timestep

In a similar fashion to the SOLA-VOF code [Nichols et al. 1980], BOUNCE uses a variable size timestep in calculating droplet dynamics. Timestep adjustment is accomplished in subroutine \textit{DELTADJ}. The criteria considered when decreasing the timestep are:

1. Implicit iterative convergence. The timestep is decreased by 5\% if between four and nine iterations were required to converge the implicit solution, or 40\% if over ten iterations were required to converge the implicit solution.

2. VOF convection algorithm accuracy. As detailed in Chapters 5 and 6, each of the VOF convection algorithms are subject to an accuracy criterion. If the relevant criterion is likely to be violated during the next timestep, the timestep value is decreased.

3. Surface tension algorithm stability. The surface tension algorithm, being explicit, is subject to a stability criterion as detailed in Section 4.5.1. Again, if this criterion is likely to be violated during the next timestep, the timestep value is decreased.

4. Viscous vapour layer accuracy. The viscous vapour layer algorithm is subject to an air volume fraction Courant type accuracy criterion, and an explicit liquid temperature convection stability criterion, both of which are detailed in Chapter 9. As in the above cases, if either of these criteria are likely to be violated during the next timestep, the timestep value is decreased.

If none of the above conditions have been violated, then Subroutine \textit{DELTADJ} increases the timestep by a small percentage. In this manner,
the computational timestep is continuously changing to ensure the highest computational efficiency.

4.3.5 Failed Iterations

In order for a solution timestep to be accepted, certain accuracy and stability criteria have to be met. The individual criteria are discussed alongside each relevant computational algorithm. If one or more criteria are violated during a timestep, or alternatively if over thirty implicit iterations have been performed and implicit solution convergence has not been achieved, the timestep value is halved, all variables are reset to their beginning timestep values, and the timestep is re-computed.

4.4 PPE-ICCG solution

4.4.1 Pressure Poisson Equation

An incompressible fluid dynamics treatment is valid within the droplet, as maximum fluid velocities in the present region of study are considerably less than the speed of sound in water. Consequently, fluid flow within the droplet is governed by the incompressible continuity equation

\[ \nabla \cdot \mathbf{V} = 0, \quad (4.3) \]

and the incompressible Navier-Stokes equation

\[ \frac{\partial \mathbf{V}}{\partial t} + \nabla \cdot (\mathbf{V} \otimes \mathbf{V}) = -\frac{1}{\rho} \nabla P + \frac{1}{\rho} \nabla \cdot \tau + \mathbf{g} + \frac{1}{\rho} \mathbf{F}_b. \quad (4.4) \]

Here \( \mathbf{V} \) is the fluid velocity vector, \( P \) the scalar pressure, \( \tau \) the viscous stress tensor, \( \mathbf{g} \) the vector acceleration due to gravity and \( \mathbf{F}_b \) a vector body...
force. As the BOUNCE VOF code is a single phase algorithm, the density in equation (4.4) is defined as

\[ \rho = F \rho_f, \]  

(4.5)

where \( \rho_f \) is the density of the droplet liquid.

As previously discussed, surface location, momentum advection and viscous dissipation terms used in BOUNCE are calculated implicitly, so when differenced in time, equation (4.4) becomes

\[ \frac{V^{n+1} - V^n}{\delta t} = -\nabla \cdot (V \otimes V)^{n+1} - \frac{1}{\rho^{n+1}} \nabla P^{n+1} + \frac{1}{\rho^{n+1}} \nabla \cdot \tau^{n+1} + g + \frac{1}{\rho^{n+1}} F_{b}^{n,n+1}. \]  

(4.6)

Terms with the \( n + 1 \) superscript are evaluated at the end of the timestep, terms with the \( n \) superscript are evaluated at the beginning of the timestep and \( \delta t \) is the discrete timestep through which the solution is advanced. The vector body force comprises terms calculated at both the beginning and end of the timestep.

The continuity equation (4.3) and Navier-Stokes equation (4.6) are combined in BOUNCE using the method of Nichols et al. [1980] and Kothe et al. [1994], to form the so called Pressure Poisson Equation (PPE). Here, a new mid-timestep velocity, \( \tilde{V} \), is defined such that

\[ \frac{\tilde{V} - V^n}{\delta t} = -\nabla \cdot (V \otimes V)^{n+1} + \frac{1}{\rho^{n+1}} \nabla \cdot \tau^{n+1} + g + \frac{1}{\rho^{n+1}} F_{b}^{n,n+1}. \]  

(4.7)

Thus, the mid-timestep velocity, \( \tilde{V} \), is composed of inertial, momentum advection, viscous dissipation, gravitational and body force terms.

Combining equations (4.6) and (4.7) gives,

\[ \frac{V^{n+1} - \tilde{V}}{\delta t} = -\frac{1}{\rho^{n+1}} \nabla P^{n+1}. \]  

(4.8)
Taking the divergence of both sides of this equation and rearranging gives the Pressure Poisson Equation

\[ \nabla \cdot \left( \frac{1}{\rho^{n+1}} \nabla P^{n+1} \right) = \frac{\nabla \cdot \tilde{V}}{\delta t}, \quad (4.9) \]

where use has been made of the continuity equation (4.3) evaluated at time \( n+1 \). Equation (4.9), along with the appropriate free surface and wall boundary conditions, is formulated as a finite difference equation and used in BOUNCE to determine the change in the pressure field within the droplet during a discrete timestep. Once calculated, the new pressure field is used in equation (4.8) to determine the new end of timestep fluid velocities.

### 4.4.2 Variable Placement and Computational Mesh Geometry

Similar to the SOLA-VOF code [Nichols et al. 1980], BOUNCE uses a staggered rectangular grid in solving equation (4.9). As shown in Figure 4.2, pressure terms are located at the centre of each computational cell, while the horizontal and vertical velocity components are located separately at each of the cell borders. The volume of fluid fraction represents an average determined over the volume of each cell and for convenience it is located at the cell centre.

Note that in Figure 4.2 subscripts to variables indicate the position of that variable. Integer subscripts imply that the quantity is located at the cell centre, while an integer plus or minus a half indicates the boundary above or below the relevant cell centre, respectively.

While droplet impacts are calculated in cylindrical coordinates, BOUNCE allows solutions to equation (4.9) in either two dimensional Cartesian or two dimensional cylindrical coordinates. In the remainder of this study, the
variables $x$ and $u$ refer to displacement and velocity in the horizontal direction when the context implies Cartesian coordinates, or displacement and velocity in the radial direction when the context implies cylindrical coordinates. The variables $y$ and $v$ refer to displacement and velocity in the vertical direction in the context of either Cartesian or cylindrical coordinates.

The total number of VOF cells generated in the $x$ direction is specified by $i_{\text{max}}$, while the total number of VOF cells generated in the $y$ direction is specified by $j_{\text{max}}$. Two columns of dummy cells, located to the left and right of the computational domain, are used to apply boundary conditions at the right and left boundaries of the computational domain, respectively. Thus, the number of columns of computational cells contained within the domain is $i_{\text{max}} - 2$, and the leftmost cells within the computational domain are identified by the subscript $i = 2$. Similarly, rows of dummy cells are generated above and below the computational domain, so that the number of rows of cells contained within the domain is $j_{\text{max}} - 2$, and the lowest cells existing within the computational domain are specified by the subscript $j = 2$. 
The mesh generation algorithm used in BOUNCE is taken directly from the SOLA-VOF code [Nichols et al. 1980]. The algorithm allows for variable rectangular cell sizes and embedded submeshes, although droplet impacts are usually calculated using square cells of uniform dimensions. Further details of the algorithm are given in Nichols et al. [1980].

We now divert our attention to the individual terms required in evaluating the mid-timestep velocity, $\tilde{V}$.

4.4.3 Momentum Advection and Viscous Stresses

The momentum advection and viscous dissipation algorithms used in evaluating $\tilde{V}$ were developed by Nichols et al. [1980], and are taken from the SOLA-VOF code [Nichols et al. 1980]. Importantly in BOUNCE however, both these terms are evaluated implicitly using end of timestep velocities and surface positions, whereas in SOLA-VOF, the terms were evaluated explicitly.

Momentum Advection

In either two dimensional Cartesian coordinates or two dimensional cylindrical coordinates the momentum advection terms can be written as

$$ \nabla \cdot (V \otimes V) = \left( \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right) \left( \begin{array}{cc} u^2 & uv \\ uv & v^2 \end{array} \right). \tag{4.10} $$

Performing the expansion and differentiating we find

$$ \nabla \cdot (V \otimes V) = \left( u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} \right) \mathbf{i} + \left( u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} \right) \mathbf{j}, \tag{4.11} $$

where use has been made of the two dimensional continuity equation (4.3), namely

$$ \nabla \cdot V = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{4.12} $$
In BOUNCE, equation (4.11) is differenced as a non-conservative equation. To illustrate the method, an example is taken from [Nichols et al. 1980] for the determination of $u \frac{\partial u}{\partial x}$:

$$u \frac{\partial u}{\partial x} = \left( u_{i+\frac{1}{2},j} \right) \left[ \delta x_{i+1} DUL + \delta x_i DUR 
+ \alpha sgn(u_{i+\frac{1}{2},j}) (\delta x_{i+1} DUL - \delta x_i DUR) \right], \tag{4.13}$$

where

$$DUL = \frac{\left( u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j} \right)}{\delta x_i}, \tag{4.14}$$

$$DUR = \frac{\left( u_{i+\frac{3}{2},j} - u_{i+\frac{1}{2},j} \right)}{\delta x_{i+1}} \tag{4.15}$$

and

$$\delta x_\alpha = \delta x_{i+1} + \delta x_i + \alpha sgn(u_{i+\frac{1}{2},j}) (\delta x_{i+1} - \delta x_i). \tag{4.16}$$

Referring to equation (4.13), we see that the velocity gradient $\frac{\partial u}{\partial x}$ is determined using a combination of a first order upwind scheme and a second order central difference scheme. The extent to which each scheme is used is determined by a user set variable $\alpha$, where $0 \leq \alpha \leq 1$. When $\alpha = 1$, equation (4.13) reduces to a first order upwind scheme, while when $\alpha = 0$, equation (4.13) reduces to a second order central difference scheme. In BOUNCE a value of 0.7 was used for $\alpha$, as this value was found to give a good compromise between momentum advection stability and accuracy.
Viscous Stresses

Assuming Newtonian flow, the viscous terms of equation (4.4) in two dimensions become

\[
\frac{1}{\rho} \nabla \cdot \tau = \nu \left\{ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \xi \left( \frac{1}{x} \frac{\partial u}{\partial x} - u \frac{1}{x^2} \right) \right\} i + \nu \left\{ \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\xi u}{x} \frac{\partial u}{\partial x} \right\} j,
\]

(4.17)

where \(\xi = 0\) in Cartesian coordinates and \(\xi = 1\) in cylindrical coordinates [Nichols et al. 1980]. The terms of equation (4.17) are implemented using standard finite difference expressions on a variable mesh size grid — the interested reader is referred to the code listing in Appendix E for further details.

We now examine the method used to integrate the Pressure Poisson Equation.

### 4.4.4 Numerical Solution

Once \(\tilde{\mathbf{V}}\) has been determined, equation (4.9) can be integrated to find the new pressure field solution. Noting that \(\rho^{n+1} = \rho_f F^{n+1}\), equation (4.9) becomes

\[
\nabla \cdot \left( \frac{1}{F^{n+1}} \nabla P^{n+1} \right) = \rho_f \frac{\nabla \cdot \tilde{\mathbf{V}}}{\delta t}.
\]

(4.18)

Performing the vector expansions,

\[
\frac{1}{x^\xi} \frac{\partial}{\partial x} \left( x^\xi \frac{\partial P^{n+1}}{\partial x} \right) + \frac{\partial}{\partial y} \left( \frac{1}{F^n} \frac{\partial P^{n+1}}{\partial y} \right) = \rho_f \frac{\partial}{\delta t} \left\{ \frac{1}{x^\xi} \frac{\partial}{\partial x} \left( x^\xi \bar{w} \right) + \frac{\partial}{\partial x} \left( \bar{v} \right) \right\},
\]

(4.19)

where

\[
\tilde{\mathbf{V}} = \bar{u} i + \bar{v} j,
\]

(4.20)

and as before, \(\xi = 0\) in Cartesian coordinates and \(\xi = 1\) in cylindrical coordinates.
In finite difference form, equation (4.19) can be evaluated using a five cell kernel as

\[ A_{i,j} P_{i,j} + B_{U,i,j} P_{i+1,j} + B_{L,i,j} P_{i-1,j} + C_{U,i,j} P_{i,j+1} + C_{L,i,j} P_{i,j-1} = S_{i,j}, \] (4.21)

where the coefficients are given by

\[ A_{i,j} = -\frac{1}{x_i^\xi \delta x_i} \left\{ \frac{x_{i+\frac{1}{2}}^\xi}{F_{i+\frac{1}{2},j} \delta x_{i+\frac{1}{2}}} + \frac{x_{i-\frac{1}{2}}^\xi}{F_{i-\frac{1}{2},j} \delta x_{i-\frac{1}{2}}} \right\} - \frac{1}{\delta y_j} \left\{ \frac{1}{F_{i,j+\frac{1}{2}} \delta y_{j+\frac{1}{2}}} + \frac{1}{F_{i,j-\frac{1}{2}} \delta y_{j-\frac{1}{2}}} \right\}, \] (4.22)

\[ B_{U,i,j} = \frac{x_{i+\frac{1}{2}}^\xi}{x_i^\xi \delta x_i F_{i+\frac{1}{2},j} \delta x_{i+\frac{1}{2}}}, \] (4.23)

\[ B_{L,i,j} = \frac{x_{i-\frac{1}{2}}^\xi}{x_i^\xi \delta x_i F_{i-\frac{1}{2},j} \delta x_{i-\frac{1}{2}}}, \] (4.24)

\[ C_{U,i,j} = \frac{1}{\delta y_j F_{i,j+\frac{1}{2}} \delta y_{j+\frac{1}{2}}}, \] (4.25)

\[ C_{L,i,j} = \frac{1}{\delta y_j F_{i,j-\frac{1}{2}} \delta y_{j-\frac{1}{2}}}, \] (4.26)

and

\[ S_{i,j} = \rho F \frac{\delta}{\delta t} \left\{ \frac{x_{i+\frac{1}{2}}^\xi \tilde{u}_{i+\frac{1}{2},j} - x_{i-\frac{1}{2}}^\xi \tilde{u}_{i-\frac{1}{2},j}}{x_i^\xi \delta x_i} + \frac{\tilde{v}_{i,j+\frac{1}{2}} - \tilde{v}_{i,j-\frac{1}{2}}}{\delta y_j} \right\}. \] (4.27)

Note that variables in the above equations with non-integer subscripts refer to boundary position values, thus

\[ F_{i+\frac{1}{2},j} = \frac{\delta x_i F_{i+1,j} + \delta x_{i+1} F_{i,j}}{\delta x_i + \delta x_{i+1}}, \] (4.28)

\[ F_{i,j+\frac{1}{2}} = \frac{\delta y_j F_{i,j+1} + \delta y_{j+1} F_{i,j}}{\delta y_j + \delta y_{j+1}}, \] (4.29)
\[ \delta x_{i+\frac{1}{2}} = x_{i+1} - x_i, \quad (4.30) \]

and

\[ \delta y_{j+\frac{1}{2}} = y_{j+1} - y_j. \quad (4.31) \]

### 4.4.5 Boundary Conditions

In order to solve equation (4.21) throughout the computational domain, pressure boundary conditions are required at fluid free surfaces and at computational domain boundaries.

**Free Surfaces**

When a cell or a neighbour of a cell is void of fluid, the coefficients of equations (4.22)–(4.27) must be modified to account for the existence of a free surface. The only boundary condition at a free surface is

\[ P_{\text{boundary}} = P_{\text{ambient}}, \quad (4.32) \]

noting that surface tension and pressures in excess of atmospheric pressure resulting from the viscous vapour layer are modelled using a fluid interface volume force.

In BOUNCE, internal fluid pressures are gauge pressures, and are thus measured relative to ambient atmospheric pressure, so we have

\[ P_{\text{void}} = P_{\text{ambient}} = 0. \quad (4.33) \]

Equation (4.18) has the fortunate property, that for finite values of \( \vec{V} \) we have

\[ P \to \text{constant as } F \to 0. \quad (4.34) \]

Thus, we can model a void cell in the VOF code by simply specifying the pressure of that cell as equal to the ambient pressure, or zero.
A Cell is Void  Applying this principle to equation (4.21), in the case of a void cell \( i, j \), the coefficients specified by equations (4.22)–(4.27) become,

\[ A_{i,j} = 1 \]  
(4.35)

and

\[ B_{U,i,j} = B_{L,i,j} = C_{U,i,j} = C_{L,i,j} = S_{i,j} = 0. \]  
(4.36)

This gives

\[ P_{i,j} = 0. \]  
(4.37)

A Neighbour Cell is Void  If a neighbour cell of an examined cell is void of fluid, then the coefficients referring to the void neighbour cell used in equation (4.21) must be set to zero. For example, if the neighbour of cell \( i, j \) at position \( i + 1, j \) is void, then equation (4.23) is modified to

\[ B_{U,i,j} = 0, \]

while the remaining coefficients \( B_{L,i,j}, C_{U,i,j}, C_{L,i,j} \) and \( S_{i,j} \) for cell \( i, j \) remain unchanged.

Computational Domain Boundaries  
Two types of boundary conditions are allowed at computational domain boundaries in BOUNCE. Symmetry boundaries, existing at the centreline of a cylindrical calculation for example, are formulated implicitly. Outflow boundaries, conceived primarily to allow fragments of fluid that may detach from the main fluid droplet to exit the computational domain, are formulated explicitly.
**Symmetry Boundaries**  At a symmetry boundary, fluid velocity tangential to the boundary surface is allowed, but fluid velocity normal to the surface is zero.

To illustrate the BOUNCE formulation, we consider the case of a symmetry boundary at $x = 0$, as in the centreline of a cylindrical coordinate calculation. Taking the $x$ component of equation (4.8) at the boundary, we have

$$
\frac{u_{n+1}^{1+\frac{1}{2},j} - \tilde{u}_{1+\frac{1}{2},j}}{\delta t} = -\frac{1}{\rho_{n+1}^{1+\frac{1}{2},j}} \frac{\partial P}{\partial x_{1+\frac{1}{2},j}},
$$

(4.38)

as $x_{1+\frac{1}{2}} = 0$ represents the position of the symmetry boundary, or centreline. Differencing the pressure derivative, and noting that the normal velocity to the boundary, $u_{n+1}^{1+\frac{1}{2},j}$, is zero, gives

$$
P_{1,j} = P_{2,j} - \frac{\delta x_{1+\frac{1}{2}} \rho_{n+1}^{1+\frac{1}{2},j} \tilde{u}_{1+\frac{1}{2},j}}{\delta t}.
$$

(4.39)

Substituting this value into equation (4.21) for the pressure in the dummy cell $P_{1,j}$ leads to the corrected coefficients for the PPE equation in cells $i = 2$ of

$$
A_{2,j}^* = A_{2,j} + B_{L,2,j},
$$

(4.40)

$$
S_{2,j}^* = S_{2,j} + B_{L,2,j} \frac{\delta x_{1+\frac{1}{2}} \rho_{F}^{1+\frac{1}{2},j} \tilde{u}_{1+\frac{1}{2},j}}{\delta t},
$$

(4.41)

and

$$
B_{L,2,j}^* = 0,
$$

(4.42)

where asterisks indicate the new PPE coefficients.

Once the PPE equation has been solved implicitly for the new pressure field, pressures in the dummy border cells located at $i = 1$ are set using

$$
P_{1,j} = P_{2,j}
$$

(4.43)

for all $j$. 

56
Outflow Boundaries  Outflow boundaries are typically applied at the right and top boundaries of the solution space in droplet impact simulations, to allow any unattached fragments of fluid to exit the computational region.

For the purpose of illustration, consider again equation (4.8), but this time at the right computational domain boundary,

\[
\frac{u_{i_{\text{max}}-\frac{1}{2},j}^{n+1} - \tilde{u}_{i_{\text{max}}-\frac{1}{2},j}^n}{\Delta t} = -\frac{1}{\rho_{i_{\text{max}}-\frac{1}{2},j}^{n+1}} \frac{\partial P}{\partial x_{i_{\text{max}}-\frac{1}{2},j}}. \tag{4.44}
\]

Note that the subscript \(i = i_{\text{max}} - 1\) represents the rightmost cells in the calculated fluid domain, and consequently, the outflow boundary is located at \(x_{i_{\text{max}}-\frac{1}{2}}\).

At an outflow boundary we assume that the gradient of the velocity normal to the boundary is zero. This assumption can be explicitly differenced as

\[
u_{i_{\text{max}}-\frac{1}{2},j}^{n+1} - u_{i_{\text{max}}-\frac{3}{2},j}^n = 0, \tag{4.45}\]

where the difference is centred at the centre of cell \(i_{\text{max}} - 1, j\). Substituting equation (4.45) into equation (4.44), differencing the pressure term and rearranging we have,

\[
P_{i_{\text{max}},j} = P_{i_{\text{max}}-1,j} - \frac{\Delta x_{i_{\text{max}}-\frac{1}{2}} \rho_{i_{\text{max}}-\frac{1}{2},j}^{n+1}}{\Delta t} \left( u_{i_{\text{max}}-\frac{3}{2},j}^n - \tilde{u}_{i_{\text{max}}-\frac{1}{2},j}^n \right). \tag{4.46}\]

Substituting again into equation (4.21) gives the correct PPE coefficients for the cells in the column \(i = i_{\text{max}} - 1\),

\[
A^*_{i_{\text{max}}-1,j} = A_{i_{\text{max}}-1,j} + B_{U,i_{\text{max}}-1,j}, \tag{4.47}
\]

\[
S^*_{i_{\text{max}}-1,j} = S_{i_{\text{max}}-1,j} + B_{U,i_{\text{max}}-1,j} \frac{\Delta x_{i_{\text{max}}-\frac{1}{2}} \rho F_{i_{\text{max}}-\frac{1}{2},j}}{\Delta t} \left( u_{i_{\text{max}}-\frac{3}{2},j}^n - \tilde{u}_{i_{\text{max}}-\frac{1}{2},j}^n \right). \tag{4.48}\]
and

\[ B_{U, i_{\text{max}}-1,j}^* = 0, \quad (4.49) \]

where again, asterisks denote the new PPE coefficients.

### 4.4.6 Incomplete Cholesky Conjugate Gradient PPE Matrix Inversion

Equation (4.21), together with the pressure boundary conditions outlined in Section 4.4.5, specifies a set of equations which can be solved to find the pressure field throughout the computational domain. The coefficients of this set of linear equations can be written as a square, positive definite, symmetric and band diagonal matrix of side dimension \((i_{\text{max}} - 2) \times (j_{\text{max}} - 2)\) [Kothe et al. 1994]. Such a matrix can be inverted efficiently to find the pressure field solution using an Incomplete Cholesky Conjugate Gradient (ICCG) inversion algorithm [Kershaw 1978, Meijerink & van der Vorst 1977].

The ICCG inversion algorithm used in the BOUNCE code is taken from the work of Kuo-Petravic & Petravic [1981]. The size of the PPE matrix is specified at the time of BOUNCE code compilation, so that the minimum size of PPE coefficient matrix is inverted during each iterative implicit computational step.

### 4.5 Free Surface Forces

#### 4.5.1 Surface Tension

**Continuum Surface Force Model**

At the free surface of a fluid, an imbalance in molecular forces between adjacent sides of the interface can result in a surface force existing at the interface.
This surface force, known as surface tension, is directed normal to the inter-
face and is dependent on the curvature of the fluid interface for magnitude.

Surface tension is calculated in BOUNCE using the Continuum Surface
Force (CSF) model of Brackbill et al. [1992]. Under the CSF model, surface
tension surface forces are replaced by a volume force, which acts on the fluid
within a small transitional region surrounding the interface. The CSF model
has been shown to represent surface tension behaviour accurately for a wide
variety of problems [Brackbill et al. 1992].

Under the CSF model, the surface tension volume force is defined by
[Brackbill et al. 1992],
\[
\lim_{h \to 0} \int_{\Delta V} F_{sv}(x) \, d^3x = \int_{\Delta A} F_{sa}(x_s) \, dA. \tag{4.50}
\]
In this equation, \( F_{sa}(x_s) \) is the surface force resulting from surface tension
at location \( x_s \). This force is integrated over a small area of the free surface,
\( \Delta A \). \( F_{sv}(x) \) is the CSF volume force equivalent to \( F_{sa} \), and is integrated over
a small volume surrounding the surface, \( \Delta V \). The width of the transition
region surrounding the surface is \( h \), such that \( \Delta V = h \times \Delta A \).

The surface tension surface force, \( F_{sa} \), can be defined as
\[
F_{sa}(x_s) = \sigma \kappa(x_s) \hat{n}(x_s), \tag{4.51}
\]
where \( \sigma \) is the surface tension coefficient for the fluids, \( \kappa(x_s) \) is the curvature
of the free surface, defined positive if the centre of curvature is located on
the fluid side of the interface, and \( \hat{n}(x_s) \) is a unit normal to the interface,
directed towards the fluid side of the interface.

Using the method of Brackbill et al. [1992], equations (4.50) and (4.51)
can be combined to give the CSF volume force simply as,
\[
F_{sv}(x) = \sigma \kappa(x) \, n(x), \tag{4.52}
\]
where the surface normals are defined as the gradient of the VOF function,

\[ \mathbf{n}(\mathbf{x}) = \nabla F(\mathbf{x}) \]  

(4.53)

and the curvature is defined as the negative divergence of the unit normals,

\[ \kappa(\mathbf{x}) = - (\nabla \cdot \hat{\mathbf{n}}(\mathbf{x})). \]  

(4.54)

Model Implementation

Free Surface Normals  
Numerically, the free surface normals of equation (4.53) are calculated in BOUNCE using the RIPPLE method [Kothe et al. 1994]. Firstly, cell vertex centred normals are defined as gradients to the VOF function over a 2×2 kernel, thus

\[
\mathbf{n}_{i+1/2,j+1/2} = \begin{pmatrix}
    \frac{(F_{i+1,j+1} - F_{i,j+1}) \delta y_j + (F_{i+1,j} - F_{i,j}) \delta y_{j+1}}{\delta y_j + \delta y_{j+1} \delta x_{i+1/2}} \\
    \frac{(F_{i+1,j+1} - F_{i+1,j}) \delta x_i + (F_{i,j+1} - F_{i,j}) \delta y_{i+1}}{\delta x_i + \delta y_{i+1} \delta y_{j+1/2}}
\end{pmatrix} \mathbf{i} + \begin{pmatrix}
    (F_{i+1,j+1} - F_{i+1,j}) \delta y_j + (F_{i,j+1} - F_{i,j}) \delta y_{j+1}
\end{pmatrix} \mathbf{j}.
\]  

(4.55)

The cell centred normals are then calculated as averages of the four vertex normals surrounding each cell,

\[
\mathbf{n}_{i,j} = \frac{1}{4} \left( \mathbf{n}_{i+1/2,j+1/2} + \mathbf{n}_{i+1/2,j-1/2} + \mathbf{n}_{i-1/2,j-1/2} + \mathbf{n}_{i-1/2,j+1/2} \right).
\]  

(4.56)

Free Surface Curvature Calculation  
The numerical calculation of free surface curvatures can be problematic because the VOF function varies continuously between zero and one within the free surface transition region of a fluid interface, but does not vary beyond this region. Thus, defining unit normals to the interface at the extremities of the free surface transition region is difficult because the gradient of the VOF function is not continuous at these locations. Three features are included in BOUNCE to overcome this problem.
Volume of Fluid Function Smoothing  The first technique aimed at improving free surface curvature calculation is to smooth the VOF function prior to differentiating. This technique has been used in the RIPPLE code [Kothe et al. 1994]. It effectively allows the free surface curvature to be defined over a larger transition region than the free surface normals, as given by equation (4.55), leading to a more accurate surface force calculation at the extremities of the transition region.

In BOUNCE, a smoothed VOF function, $\tilde{F}(x)$, is defined as the convolution of the calculated VOF function and a two dimensional B-Spline,

$$\tilde{F}(x) = \int_{\text{all volume}} F(x') K(x' - x) \, dx'.$$

The two dimensional B-Spline used in BOUNCE is taken from the surface tension work of Morton, Rudman & Liow [1997], where

$$K(x' - x) = AK_x(x' - x) K_y(y' - y),$$

and

$$x = xi + yj$$

and

$$x' = x'i + y'j.$$  

The one-dimensional B-Splines in equation (4.58) are identical in both dimensions, and are given by

$$K_w(w) = \begin{cases} 
\frac{8}{3} \left[ 1 - 6 \left( \frac{|w|}{w_m} \right)^2 + 6 \left( \frac{|w|}{w_m} \right)^3 \right], & \text{if } \frac{|w|}{w_m} < \frac{1}{2} \\
\frac{16}{3} \left[ 1 - \left( \frac{|w|}{w_m} \right)^3 \right], & \text{if } \frac{1}{2} \leq \frac{|w|}{w_m} < 1 \\
0, & \text{otherwise}, 
\end{cases}$$

where $w$ can represent either $x' - x$ or $y' - y$. On a uniformly sized grid, the two-dimensional B-Spline of equation (4.58) is evaluated over an area
of $2s_k \delta x \times 2s_k \delta y$, where $s_k$ is the half width of the smoothing kernel in cell spacings, and is not necessarily an integer value. Thus, in equation (4.61), $w_m = s_k \delta x$ in the $x$ direction, and $w_m = s_k \delta y$ in the $y$ direction.

The constant $A$ in equation (4.58) is found by noting that if the VOF function were constant throughout the entire space, it would equal the smoothed VOF function. Thus, from (4.57) we have,

$$1 = \int_{\text{all volume}} K(x) \, dx,$$

or

$$A = \left\{ \int_{\text{all volume}} K_x(x)K_y(y) \, dx \, dy \right\}^{-1}. \quad (4.63)$$

Computationally, equation (4.57) is applied in BOUNCE using

$$\bar{F}_{i,j} = \sum_{n=i-n_k,i+n_k}^{x_{n+\frac{1}{2}}} K_x \left( x - x_{i+\frac{1}{2}} \right) dx \int_{y_{m-\frac{1}{2}}}^{y_{m+\frac{1}{2}}} K_y \left( y - y_{j+\frac{1}{2}} \right) dy,$$

where the products of the integrals on the right side of the equation are constants, and can be calculated and stored prior to a simulation commencing.

In practice the size of the kernel over which the VOF function is smoothed is determined by compromise. The advantage of using a large kernel is that surface curvatures are defined consistently throughout the whole fluid interface transition region. The disadvantage of using a kernel that is too large is that ripples of dimensions smaller than the kernel size may develop in the free surface, and these ripples may grow to become serious instabilities. A second disadvantage of a kernel that is too large is that features of the free surface whose dimensions are smaller than those of the kernel may not be realistically represented. An examination of the effect of kernel size on stationary cylinder pressures in given in Section 4.5.1. A kernel size within the range $1.0 \leq s_k \leq 1.5$ is used in droplet impact simulations.
Curvature Difference Equation  The second technique used to improve free surface curvature calculation is to expand equation (4.54) in terms of the vector normal and the magnitude of the vector normal prior to differencing [Brackbill et al. 1992, Kothe et al. 1994].

Equation (4.53) can be differentiated to give,

\[
\kappa = -(\nabla \cdot \hat{n}) = -\left( \nabla \cdot \frac{n}{|n|} \right) = \frac{1}{|n|} \left[ \left( \frac{n}{|n|} \cdot \nabla \right) |n| - (\nabla \cdot n) \right]. \tag{4.65}
\]

Expanded into components, this becomes

\[
\kappa = \frac{1}{|n|} \left[ \frac{n_x}{|n|} \frac{\partial |n|}{\partial x} + \frac{n_y}{|n|} \frac{\partial |n|}{\partial y} - \frac{1}{x^\xi} \frac{\partial x^\xi n_x}{\partial x} - \frac{\partial n_y}{\partial y} \right], \tag{4.66}
\]

where \( \xi \) is zero or one in Cartesian or cylindrical coordinates, respectively.

We can expand equation (4.66) still further to give

\[
\kappa = \frac{1}{|n|} \left[ \left( \frac{n_x}{|n|} \right)^2 \frac{\partial n_x}{\partial x} + \frac{n_x n_y}{|n|^2} \left( \frac{\partial n_x}{\partial x} + \frac{\partial n_y}{\partial y} \right) + \left( \frac{n_y}{|n|} \right)^2 \frac{\partial n_y}{\partial y} - \frac{1}{x^\xi} \frac{\partial x^\xi n_x}{\partial x} - \frac{\partial n_y}{\partial y} \right]. \tag{4.67}
\]

When differenced, equation (4.67) tends to place greater emphasis on regions with larger, and thus more reliable, free surface normals than the straight difference form of equation (4.53). It is in this form that equation (4.54) is differenced in the RIPPLE code, and used for evaluating free surface curvature [Kothe et al. 1994].

In BOUNCE, equation (4.67) is split into Cartesian and cylindrical components before differencing. Thus, free surface curvature is calculated using

\[
\kappa_{i,j} = \kappa_{(\text{Cartesian})_{i,j}} + \kappa_{(\text{cylindrical})_{i,j}}, \tag{4.68}
\]

63
where

$$\kappa_{(Cartesian) i,j} = \frac{1}{|n|_{i,j}} \left[ \left( \frac{n_x}{|n|} \right)_{i,j}^2 \frac{\partial n_x}{\partial x_{i,j}} + \left( \frac{n_x n_y}{|n|^2} \right)_{i,j} \left( \frac{\partial n_x}{\partial x} + \frac{\partial n_y}{\partial y} \right)_{i,j} \right. \right.$$

$$\left. + \left( \frac{n_y}{|n|} \right)_{i,j}^2 \frac{\partial n_y}{\partial y_{i,j}} - \frac{\partial n_x}{\partial x_{i,j}} - \frac{\partial n_y}{\partial y_{i,j}} \right]$$

(4.69)

and

$$\kappa_{(cylindrical) i,j} = - \left( \frac{n_x}{|n|} \right)_{i,j} \frac{1}{x_i}.$$  \hspace{2cm} (4.70)

The reason for splitting equation \([4.67]\) in BOUNCE is primarily to accommodate the curvature correction technique, as detailed in the next section. However, experience with using the difference form of the split equation \([4.68]\) suggests that it may be slightly more accurate in calculating curvatures in cylindrical coordinates than the finite difference form of equation \([4.67]\).

In evaluating equations \([4.69]\) and \([4.70]\), cell vertex normals are first calculated as per equation \([4.55]\), but in this case the normals are defined using the smoothed VOF function, \(\tilde{F}_{i,j}\). The cell centred normal variables \(n_x\), \(n_y\) and \(|n|\) are calculated by averaging of the four vertex normals surrounding each cell. Thus

$$n_{i,j} = \frac{1}{4} \left( n_{i+1/2,j+1/2} + n_{i+1/2,j-1/2} + n_{i-1/2,j-1/2} + n_{i-1/2,j+1/2} \right),$$  \hspace{2cm} (4.71)

where

$$n_{i,j} = n_x i + n_y j$$  \hspace{2cm} (4.72)

and

$$|n|_{i,j} = \sqrt{n_x^2 + n_y^2}.$$  \hspace{2cm} (4.73)

Normal derivatives are calculated as differences between vertex centred normals. Thus,

$$\frac{\partial n_x}{\partial x_{i,j}} = \frac{1}{2\delta x_i} \left[ (n_x)_{i+1/2,j+1/2} + (n_x)_{i+1/2,j-1/2} - (n_x)_{i-1/2,j+1/2} - (n_x)_{i-1/2,j-1/2} \right],$$  \hspace{2cm} (4.74)

64
\[ \frac{\partial n_y}{\partial x_{i,j}} = \frac{1}{2\delta x_i} \left[ (n_y)_{i+1/2,j+1/2} + (n_y)_{i+1/2,j-1/2} - (n_y)_{i-1/2,j+1/2} - (n_y)_{i-1/2,j-1/2} \right], \]  
(4.75)

\[ \frac{\partial n_x}{\partial y_{i,j}} = \frac{1}{2\delta y_j} \left[ (n_x)_{i+1/2,j+1/2} + (n_x)_{i-1/2,j+1/2} - (n_x)_{i+1/2,j-1/2} - (n_x)_{i-1/2,j-1/2} \right], \]  
(4.76)

and

\[ \frac{\partial n_y}{\partial y_{i,j}} = \frac{1}{2\delta y_j} \left[ (n_y)_{i+1/2,j+1/2} + (n_y)_{i-1/2,j+1/2} - (n_y)_{i+1/2,j-1/2} - (n_y)_{i-1/2,j-1/2} \right], \]  
(4.77)

**Transition Region Curvature Correction**  
A correction to the curvature calculation has been developed in this study. To illustrate the purpose of the correction, consider Figure 4.3, where free surface curvatures at the circumference of a cylinder have been calculated by BOUNCE. In this example the width of the fluid interface transition region has been exaggerated by using a VOF smoothing kernel size of \( s_k = 4.5 \).

As shown in Figure 4.3, free surface curvatures calculated by BOUNCE are not constant across the interface transition region of the cylinder, but rather are higher than average towards the centre of the cylinder, and lower than average away from the centre of the cylinder. As surface tension volume forces are applied with a weighting towards the fluid side of the transition region, as detailed in the next section, curvature variations across the transition region can cause the magnitude of surface tension force to be under or over estimated.

The purpose of the BOUNCE correction is to alter free surface curvatures calculated in areas located at the extremes of the fluid interface transition.
Curvature

<table>
<thead>
<tr>
<th>Curvature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.20</td>
</tr>
<tr>
<td>1.13</td>
</tr>
<tr>
<td>1.06</td>
</tr>
<tr>
<td>0.99</td>
</tr>
<tr>
<td>0.91</td>
</tr>
<tr>
<td>0.84</td>
</tr>
<tr>
<td>0.77</td>
</tr>
<tr>
<td>0.70</td>
</tr>
</tbody>
</table>

**Figure 4.3:** Curvature contour plot surrounding a cylinder of unit radius as calculated by BOUNCE. No curvature correction is employed in this calculation. The thin black line represents the actual surface of cylinder.

region to more accurately resemble curvatures existing at the centre of the transition region.

Figure 4.4 shows a Cartesian computational cell, centred at $x_c$, and located within a fluid interface transition region. The transition region surrounds an actual fluid free surface, which has a constant curvature of $\kappa^*$. The cell has a VOF value $F(x_c)$, which is higher than $\frac{1}{2}$, so consequently the cell is located towards the fluid side of the fluid interface transition region. The uncorrected curvature calculated at $x_c$ is $\kappa$, which defines the centre of
curvature for the free surface as

\[ x_o = \frac{1}{\kappa} \hat{n}(x_c). \]  

(4.78)

Here \( \hat{n}(x_c) \) is the unit normal associated with the free surface.

To calculate the actual free surface curvature for the transition region, \( \kappa^* \), from a knowledge of the curvature at the cell centre, \( \kappa \), we require the distance along the radius of curvature between the actual free surface location and the cell centre location. This distance is represented by \( c \) in Figure 4.4.
Assuming that the actual free surface location is approximately at the centre of the fluid interface transition region, and assuming that the VOF function varies linearly across the width of the transition region, we have

\[ c = \left( F - \frac{1}{2} \right) / \frac{\partial F}{\partial s}, \]  

(4.79)

where \( s \) is a coordinate directed from the cell centre, \( x_c \), along the free surface normal, \( \hat{n}(x_c) \).

As the free surface normal at \( x_c \) is calculated as the gradient of the VOF function, the gradient of the VOF function in the direction perpendicular to \( \hat{n}(x_c) \) is zero. Thus, equation (4.53) can be redefined in terms of the coordinate \( s \) as

\[ n(x_c) = |n(x_c)| \hat{n}(x_c) = \frac{\partial F}{\partial s}(x_c) \hat{n}(x_c), \]  

(4.80)

or

\[ \frac{\partial F}{\partial s}(x_c) = |n(x_c)|. \]  

(4.81)

Combining equations (4.79) and (4.81) we have

\[ c = \frac{\left( F(x_c) - \frac{1}{2} \right)}{|n(x_c)|}, \]  

(4.82)

and the corrected curvature for the cell can be calculated as

\[ \kappa^* = \frac{1}{\left( \frac{1}{\kappa} + c \right)}. \]  

(4.83)

While this correction has been developed using an example in Cartesian coordinates, the derivation also holds in cylindrical coordinates. Thus, in BOUNCE, equation (4.83) is applied separately to both the Cartesian and cylindrical curvatures, as calculated using equations (4.69) and (4.70), respectively.
In practice, equation (4.83) must be modified to guard against inaccurate normal evaluations causing spurious curvature corrections. Consequently, the numerical implementation of equation (4.83) is

\[ \kappa^* = \max \left\{ \min \left[ \frac{1}{\kappa + C_{correct}}, 0.5\kappa \right], 1 \right\} \]. \quad (4.84) \]

Here \( C_{correct} \) is a user-defined coefficient, necessary because typically VOF functions do not vary linearly across the entire width of fluid transition regions. A value of 0.5 for this coefficient gives good results for both Cartesian and cylindrical curvatures.

Figure 4.5 shows the example cylinder of Figure 4.3 recalculated with the curvature correction implemented. As shown, curvatures are now more uniform throughout the fluid interface transition region surrounding the actual fluid surface. A quantitative assessment of the effect of the curvature correction on the pressure generated within a static cylinder is given later in the chapter.

**Surface Volume Force Calculation** Once the free surface normals and curvatures have been calculated, the surface tension volume force is calculated using the computational form of equation (4.52), namely,

\[ F_{(sv)i,j} = \sigma \kappa_n. \quad (4.85) \]

In BOUNCE, surface tension volume forces are calculated explicitly. While an iterative implicit implementation of the surface tension force calculation appears straightforward, in practice difficulties occur in the form of fluid wisp generation. Under an implicit surface tension scheme, forces are calculated using end of timestep fluid interface positions. If a cell were to contain fluid at
Figure 4.5: A curvature contour plot surrounding a cylinder of unit radius, as calculated by BOUNCE. The curvature correction is employed with $C_{\text{correct}} = 0.5$. The thin black line represents the actual surface of the cylinder.

At the commencement of a timestep, but not at the end, then no surface tension volume force would be calculated for the cell. Under certain conditions, for example at a moving free surface boundary, this absence of a surface tension force can allow small amounts of fluid to remain stationary, thus separating from the bulk of the fluid. In this fashion, wisps of fluid can trail behind a moving boundary under an implicit surface tension formulation.

As the surface tension calculation is explicit, the simulation timestep used in BOUNCE is subject to a capillary wave stability criterion. Thus, as given
by Kothe et al. [1994], the computational timestep must satisfy,

\[ \delta t \leq \sqrt{\frac{\rho \min(\delta x_i \times \delta y_j)}{4\pi\sigma}}. \]  

(4.86)

**Surface Tension Test — Pressure within a Stationary Cylinder**

A series of tests was undertaken to quantify the effect of the curvature smoothing kernel size and curvature correction technique on surface tension force application. The accuracy of surface tension force application was gauged by comparing the average pressure generated within a cylinder of fluid to the theoretically correct pressure generated within an equivalently sized cylinder.

The results of the tests are shown in Table 4.1. In this table $P_{av}$ is the non-dimensional average pressure within the cylinder. It is defined as,

\[ P_{av} = \left[ \text{average}_{r<\frac{R}{2}} P(r) \right] \times \frac{R}{\sigma}. \]  

(4.87)

where $R$ is the radius of the cylinder, $\sigma$ is the surface tension coefficient of the fluid interface, $r$ is the radial distance between the centre of the cylinder and the position of $P$. The average in equation (4.87) is performed within a concentric cylinder to the fluid cylinder, having half the radius of the fluid cylinder. The reason for this is to exclude the transitional fluid interface zone from the pressure average. Also, $n_{cells}$ is the number of computational cells used in generating $P_{av}$. This is a constant for each cylinder radius.

As shown in Table 4.1 tests were performed on cylinders having radii of from three to twenty uniformly sized computational cells, and curvature smoothing was varied between no smoothing and a large smoothing kernel ($s_k = 3.5$). Also, each test was performed with and without the curvature correction algorithm implemented. Ideally, $P_{av}$ should be unity in these tests.
Table 4.1: Pressures $P_{av}$ calculated within a stationary cylinder of radius $R$.

Values in brackets indicate those calculated without the curvature correction algorithm, those without brackets are calculated with the curvature correction algorithm on.

Examining Table 4.1 we note that static pressures are calculated reasonably accurately, with all but the most extreme examples falling within a few percent of the theoretically correct value. Smoothing of the VOF function prior to curvature evaluation is beneficial for all but the smallest radius cylinder tested. However, it should be noted that the average pressure for the three cell radius case was averaged from just four internal computational cells, so the accuracy of this small radius measurement is the least of all the results presented. A smoothing kernel size of around $s_k = 1.5$ appears to be optimal.

A comparison of pressures calculated with and without the curvature correction, indicated by plain text and bracketed text, respectively, shows that with almost no exception, the curvature correction improves the calculated cylinder pressure. Improvements are most noticeable where fluid interface transition regions would have a large width relative to the cylinder size — specifically, when a large smoothing kernel is combined with a small cylinder radius.
4.5.2 The Viscous Vapour Layer Surface Force

In droplet impact simulations, a thin layer of viscous vapour flow exists between the underside of the droplet and the solid surface. This vapour layer, generated by the vaporisation of liquid at the underside of droplet, is responsible for repelling the droplet from the solid surface. Consequently, the pressure at the lower surface of an impacting droplet may be different to the ambient pressure experienced by the remainder of the surface of the droplet, and is expressed by,

\[ P_{\text{droplet underside}} = P_{\text{ambient}} + P_{vl} = P_{vl}. \] (4.88)

Here pressures are expressed as gauge pressures, and consequently \( P_{\text{ambient}} = 0 \). The vapour layer pressure, \( P_{vl} \), is calculated as a function of radial displacement by the viscous vapour layer code, as detailed in Chapters 8 and 9.

As a feature developed in BOUNCE, the vapour layer surface pressure is applied to the VOF code using a surface volume force, in an analogous fashion to the surface tension volume force. Considering that the vapour layer surface force is given by

\[ F_{vl} = P_{vl} \hat{n}, \] (4.89)

we have by a direct analogy with equations (4.51) and (4.85) the surface volume vapour layer force,

\[ F_{(vl)i,j} = P_{(vl)i} n_{i,j}. \] (4.90)

The free surface normals used in equation (4.90) are calculated in the same manner as the normals used for the surface tension volume force, however for the surface volume vapour layer force, the VOF values used to calculate these
Loop through cells outward in the radial direction

Loop through cells upward in the vertical direction

Implicitly calculate free surface vapour volume force normals, $n_x$ and $n_y$.

Is $n_y < 0$ and $F_{ij,y} \geq \varepsilon_y$?

Yes. Move to next column.

No.

Is $n_x < 0$?

Yes. Move to next cell.

No.

$$s = \min \left( 1.0, \frac{10|n_y|}{\sqrt{n_x^2 + n_y^2}} \right)$$

$$F_{ij,x} = F_{(v_{l,x})i,j} n_x s$$

$$F_{ij,y} = F_{(v_{l,y})i,j} n_y$$

Figure 4.6: A schematic showing the algorithm used to apply the viscous vapour layer volume force. Note that $F_{(v_{l})i,j} = F_{(v_{l,x})i,j} i + F_{(v_{l,y})i,j} j$ and $n_{i,j} = n_x i + n_y j$.

normals are end of timestep values. Thus, the normals used in applying the vapour layer surface force are implicitly calculated, in line with the implicit calculation of viscous vapour layer height.

The algorithm used to apply this volume force is shown schematically in Figure 4.6. As shown, BOUNCE loops through each column of cells, from the lowest cell first, applying the volume force. If a cell is encountered which contains some fluid, but has a surface normal directed downwards, the loop terminates and the next column of cells is considered. Alternatively, if a cell is encountered which contains no fluid, but has a downward facing normal,
then no surface force is applied to that cell, and the next cell of the same column is considered. The purpose of these checks is to ensure that only the downward facing underside of the droplet has a viscous vapour layer force applied to it.

As also shown in Figure 4.6, a scale factor is introduced when applying the $x$ component of the vapour layer volume force. This scale factor, represented in the figure as $s$, is introduced so that the iterative implicit convergence criterion of Section 4.3.3 is not violated as $n_y$ varies between small positive and negative values. Tests run with and without the scale factor included show that its effect on droplet dynamics is negligible, but its effect on iterative implicit convergence can be significant.

An integrity test of the vapour layer pressure application technique is presented in Appendix D.

4.5.3 Implementation of the Surface Volume Forces in the PPE Equation

Two factors influence the application of the calculated free surface volume forces into the Pressure Poisson Equation. Firstly, the PPE equation requires volume forces centred at cell boundaries. Thus, the cell centred volume forces $F_{(sv)_{i,j}}$ and $F_{(vl)_{i,j}}$ require averaging to generate cell boundary centred forces. Secondly, the BOUNCE VOF code is a single fluid algorithm and as such, fluid flow within the void side of the fluid interface transition region is not calculated. For this reason, these transition region body forces must be weighted towards the fluid side of the transition region [Brackbill et al. 1992].

In BOUNCE, the cell border averaging and fluid side weighting are ac-
complished using

\[ F_{(b)i,j} = F_{(b,x)i,j} \mathbf{i} + F_{(b,y)i,j} \mathbf{j} \]
\[ = (F_{(sv,x)i,j} + F_{(vl,x)i,j}) \mathbf{i} + (F_{(sv,y)i,j} + F_{(vl,y)i,j}) \mathbf{j} \tag{4.91} \]

and

\[ F_{(b)i+\frac{1}{2},j+\frac{1}{2}} = 2F_{n+1}^{i+\frac{1}{2},j+\frac{1}{2}} \left[ \frac{F_{(b,x)i,j} \delta x_{i+1} + F_{(b,x)i,j} \delta x_{i+1}}{\delta x_i + \delta x_{i+1}} \right] \mathbf{i} \]
\[ + 2F_{n+1}^{i+\frac{1}{2},j+\frac{1}{2}} \left[ \frac{F_{(b,y)i,j} \delta y_{j+1} + F_{(b,y)i,j} \delta y_{j+1}}{\delta y_j + \delta y_{j+1}} \right] \mathbf{j} \tag{4.92} \]

for all \( i, j \) within the fluid region.

The next chapter details the first of the new VOF advection algorithms developed in this study.
Chapter 5

The Defined Donating Region
VOF Advection Algorithm

5.1 VOF Advection

As discussed in the previous chapter, fluid location is recorded in BOUNCE using a Volume of Fluid (VOF) function, $F$. The Volume of Fluid function is unity in fluid cells, zero in void cells, and varies between zero and one in cells containing a free surface.

Excellent reviews of past and present VOF advection methods have been given by Rider & Kothe [1998] and Rudman [1997], so only a brief overview of some of the methods available will be given here.

As a Lagrangian invariant of the fluid, the VOF function, $F$, satisfies [Kothe et al. 1994]

$$\frac{\partial F}{\partial t} + (\mathbf{V} \cdot \nabla) F = 0. \quad (5.1)$$

Equation (5.1) describes the transport of a scalar quantity with the fluid, where the quantity varies continuously from one spatial point to the next. In reality however, the VOF function does not vary continuously from one point...
to the next, but rather experiences a discrete change over the infinitesimal dimension of each free surface interface. Consequently, special methods must be used to difference equation (5.1) so that diffusion of free surface interfaces does not result [Rudman 1997].

One such method uses the Flux-Corrected Transport (FCT) Algorithm developed by Zalesak [1979]. The FCT algorithm was developed as a general method for advecting any scalar quantity, but was applied to the process of VOF advection by Rudman [1997]. Under the FCT method, equation (5.1) is differenced using a combination of diffusive upwind and dispersive downwind first order difference schemes. The dependence on each scheme is chosen so that the advected solution contains no extrema that were not present in the previous timestep solution. Rudman [1997] demonstrated that the FCT-VOF advection method is not as accurate as modern piecewise linear advection methods.

The majority of VOF advection methods are not derived from a direct difference formulation of equation (5.1), but are instead developed using a two stage process. Firstly, free surface interfaces are ‘reconstructed’ from the VOF data, so that a geometrical profile is found which approximates the actual free surface location. Changes in VOF values are then calculated by integrating fluid fluxes over cell boundaries, using the geometrical profile to indicate the location of fluid regions. The different advection algorithms based on this two stage process can be loosely classified according to the technique used to reconstruct the free surfaces in each cell, and by the method used to perform the boundary flux integrations [Rider & Kothe 1998].

VOF advection methods that represent free surface interfaces as lines directed parallel to one of the grid coordinates are known as piecewise constant schemes. The SLIC (Simple Line Interface Calculation) of Noh & Woodward
and the SURFER code of Lafaurie, Nardone, Scardovelli, Zaleski & Zanetti [1994] are examples of a piecewise constant scheme.

A variation on the piecewise constant technique is the method used in the SOLA-VOF code of Nichols et al. [1980]. Under the Hirt-Nichols scheme, free surface interfaces are orientated in directions parallel to grid coordinates, but are also allowed the greater freedom of a stair-shaped profile if local VOF distribution conditions permit. Similar algorithms include those developed by Chorin [1980] and Barr & Ashurst [1984].

The alternative to representing free surface interfaces as lines parallel to one of the grid coordinates is to orientate free surface interfaces in a direction perpendicular to the locally evaluated VOF gradient. Free surface interfaces within each cell can then acquire any orientation, and the geometrical profile of the fluid can more closely represent the actual fluid geometry. Such schemes are known as piecewise linear schemes, and include those developed by Debar [1974], Youngs [1984], Ashgriz & Poo [1991], Puckett, Almgren, Bell, Marcus & Rider [1997] and Rider & Kothe [1998]. These schemes tend to be more complex than their piecewise constant cousins, but have been shown to be significantly more accurate [Pilliod Jr 1992, Rudman 1997, Rider & Kothe 1998].

The method of integration used to determine cell boundary fluxes is also used to classify VOF advection techniques. Under operator or dimensionally split schemes, boundary fluxes are calculated independently in each coordinate direction, often with some type of limiter employed to reduce possible undershoots or overshoots occurring in cell VOF values. Free surfaces are usually reconstructed between integrations in each of the coordinate directions under this technique. The Youngs algorithms are examples of operator split schemes [Youngs 1982, Youngs 1984].
Multi-dimensional schemes can be more accurate and efficient in calculating cell boundary fluxes than operator split schemes [Rider & Kothe 1998]. Under a multi-dimensional scheme, cell boundary fluxes are calculated with a dependence between fluxes calculated in each of the coordinate directions. Example multi-dimensional schemes include those developed by Rider & Kothe [1998] and Puckett et al. [1997].

Two separate VOF advection models have been developed in this study. To avoid confusion, the first new method developed will be referred to as the Defined Donating Region (DDR) scheme, while the second new method will be referred to as the Stream scheme, for reasons that will become obvious. Both methods can be classified as piecewise linear schemes with cell boundary fluxes integrated using a fully multidimensional technique. However, both methods are quite different, and the multidimensional integration techniques used in both are unique. The performance of both methods will be compared with the Hirt-Nichols (H-N) advection algorithm as employed in the SOLA-VOF [Nichols et al. 1980] and RIPPLE [Kothe et al. 1994] codes.

In this chapter we provide a description of the Hirt-Nichols Algorithm and analyse the performance of this algorithm with reference to a simple translation test. The new Defined Donating Region algorithm is then presented, and the performance of this algorithm is also analysed with reference to the same simple translation test. The Stream advection algorithm is presented in Chapter 6 and a detailed comparison of the performance of all three VOF advection algorithms is given in Chapter 7.
5.2 The Hirt-Nichols VOF advection Algorithm

5.2.1 Translation Test - The Box Problem

The simplest advection algorithm test involves translating a geometric shape around the computational domain. Assuming the velocity field contains constant vorticity, the geometric shape should remain intact and the total volume of fluid within the region should be conserved. The following example, that of a block of fluid moving diagonally across a computational region, was chosen to highlight some of the problems existing with the H-N algorithm.

The two dimensional cartesian region shown in Figure 5.1 has dimensions of $1 \times 1 \text{ m}^2$, and is composed of 10,000 equally sized cells, each $0.01 \times 0.01 \text{ m}^2$. A square block of fluid, of dimensions $0.1 \times 0.1 \text{ m}^2$, moves with equal horizontal and vertical velocities of 1 m/s towards the top right-hand corner of the computational region. Figure 5.1 shows the fluid position computed using the H-N algorithm every 0.1 s until 0.7 s. The computational timestep used in this calculation was 1 ms and the exact velocity field was imposed on the VOF advection algorithm rather than being calculated from the discretised Navier-Stokes equations.

The VOF contours in Figure 5.1 show significant diffusion of fluid in a direction normal to the velocity of the fluid. While this diffusion tends to decrease with decreasing timestep, the timestep used in the example, 1 ms, is significantly smaller than the maximum time step for stability of 5 ms, and is of the same order of magnitude as time steps required by the pressure solution algorithm.

More concerning however, is the total area of fluid within the computational region, as shown in Figure 5.2. The fluid volume, computed by
summing the VOF function over all cells at each timestep, increases to approximately 118% of the initial amount by 0.6 s, before decreasing to 114% of the initial amount by 0.7 s. Thus, the total volume of fluid is not being conserved, and VOF equation (5.1) is not being satisfied. Similar results were found using cylindrical coordinates. The H-N algorithm will now be examined with reference to these two deficiencies.

5.2.2 The Advection Model

The Hirt-Nichols algorithm reconstructs the free surface in donating and accepting cells as lines parallel to either of the cell boundaries. To aid descrip-
tion of the method, the case is examined of fluid fluxing through the right-
hand boundary of a cell during a timestep of duration $\delta t$ [Nichols et al. 1980],
as illustrated in Figure 5.3.

Equations (5.2) and (5.3) are used to calculate the amount of VOF flux
over the boundary,

$$\delta F = \min \{ F_{AD} |u\delta t| + CF, F_D \delta x_D \}, \quad (5.2)$$

where

$$CF = \max \{(1 - F_{AD}) |u\delta t| - (1 - F_D) \delta x_D, 0\}. \quad (5.3)$$

Subscript $A$ describes the acceptor cell and $D$ describes the donor cell,
where fluid velocity over the boundary is directed from the donor cell towards
the acceptor cell. Subscript $AD$ describes the cell which best represents the magnitude of the VOF function at the boundary. Determination of $AD$ is made using the mean surface orientation and VOF magnitudes in both the acceptor and donor cells [Nichols et al. 1980]. The velocity over the boundary is $u$.

The minimum feature of equation (5.2) ensures that no more fluid is fluxed from the donating cell than it contains. The maximum feature of equation (5.3) ensures that no more void is fluxed from the donating cell than it contains. Once the flux is found, the VOF functions are incremented in Cartesian coordinates using

$$
F_A^* = F_A + \delta F/\delta x_A
$$

(5.4)
and

\[ F_D^* = F_D - \delta F / \delta x_D. \quad (5.5) \]

Different regimes of advection are shown in Figure 5.3. Case (A) shows the relationship between fluid velocity, donor and acceptor cells. Case (B) has the donor cell free surface normal to the cell boundary, resulting in \( AD = D \) and

\[ \delta F = F_D |u\delta t|. \quad (5.6) \]

The minimum feature of equation (5.2) is used in case (C) where the entire contents of the donor cell is removed during the timestep. Case (D) illustrates the maximum feature of equation (5.3) where the amount of void fluxed from the donor cell is limited by the amount of void it contains. In both these cases the VOF function at the boundary is represented by the acceptor cell, thus \( AD = A \).

### 5.2.3 Stability Criterion

As an advection scheme, the H-N algorithm is subject to the Courant condition. Under the H-N algorithm this criterion is given by,

\[ \delta t_{\text{max}} = \min \left\{ \frac{C \delta x}{|u|}, \frac{C \delta y}{|v|} \right\} \text{ over all boundaries}. \quad (5.7) \]

In an explicit analysis, the constant \( C \) should be less than 1.0 for stability [Kothe et al. 1994], and is defaulted to 0.5 and 0.3 in the SOLA-VOF and RIPPLE codes, respectively.

### 5.2.4 The Computational Procedure

Figure 5.4 shows schematically how the H-N algorithm is implemented by the SOLA subroutine \textit{VFCONV}. When the subroutine is called, the new timestep
VOF array $f(i,j)$ is equal to the previous timestep VOF array $fn(i,j)$. A loop is initiated which sweeps through all boundaries within the computational domain. First the fluid velocities over both the upper and right boundaries are checked to ensure stability of the advection algorithm. The fluid fluxes over each boundary are then computed, based on the previous timestep VOF function $fn(i,j)$ and velocity field. Each cycle of the loop computes first the flux over the right $x$ direction boundary, followed by the flux over the upper $y$ direction boundary. These fluxes are added to the $f(i,j)$ array, such that at the completion of the loop, the $f(i,j)$ array represents the new fluid position.

A second sweep of the domain is then made, checking that each VOF value indicates either an empty cell, a full cell or a partially full ‘surface’ cell. If the $f(i,j)$ value is below $emf\ (\varepsilon_F)$, the lower surface cell VOF limit, it is set to zero and any change in fluid volume is incremented to the monitoring variable $vchgt$. Conversely, if the $f(i,j)$ value is above $1-emf=emf_1$, the upper surface cell VOF limit, $f(i,j)$ is set to one and any change in fluid volume incremented to $vchgt$. Thus, the computational variable $vchgt$ records the total volume change caused by forcing the VOF function to these limits in fluid and void regions, as discussed in Chapter 4.

### 5.2.5 VOF Conservation

As described above, under the H-N algorithm a check is made to ensure that the amount of fluid crossing a boundary is less than the amount of fluid contained in the donating cell. Similarly a check is made that the amount of void crossing a boundary is less than the amount of void contained in the donating cell. Unfortunately, these checks are made separately at each boundary. When the fluid flux is determined over two or more boundaries and is calculated using only previous timestep VOF values, as in the $VFCONV$
SUBROUTINE VFCONV

Loop through all cells within computational region

Calculate product of velocity and timestep over upper and right cell boundaries

Check stability criterion over both upper and right cell boundaries

Satisfied

Calculate VOF flux over right boundary and increment VOF function in acceptor and donor cells

Calculate VOF flux over upper boundary and increment VOF function in acceptor and donor cells

Loop through all cells within computational region

Reset new VOF value of cell to zero if less than emf

Reset new VOF value of cell to one if greater than emf

Return to main program

Not Satisfied

Return to main program and repeat with half timestep

Figure 5.4: A schematic of the SOLA-VOF subroutine VFCONV.
Figure 5.5: The advection of a VOF corner under the H-N algorithm. Case (A) shows the actual fluid location or VOF function before the timestep, and indicates the velocities over each boundary. Case (B) shows the actual fluid location after the timestep, and case (C) the VOF function after advection by the H-N algorithm. The dark shaded region indicates an unphysical negative VOF.

subroutine, more fluid or void can be fluxed from a donating cell than it contains.

To illustrate, consider the movement of the lower left corner of the test case block, schematically depicted in Figure 5.5. In this example the velocity out of the examined lower left cell is such that at the end of the timestep no fluid remains in this cell, and the fluid would ideally have the position depicted by diagram (B) at the end of the time step.

Under the H-N algorithm, fluid flux is first calculated over the right $x$ direction boundary of the lower left-hand cell. As this flux is calculated independently of the other dimension, the amount of flux over the boundary equals the amount of fluid contained in the cell. The flux over the top or $y$ direction boundary is then calculated. As this flux is based on the original fluid position, the flux over the top boundary also equals the contents of the lower left-hand cell. Once both fluxes are applied to the VOF function, the new VOF function in the lower left-hand cell becomes negative, equal in this
case to the magnitude of VOF originally contained in the cell. This situation is illustrated in diagram (C) of Figure 5.5. Once the VOF function is reset to zero, a net increase in fluid volume has occurred.

In a similar fashion a cell having fluid entering from two adjacent boundaries may obtain a VOF function greater than one, this also being an unphysical result causing a net decrease in fluid volume.

The example presented here obviously represents a worse case scenario, where the fluid velocity has equal magnitudes in both dimensions and the timestep is large. In many instances where the fluid velocity is independent of boundary orientations, gains in fluid volume can cancel losses. However the example does demonstrate that fluid volume is not rigorously conserved, and as illustrated by the box test case of Figure 5.2 the effect under certain flows can be severe.

5.2.6 VOF Diffusion

Diffusion of the VOF function is partly a result of the free surface reconstruction technique, where fluid is represented by rectangles having faces parallel to cell boundaries, and partly a result of the staggered layout of the velocity components. Both diffusion types will be illustrated using corners of the previously described test case box.

Consider first the diffusion resulting from the rectangular surface reconstruction technique. Here the lower right corner of the test box is used to demonstrate.

Five timesteps of fluid movement are shown in Figure 5.6 and the equal components of velocity are such that fluid moves one half of a square cell dimension per timestep. As illustrated, after the first timestep, the lower right fluid cell contains twice as much fluid as the actual case. The H-N
algorithm now reconstructs the fluid interface in this cell parallel to one of
the cell boundaries, where here a horizontal free surface was chosen. Over
the second timestep, fluid moves from this lower right fluid cell both upwards
and right, leaving two fluid cells with unrealistic VOF values. This diffusion
continues over the next three time steps, so that by the last, the corner region
contains considerably more fluid than the actual case, and two cells of low
VOF density have been generated some distance from the actual corner.

One can see that by forcing the fluid surfaces to be parallel to cell bound-
daries, fluid can accumulate at free surfaces moving normal to the fluid ve-
locity. Further, wisps of fluid are generated which propagate outwards from
free surface corners, also in a direction normal to fluid flow.

A method of reducing VOF wisp creation is to impose a minimum limit
on the fluid VOF density in a donating region before a new fluid cell can
be created [Torrey, Cloutman, Mjolsness & Hirt 1985]. This method is used
in both NASA-VOF2D and RIPPLE, where the VOF value in both the cell
upstream of the donating cell and the donating cell must be greater than
0.1 before donation to an empty cell is allowed. Under this method equation

\[ \delta F = \begin{cases} 
0 & F_D < 0.1 \text{ and } F_{DM} < 0.1 \\
\min \{F_AD |u\delta t| + CF, F_D\delta x_D\} & \text{otherwise}
\end{cases} \]  

(5.8)

where \( F_{DM} \) represents the cell upstream of the donating cell.

Experimenting with this minimum limit using the H-N algorithm has
shown that such a high level of 0.1 is required to reduce fluid wisp formation,
but at this level the wisp reduction algorithm can significantly alter calculated
fluid geometries. Also, wisps of fluid can be created when cells are not
entirely emptied, as well as when cells are unrealistically filled, and this
limiter method is only effective in reducing wisp formation from unrealistic
Figure 5.6: The lower right corner of the box test case moving with velocity upwards and to the right. Shown is the actual fluid position and the fluid position predicted by H-N algorithm. Cases (A)–(E) represent fluid positions at successive timesteps.
Another method of reducing fluid diffusion is to repack the VOF function after each timestep from one boundary of the computational domain across the entire region [Kothe et al. 1994]. This could be used for slowly moving liquids contained in a vessel under a gravitational field. However, such methods are limited to situations where the fluid is always in contact with at least one computational region boundary, not to the studied case of falling droplets.

The second cause of diffusion in convecting the VOF function results from the velocity components being separated between staggered locations. Here we consider the upper right corner of the test box as depicted in Figure 5.7. As in the last example, the equal fluid velocity components are such that fluid moves one half of a square cell dimension over the illustrated timestep.

Diagram (A) of Figure 5.7 shows the fluid location before the timestep,
diagram (B) shows the actual fluid location after the timestep while diagram (C) shows the computed fluid location after the time step. As fluid is fluxed over only the right and upper boundary of the fluid cell during the timestep, the upper right cell of the illustrated region is unrealistically empty at the end of the timestep. Thus, diffusion of fluid away from this vertex is occurring because the VOF function is being advected separately in a normal direction over each boundary.

These two causes of diffusion outlined are related, and to some extent interdependent. Both types are reduced with decreasing timestep by limiting the amount of fluid movement between free surface positioning. This is at the expense of computational efficiency. Both types are also reduced by limiting cell sizes, also at the expense of computational efficiency. Despite these similarities however, the two diffusion types are different and require different methods to reduce their negative effects.

5.3 The Defined Donating Region Advection Algorithm

The Defined Donating Region VOF advection algorithm, developed in this work, has two new features which provide better performance over the H-N algorithm. By defining donating regions within each computational cell, conservation of fluid volume is assured. Also, by using a more accurate free surface reconstruction model, VOF diffusion is decreased. These features are detailed separately.
5.3.1 Defined Donating Regions

The H-N algorithm, by using the previous timestep VOF values to define free surfaces and calculate VOF boundary fluxes, is essentially an explicit advection method. This explicitness is the cause of fluid volume discrepancies. An alternative is an implicit method, where fluxes are calculated from new timestep fluid geometries. Such a method would require iteration throughout the domain. To ensure VOF conservation over a large number of timesteps, the accuracy required from the VOF iteration procedure would be high, resulting in a large number of iterations to achieve convergence. Thus, an iterative implicit VOF advection method would be highly computationally expensive.

A more efficient scheme, as used in the Defined Donating Region method, is to define a donating region associated with each boundary from which fluid can cross the boundary, or be donated. The actual fluid flux over the boundary is then the intersection of the fluid region contained in the cell with the defined donating region. To conserve fluid volume and to preserve fluid shape, such a scheme must satisfy two requirements;

1. donating regions for any two boundaries must not overlap, otherwise VOF conservation is not assured, and

2. donating regions must contain a total volume equal to the volume of fluid and void which is fluxed over the associated boundary during the computational timestep.

The method used to define a donating region is illustrated in Figure 5.8. In this Cartesian example of unit depth, fluid leaves the cell over the right boundary with velocity $u_R$, enters the cell over the bottom boundary with velocity $v_B$ and leaves the cell over the top boundary with velocity $v_T$. Thus
for this cell, the right and top boundaries are donating boundaries which require defined donating regions, while the bottom boundary is an accepting boundary requiring no defined region within this cell. There is no fluid flow over the left boundary.

As shown in the figure, each donating region is defined as a trapezium having two faces parallel, or in the limiting case of a trapezium with one zero length parallel side, a triangle.

The velocity of each adjacent boundary is used to determine the gradient of each of the non-parallel trapezium sides. Considering the donating region $FBCG$ attached to the right donating boundary $BC$ in Figure 5.8, we note that the adjacent lower boundary $DC$ is an accepting boundary, and thus requires no donating region in this cell. Consequently, the lower trapezium boundary $GC$ is parallel to the lower boundary $DC$. Conversely, the upper donating boundary $AB$ does require an attached donating region, so the gradient of this trapezium side $FB$ is set to the inverse ratio of velocities of
the two adjacent boundaries. In this case,
\[
\frac{dy}{dx}\bigg|_{FB} = \frac{v_T}{u_R}.
\]  
(5.9)

The idea behind this gradient is as follows. Given a uniform flow field throughout the cell having horizontal and vertical components of \(u_R\) and \(v_T\) respectively, fluid residing on the line \(EB\) would pass through the point \(B\) when exiting the cell. Fluid above the line \(EB\) would pass through the top boundary \(AB\), while fluid below the line \(EB\) would pass through the right boundary \(BC\). Thus, the line \(EB\) defines the intersection of the two donating regions.

Once the gradients of the two non-parallel sides have been set, the position of the internal parallel side is set to give the donating region volume equal to the total fluid and void flux through the donating boundary. To locate the trapezium side \(FG\) in Figure 5.8, the volume \(V_{FBCG}\) of the trapezium \(FBGC\) must satisfy
\[
V_{FBCG} = \frac{1}{2} (Y_{BC} + Y_{FG}) X_{GC} = u_R \delta y \delta t,
\]  
(5.10)

where \(\delta y\) is the height of the cell, \(\delta t\) is the timestep and \(X\) and \(Y\) are lengths in the horizontal and vertical directions, respectively.

Further examples of donating regions for different combinations of fluid boundary velocities are given in Figure 5.9. Case (A) shows a square cell with equal magnitude velocities over each boundary. The contents of the entire cell is removed within the timestep, the lower right half out the right boundary, the upper left half out the top boundary. Case (B) shows two adjacent donating boundaries with unequal magnitudes. Three donating boundaries are shown in case (C). Case (D) shows two opposite donating boundaries of equal magnitude which, as in case (A), remove the entire contents of the cell within the timestep. Case (E) demonstrates the donating regions in a cell with unequal cell dimensions.
Figure 5.9: Example donating regions defined using boundary velocities.
Stability Analysis

A stability analysis of the donating region model is undertaken to provide the maximum stable timestep. The analysis also shows that for any given set of boundary velocities satisfying the discretised continuity equations, and for any rectangular cell dimensions, donating regions within the cell can be defined.

For stability the volume of fluid and void entering the cell during a time step must be less than the volume of the cell, otherwise any geometrical information concerning the orientation and volume of fluid within the cell becomes meaningless. Representing the total influx of fluid and void volume as $V_{IN}$, the total volume flowing out as $V_{OUT}$ and the total volume of the cell as $V_{CELL}$, we have

$$V_{IN} \leq V_{CELL}. \quad (5.11)$$

For conservation of fluid volume,

$$V_{IN} = V_{OUT} \quad (5.12)$$

$$\therefore V_{OUT} \leq V_{CELL} \quad (5.13)$$

Combining equations (5.11) and (5.13) gives,

$$V_{IN} + V_{OUT} \leq 2V_{CELL}. \quad (5.14)$$

Noting that the sum of volume inflow and volume outflow is the total volume flow over all cell boundaries, and using the velocity notation defined in Figure 5.10 for the illustrated Cartesian cell,

$$|V_T| + |V_R| + |V_B| + |V_L| \leq 2V_{CELL}. \quad (5.15)$$
Therefore,

\[
|v_T| \frac{\delta t}{\delta y} + |u_R| \frac{\delta t}{\delta x} + |v_B| \frac{\delta t}{\delta y} + |u_L| \frac{\delta t}{\delta x} \leq 2 \quad (5.16)
\]

Equation (5.16) is satisfied at each boundary if the timestep satisfies either

\[
\left| u_{i+\frac{1}{2}} \right| \delta t \leq \frac{1}{2} \min (\delta x_i, \delta x_{i+1}) \quad (5.17)
\]

for the \( x \) direction or equivalently for the \( y \) direction

\[
\left| v_{j+\frac{1}{2}} \right| \delta t \leq \frac{1}{2} \min (\delta y_j, \delta y_{j+1}) . \quad (5.18)
\]

Here the notation is taken from Figure 4.2.

Equations (5.17) and (5.18) are a form of the Courant condition equation (5.7), where \( C = 0.5 \) and \( \delta x \) and \( \delta y \) are the minimum cell dimensions from either side of each cell boundary. Our analysis performed in two dimensions has resulted in a lower constant necessary for stability than given in Nichols et al. [1980], where the analysis was performed in only one dimension. In
cylindrical coordinates equation (5.17) becomes,
\[
\left| u_{i+\frac{1}{2}} \right| \delta t \leq \frac{1}{4x_{i+\frac{1}{2}}} \min\left( x_{i+\frac{1}{2}}^2 - x_{i-\frac{1}{2}}^2, x_{i+\frac{3}{2}}^2 - x_{i+\frac{1}{2}}^2, x_{i+\frac{1}{2}}^2 - x_{i-\frac{1}{2}}^2, x_{i+\frac{3}{2}}^2 - x_{i+\frac{1}{2}}^2 \right),
\]
and equation (5.18) remains unchanged.

As previously discussed, donating regions must not overlap, and must contain the total volume of fluid and void which is fluxed over the associated boundary during the timestep. To demonstrate that donating regions can always be defined within a cell, we examine the four possible cases of donating boundary layout.

Figure 5.11: Four possible donating boundary layouts.
One donating boundary. Equations (5.17), (5.18) and (5.19) are equivalent to

\[ |V_{\text{BOUNDARY}}| \leq \frac{1}{2} V_{\text{CELL}}, \quad (5.20) \]

where \(V_{\text{BOUNDARY}}\) is the total volume flux over any cell boundary. For the case of only one donating boundary in the cell, as shown in Figure 5.11(A), equation (5.20) shows that the maximum volume the donating region can have is half the cell volume. Thus, in this trivial one-dimensional case the rectangular region \(EBCF\) can always be defined.

Two Adjacent Donating Boundaries. We wish to show that given the region boundary gradient assumption of equation (5.9) and the stability criterion of equation (5.20), the donating regions can always be defined. Examining the example shown in Figure 5.11(B), where \(v_T/u_R \leq \delta y/\delta x\), the volume of the triangular region is

\[ V_{ABE} = \frac{1}{2} Y_{AE} X_{AB} = \frac{1}{2} \delta x Y_{AE}. \quad (5.21) \]

Employing the gradient assumption equation (5.9),

\[ \frac{dy}{dx} \bigg|_{FB} = \frac{Y_{AE}}{X_{AB}} = \frac{v_T}{u_R} = \frac{V_T}{\delta x \delta t} \cdot \frac{\delta y \delta t}{V_R}. \quad (5.22) \]

\[ \therefore \frac{V_T}{V_R} = \frac{Y_{AE}}{\delta y} \quad (5.23) \]

Substitution into equation (5.21) yields

\[ V_{ABE} = \frac{1}{2} \delta x \delta y \frac{V_T}{V_R} = \frac{1}{2} V_{\text{CELL}} \frac{V_T}{V_R} \quad (5.24) \]

Employing the stability criterion equation (5.20) for the right boundary

\[ V_R \leq \frac{1}{2} V_{\text{CELL}} \quad (5.25) \]
gives

\[ V_T \leq V_{ABE}. \]  

(5.26)

Further, if the stability assumption is also applied at the top boundary

\[ V_T \leq \frac{1}{2} V_{CELL}, \]  

(5.27)

noting that the volume of the trapezium is

\[ V_{EBCD} = \frac{1}{2} (Y_{ED} + Y_{BC}) X_{AB} = \frac{1}{2} (2\delta y - Y_{AE}) \delta x, \]  

(5.28)

the right volume flux is

\[ V_R \leq V_{EBCD} \left( \frac{\delta y^2}{(2\delta y - Y_{AE}) Y_{AE}} \right). \]  

(5.29)

The bracketed term of equation (5.29) has a maximum value of 1 for real values of \( Y_{AE} \), giving

\[ V_R \leq V_{EBCD}. \]  

(5.30)

From equations (5.26) and (5.30), the volume of fluid fluxed through each boundary is contained within each associated region. Thus, provided the fluxes satisfy the stability criterion equation (5.20), both donating regions are defined. The choice of boundaries for this example was arbitrary, so the proof holds for any two adjacent donating boundaries.

**Two Opposite Donating Boundaries** This case is an extension of the trivial one dimensional, one donating boundary case. As shown in Figure 5.11(C), provided the stability criterion equation (5.20) is met, the rectangular regions \( AGHD \) and \( EDCF \) can be defined.

**Three Adjacent Donating Boundaries** The maximum volumes for the three donating regions illustrated in Figure 5.11(D) are

\[ V_{ABE} = \frac{1}{2} \delta x (\delta y - Y_{EF}), \]  

(5.31)
\[ V_{AEFD} = \frac{1}{2} X_{DF} (\delta y + Y_{EF}) \]  
(5.32)

and

\[ V_{EBCF} = \frac{1}{2} (\delta x - X_{DF}) (\delta y + Y_{EF}). \]  
(5.33)

For the donating regions to be defined, their volumes must be contained within these maxima.

Employing the region boundary gradient assumption, equation (5.9), at \( EB \) and \( AE \), respectively gives

\[ V_T = V_R \frac{\delta x (\delta y - Y_{EF})}{\delta y (\delta x - X_{DF})} \]  
(5.34)

and

\[ V_L = V_R \frac{\delta x (\delta y - Y_{EF})}{\delta y X_{DF}}. \]  
(5.35)

Combining equations (5.34) and (5.35) for the left volume flux gives,

\[ V_L = V_R \frac{X_{DF}}{\delta x - X_{DF}}. \]  
(5.36)

Now, examining the stability criterion equation (5.20) for the lower boundary,

\[ |V_B| \leq \frac{1}{2} V_{CELL}. \]  
(5.37)

The lower boundary is the only boundary with fluid entering the cell, so noting the fluid conservation equation (5.12),

\[ V_L + V_T + V_R \leq \frac{1}{2} V_{CELL}. \]  
(5.38)

Combining equations (5.33), (5.34), (5.36) and (5.38) gives for the volume flux over the right boundary

\[ V_R \leq V_{EBCF} \left[ 2 - \frac{Y_{EF}}{\delta y} + \left( \frac{Y_{EF}}{\delta y} \right)^2 \right]^{-1}. \]  
(5.39)
Examination of the bracketed denominator on the right shows that it has a minimum of 1.75 for real values of $Y_{EF}$. Thus,

$$V_R \leq V_{EBCF}. \quad (5.40)$$

Turning to the left region, we have from equations (5.32), (5.36) and (5.39),

$$V_L \leq V_{AEFD} \left[ 2 - \frac{Y_{EF}}{\delta y} + \left( \frac{Y_{EF}}{\delta y} \right)^2 \right]^{-1} \quad (5.41)$$

$$\therefore, V_L \leq V_{AEFD}. \quad (5.42)$$

For the top region, we have from equations (5.31), (5.34) and (5.39),

$$V_T \leq V_{AED} \left[ 1 + \frac{1 - \left( \frac{Y_{EF}}{\delta y} \right)^2}{1 - \frac{Y_{EF}}{\delta y}} \right]^{-1} \quad (5.43)$$

The bracketed term has a minimum of 2 for real values of $Y_{EF}$, so

$$V_T \leq V_{AED}. \quad (5.44)$$

Equations (5.40), (5.42) and (5.44) show that donating regions can always be defined within a three donating boundary cell, provided the stability criterion equation (5.20) is met. As a cell cannot possess four donating boundaries, the existence of donating regions for any combination of donating boundaries has been proven.

In cylindrical coordinates the total boundary fluxes are given by,

$$V_T = \pi \left( x_i^2 - x_{i-1}^2 \right) v_T, \quad (5.45)$$

$$V_B = \pi \left( x_i^2 - x_{i-1}^2 \right) v_B, \quad (5.46)$$

104
\[ V_R = 2\pi x_i \delta u_R \]  (5.47)

and

\[ V_L = 2\pi x_{i-1} \delta u_L, \]  (5.48)

and the existence of donating regions proof still applies. A small error in calculating the donating region geometry and fluid position within cylindrical cells is introduced by assuming the internal geometry of the cell is Cartesian, however this error decreases with cell dimensions, and is negligible in all practical cases. This error does not affect VOF conservation.

### 5.3.2 Free Surface Reconstruction

A better free surface reconstruction is achieved by representing the fluid interface within each cell as a straight line having an infinitely variable gradient. Similar linear interface models have been used to good effect by Debar [1974], Youngs [1982] and Rider & Kothe [1998]. The method involves two steps for each interface reconstruction: determining the gradient of the interface within each cell, and then positioning the interface within each cell to equate the calculated cell VOF volume to the volume contained between the free surface and cell boundaries.

#### Free Surface Gradient Calculation

There are two methods available in BOUNCE for calculating the free surface gradient.

**The RIPPLE Method**  This method of calculating the interface gradient follows that used in the RIPPLE code [Kothe et al. 1994]. Interface normals
are first calculated at each of the four vertices of the cell as the gradient of the VOF function,

\[ \mathbf{n} = \nabla F. \]  

(5.49)

Following [Kothe et al. 1994], and referring to Figure 5.12 for notation, equation (5.49) becomes

\[
\begin{align*}
\mathbf{n}_{i+1/2,j+1/2} &= \left\{ \frac{(F_{i+1,j+1} - F_{i+1,j}) \delta y_j + (F_{i+1,j} - F_{i,j}) \delta y_{j+1}}{\delta y_j + \delta y_{j+1}} \right\} \mathbf{i} \\
&\quad + \left\{ \frac{(F_{i+1,j+1} - F_{i+1,j}) \delta x_i + (F_{i,j+1} - F_{i,j}) \delta x_{i+1}}{\delta x_i + \delta x_{i+1}} \right\} \mathbf{j}. 
\end{align*}
\]  

(5.50)

Cell centred normals are calculated by averaging these vertex normals, so that the centred normals are effectively calculated using VOF data from a 3×3 mesh,

\[
\mathbf{n}_{i,j} = \frac{1}{4} \left( \mathbf{n}_{i+1/2,j+1/2} + \mathbf{n}_{i+1/2,j-1/2} + \mathbf{n}_{i-1/2,j-1/2} + \mathbf{n}_{i-1/2,j+1/2} \right). 
\]  

(5.51)
The gradient of the fluid interface is set using this cell centred normal.

The RIPPLE method is simple and robust, and as we shall demonstrate later, delivers reasonable accuracy for a relatively small computational expense. However, more complex gradient estimation techniques can yield slightly improved gradient estimations [Rider & Kothe 1998]. Given that interface reconstruction consumes only a small percentage of computational time during each solution timestep, their use in real fluid computations is often justified.

**Error Minimisation Method**  The error minimisation method presented here is based on the concept pioneered by Pilliod Jr [1992], however, a significant change to the scheme has been made. As the name implies, under the error minimisation scheme each interface gradient has associated with it an error function. When this error function is minimised, we find the optimal free surface gradient within the cell.

The error function is defined as follows;

1. Given a gradient, the interface is reconstructed such that the volume of fluid contained between the interface and cell boundaries is equal to the volume of fluid within the examined cell. The method used here follows that described in the next section.

2. The fluid interface is continued beyond the boundaries of the examined cell, so that it traverses a total of $3 \times 3 = 9$ cells, with the examined cell at the centre.

3. Volume of fluid functions, $F^*$, are calculated for each of the surrounding 8 cells based on the extended reconstructed interface.
4. Finally, the error associated with the reconstructed interface is calculated. Under the Pilliod scheme the error function is defined as

\[ E_{\text{Pilliod}_{i,j}} = \sum_{n=i-1}^{i+1} \sum_{m=j-1}^{j+1} (F_{i,j} - F^*_{i,j})^2, \]  

(5.52)

where \( F_{i,j} \) is the actual VOF function for cell \( i, j \) and \( F^*_{i,j} \) is the VOF function for cell \( i, j \) based on the extended reconstructed interface, as described above. Under the BOUNCE scheme, the error function is defined as

\[ E_{i,j} = \frac{E_{\text{void}}}{E_{\text{void,max}}} + \frac{E_{\text{fluid}}}{E_{\text{fluid,max}}}, \]  

(5.53)

where

\[ E_{\text{void}} = \sum_{n=i-1}^{i+1} \sum_{m=j-1}^{j+1} \{ \max (F_{i,j} - F^*_{i,j}, 0) \times A_{i,j} \}^3, \]  

(5.54)

\[ E_{\text{void,max}} = \sum_{n=i-1}^{i+1} \sum_{m=j-1}^{j+1} \{ (1 - F^*_{i,j}) \times A_{i,j} \}^3, \]  

(5.55)

\[ E_{\text{fluid}} = \sum_{n=i-1}^{i+1} \sum_{m=j-1}^{j+1} \{ \max (F^*_{i,j} - F_{i,j}, 0) \times A_{i,j} \}^3 \]  

(5.56)

and

\[ E_{\text{fluid,max}} = \sum_{n=i-1}^{i+1} \sum_{m=j-1}^{j+1} \{ F^*_{i,j} \times A_{i,j} \}^3. \]  

(5.57)

The BOUNCE error function differs from the Pilliod function in two significant ways. Firstly, the BOUNCE function comprises of two separate errors, one resulting from the fluid region adjacent to the reconstructed interface and the other from the void region adjacent to the reconstructed interface. As these errors are normalized separately against the maximum achievable error for each reconstructed region, the total error function is not dependent on the relative sizes of the reconstructed void and fluid regions.
To illustrate, consider the somewhat extreme example of Figure 5.13 where a void region contains a solitary central fluid cell having $F_{i,j} = 0.9$. Defining $\theta$ to be the angle between the reconstructed interface and the horizon, Figure 5.14 shows a plot of both the BOUNCE and Pilliod errors as a function the orientation of the interface.

As shown in Figure 5.14, the Pilliod error displays several minima, despite the symmetry of the fluid layout. Variations of the Pilliod error function here result from changes in the relative sizes of the void and fluid regions adjacent to the reconstructed interface as the interface is rotated. The BOUNCE error is constant in this example, in line with the indeterminate nature of the fluid geometry.

The BOUNCE error function also differs from the Pilliod error function in the value of the exponent used. BOUNCE uses an exponent of three, while
the Pilliod function uses an exponent of two. The higher exponent weights the error function more heavily against interface orientations causing high errors in individual cells. While the higher exponent used in BOUNCE does give slightly better free surface gradient estimations than the Pilliod exponent, in practice the difference in performance between the two alternative exponents is only slight.

Evaluation of the optimal gradient is accomplished numerically using the following subroutines and functions:

1. ORIENTATE: This subroutine loops through all cells within the computational domain calculating the RIPPLE estimate for the surface orientation. It calls subroutine MINGRAD to improve the estimate using the error minimisation technique. ORIENTATE also sets the surface orientation in cells bounding the computational domain, and
calculates surface curvatures.

2. **MINGRAD**: This subroutine sets the minimisation problem up for subroutine **MINMINIMISE**.

3. **MINMINIMISE**: Here the orientation error is bounded and minimised.

   (a) Bounding. Initially, the error is evaluated at three orientations; the orientation given by the RIPPLE method, and at orientations just below and above this orientation. If the gradient of the error function with respect to surface orientation is indeterminate, based on the three evaluated points, then the optimal orientation is assumed to be equal to the RIPPLE estimate. Otherwise, the algorithm steps down the error gradient towards the minimum error orientation until this minimum has been bounded.

   (b) Minimisation. Subroutine **GOLDEN** is called to minimise the error function.

4. **GOLDEN**: Using the Golden section search method [Press et al. 1992], this subroutine iterates to give the optimal surface orientation specified by the minimum error function. The tolerance used in determining this error is set to $1 \times 10^{-3}$ rad. **GOLDEN** uses function **MINERROR**.

5. **MINERROR**: Given an interface orientation and central cell, this function loops around surrounding cells calculating the reconstructed interface error function. It uses **SLICEMIN** and **INOUT**.

6. **SLICEMIN**: Given an interface position, **SLICEMIN** calculates the corresponding Volume of Fluid value for that cell.
7. **INOUT**: Given an interface position and a point, this subroutine calculates whether the point is in fluid or void region.

Further details of these algorithms can be found in the code listing, given in Appendix E.

As examined in [Rider & Kothe 1998], error minimisation methods give a higher accuracy estimate of the free surface gradient than the RIPPLE scheme, but require approximately five times the computational expense. These schemes are expensive because for each iterative gradient improvement, the free surface must be constructed in not only the examined cell, but the eight surrounding cells as well.

An advantage of the error minimisation method is that the VOF gradient is evaluated at the location of the free surface within each computational cell. Under the RIPPLE method, the gradient is evaluated at the centre of each computational cell. This suggests that the error minimisation method is a better choice for evaluating gradients which are to be used by free surface advection algorithms.

Further refinement of either the error minimisation algorithm or RIPPLE gradient algorithm might be achieved by using a larger number of cells to perform each gradient calculation, for example a $5 \times 5$ mesh rather than a $3 \times 3$ mesh. However, there would be disadvantages of using such higher order, larger stencil schemes. As the VOF function does not vary continuously within fluid regions, but rather varies between zero and one over a few transitional surface cells, calculating surface normals over this transition region using a larger stencil may not increase surface gradient accuracy. Also, using a stencil larger than that of $3 \times 3$, as presently used, would be considerably more expensive, and would add significant complexity to gradient calculations near computational boundaries, where presently only one layer
of dummy cells is provided beyond the computational region.

An evaluation of the effects of the different gradient calculation schemes on a variety of fluid advection problems is included in Chapter 7.

**Free Surface Interface Positioning**

The position of the interface within the cell is set to give the volume of fluid beneath the interface equal to the volume of fluid contained in the cell.

In BOUNCE, the fluid position within a cell is described by two points - a left point and a right point. Both points lie on a cell boundary and both points lie on the interface line. Also, the relationship between the points determines the position of the fluid within the cell. When looking along the interface line from the left point towards the right point, fluid is located to the left of the interface.

As shown in Figure 5.15, fluid interfaces are classified into eight different types. Types one and two are for exactly vertical and horizontal interfaces, respectively, while the remaining six types each describe the pair of cell boundaries through which the fluid interface passes. The interface type does not describe on which side of the interface line fluid lies. This is described by the relationship between the left and right points, as discussed previously.

The logical steps used to determine the cell interface type are shown schematically in Figure 5.16. First, $i_{type}$ is set for horizontal and vertical surfaces. Comparisons between the cell dimensions and interface normal components are then made to determine whether the interface has a negative or positive gradient, and whether it is closer to vertical or horizontal with respect to the cell dimensions. An interface is deemed closer to vertical if it has a gradient of larger magnitude than the diagonal of the cell, and closer to horizontal if it has a gradient of smaller magnitude than the diagonal of
Figure 5.15: Interfaces are classified into eight different types in BOUNCE, depending on the location and orientation of the interface. The interface type is represented by the computational variable \textit{itype}.
Figure 5.16: Steps used to determine the fluid interface type within each cell. The cell has dimensions $\delta x$ and $\delta y$ and has the cell centred VOF normal $\mathbf{n}_{i,j} = n_x \mathbf{i} + n_y \mathbf{j}$.

Once the gradient characteristics are known, the corrected fluid volume $F_{low}$ is compared with the volumes $F_{min}$ and $F_{max}$ to determine the correct interface type. The fluid volume is corrected before comparison so that a cell
Figure 5.17: Example surface reconstructions showing $F_{\text{min}}$ (A) and $F_{\text{max}}$ (B) for a negative gradient interface closer to vertical than horizontal. Also shown are the location of the left and right points for an $itype = 4$ reconstruction with the fluid positioned to the left of the interface (C).

with fluid volume $F$ and interface normal $\mathbf{n}$ is calculated using the same code as the complementary cell having volume $1 - F$ and interface normal $-\mathbf{n}$. As shown in the examples of Figure 5.17(A) and (B), $F_{\text{min}}$ and $F_{\text{max}}$ are critical fluid volumes representing the boundaries between three possible interface types. If either $F_{\text{low}} < F_{\text{min}}$ or $F_{\text{low}} > F_{\text{max}}$ then the interface intersects two adjacent cell boundaries, otherwise the interface intersects the two opposite boundaries. The interface type is thus set accordingly.

Once $itype$ has been determined, final positioning of the interface is accomplished by equating the volume of fluid contained under the interface to the volume in the cell. In the example of Figure 5.17(C), where the interface has a negative gradient closer to vertical than horizontal, the fluid is located to the left of the interface line, $itype = 4$, and the left and right interface
points are given by

\[ x_{\text{left}} = x_{i-1} + F_{i,j} \delta x + \frac{1}{2} \delta y \frac{n_y}{n_x}, \quad (5.58) \]

\[ y_{\text{left}} = y_{j-1}, \quad (5.59) \]

\[ x_{\text{right}} = x_{i-1} + F_{i,j} \delta x - \frac{1}{2} \delta y \frac{n_y}{n_x}, \quad (5.60) \]

and

\[ y_{\text{right}} = y_j. \quad (5.61) \]

In cylindrical coordinates the interface positioning is accomplished using internal cell volumes calculated in Cartesian coordinates. This introduces a small interface gradient error, but like the cylindrical treatment of the defined donating regions, the error is negligible for all practical cases and does not effect fluid conservation.

### 5.3.3 Integration of the VOF Fluxes

The amount of flux donated over each boundary is calculated as the volume intersection between the defined donating region of the donating cell and the reconstructed position of fluid within that cell.

In the example of Figure 5.18, the fluid interface is of type five, where \( G \) is the left interface point and \( E \) is the right interface point. Thus, within this cell, the fluid volume lies within the quadrilateral \( EGDH \). The velocity over boundary \( AD \) is positive, so the boundary has a donating region associated with it, namely the trapezium \( ADCB \). The volume of fluid fluxed over \( AD \) is the intersection between the fluid region \( EGDH \) and the donating region.
Figure 5.18: Example flux calculation over a right donating boundary.

$ADCB$, the region $FGDC$. In this example, the amount fluxed is

$$\delta F_D = \frac{1}{2\delta x\delta y} (Y_{FC} + Y_{GD}) X_{CD},$$  \hspace{1cm} (5.62)

where $\delta x$ and $\delta y$ are the cell dimensions.

In Cartesian coordinates, the change in the VOF value of the donating cell is calculated using,

$$F_A^* = F_A + \frac{\delta F_D \delta x_D}{\delta x_A}$$  \hspace{1cm} (5.64)

while the VOF value for the accepting cell is incremented using either

$$F_A^* = F_A + \frac{\delta F_D \delta y_D}{\delta y_A}$$  \hspace{1cm} (5.65)

in the $x$ direction, or in the $y$ direction,

Here $F^*$ are the incremented VOF values, $\delta F_D$ is the calculated VOF loss from the donating cell, $\delta x$ and $\delta y$ are cell dimensions and subscripts $A$ and $D$ refer to the accepting and donating cells, respectively.
In cylindrical coordinates equation (5.65) still holds in the $y$ direction, but in the radial direction equation (5.64) is replaced by

$$F_A^* = F_A + \delta F_D \frac{(x_{D+}^2 - x_{D-}^2)}{(x_{A+}^2 - x_{A-}^2)},$$  \hspace{1cm} (5.66)

where the previous notation is supplemented by $x_+$ and $x_-$, the positions of the boundaries on either side of the relevant cell. Equation (5.63) still holds in both directions.

### 5.3.4 The Code Structure

Figure 5.19 shows schematically how the VOF calculation is accomplished using the subroutine $VFGRAD$. In similar fashion to the SOLA-VOF subroutine $VFCONV$, all boundaries are first checked to ensure that the VOF advection stability criterion is not being violated. A loop through each cell within the computational region then calculates first $itype$ for the cell, and then the left and right interface points. A third loop, this time through all cell boundaries within the computational region, calculates the VOF advection over the boundary, and increments these changes to the VOF values for the adjacent donating and accepting cells. This calculation is performed for both the $x$ and $y$ direction boundaries.

In determining the VOF change over each boundary, the utility function $SLICEX$ is used to calculate the volume intersection between the given fluid region and donating region. The $SLICEX$ function, which requires the four vertex points of the donating region and the left and right points of the fluid interface, calculates the intersection region by first determining where the fluid interface line intersects with the boundaries of the donating region. A listing of the $SLICEX$ function, with descriptive comments, is given as part of the code listing, in Appendix E.
SUBROUTINE VFGRAD

Loop through all cells within computational region

Calculate product of velocity and timestep over upper and right cell boundaries

Check stability criterion over both upper and right cell boundaries

Satisfied

Not Satisfied

Return to main program and repeat with half timestep

Loop through all cells within computational region

Determine type from fluid interface normal and VOF value

Using type, determine the 'left' and 'right' intersection points which describe the fluid interface within the cell

Loop through all boundaries within computational region

Determine which cell is a donating cell and which is an accepting cell over the x direction boundary

Calculate geometry of donating region for the x direction boundary - region defined by the four vertex points

Use the function SLICE to determine the volume intersection between the fluid region and donating region for the x direction boundary

Increment the VOF function in the x direction donor and acceptor cells

Repeat VOF calculation for the y direction boundary

Loop through all cells within computational region

Reset new VOF value of cell to zero if less than emf

Reset new VOF value of cell to one if greater than emf

Return to main program

Figure 5.19: Schematic of the BOUNCE subroutine VFGRAD.
After all VOF changes have been calculated, a final loop through all computational cells is used to adjust any new VOF values that are unphysical. The procedure used here is the same as that used in the final stages of the SOLA-VOF subroutine \textit{VFCONV} — if $F_{i,j}$ is less than $emf \ (\varepsilon_F)$, it is set to zero, or alternatively if $F_{i,j}$ is greater than $emf1 \ (1 - \varepsilon_F)$, it is set to one. The computational variable $vchgt$ records any changes in fluid volume resulting from these adjustments.

5.3.5 Translation Test - The Box Problem

Figure [5.20] shows the previously described test case box problem (see Section 5.2.1) now calculated using the Defined Donating Region VOF advection method. Interface gradients have been calculated using the error minimisation algorithm. The computational timestep, fluid velocity, box dimensions and grid geometries remain unchanged from the H-N algorithm test.

Comparing the results of the H-N algorithm test box, shown in Figure 5.1, with the results from the BOUNCE algorithm, shown in Figure 5.20, shows that the new algorithm is significantly more accurate in predicting VOF densities. The shape of the box after the last BOUNCE iteration has not ‘flattened’ tangentially to the extent that the H-N test case did, and the total distance moved by the centre of the box is predicted accurately. Also, there is no wisp generation evident at the new box corners moving parallel to the fluid velocity, despite the algorithm not including any wisp suppressing features. Fluid volume is now rigorously conserved.

Some diffusion of the box shape has still occurred in Figure 5.20, as evidenced by rounding of the box corners and some spreading of the box in a direction normal to the velocity direction. The rounding of the corners is a feature of the finite grid size and interface reconstruction method used in
calculating boundary fluid fluxes — for ‘real’ fluid calculations, where free surface curvatures are of much larger magnitude than cell dimensions, the effect of such rounding becomes negligible.

Spreading of the box shape is caused by the staggered location of the fluid velocity components, as previously described under the H-N algorithm case test box Section [5.2.1]. The test case chosen here represents a worse case scenario for this type of diffusion, as the timestep is quite large, and the curvature of the corners is infinitely small compared with the grid dimensions. While this diffusion is reduced with decreasing time step and decreasing cell
dimensions, the second method we present, namely the Stream method, is more effective in reducing this type of diffusion. It is the Stream method that we turn our attention to in the next chapter.
Chapter 6

The Stream VOF Convection Algorithm

6.1 The Basic Method

The Defined Donating Region algorithm has a number of advantages over the H-N algorithm, but as previously discussed, suffers from staggered velocity component induced diffusion. More specifically, fluid interfaces are dispersed under the Defined Donating Region scheme because fluid velocities are assumed to have only a normal component at each cell boundary. The Stream VOF convection algorithm was developed as part of this work to counteract this phenomenon.

The idea behind the Stream algorithm is simple, and can be summarized as follows:

1. Fluid interfaces are reconstructed in each cell, as per the Defined Donating Region Scheme.

2. A continuous velocity field is defined throughout the computational
region, based on the staggered cell boundary velocities.

3. Donating regions for each cell boundary are defined by integrating back in time for the duration of the computational timestep along fluid streamlines passing through the examined boundary.

4. Boundary fluxes are calculated as the intersection between each donating region and all fluid locations, as per the Defined Donating Region method.

Alongside simplicity, the Stream method has a number of other advantages. Fluid crossing a given boundary can originate from any cell, not just the two cells adjacent to the boundary. This property is of particularly importance in problems with highly varying vorticity, as will be demonstrated later. Another advantage of the Stream method is that provided the donating regions are integrated backwards in time from each examined cell boundary exactly, fluid and void volume conservation is assured. Finally, the Stream method is not limited by any conventional stability criterion, but naturally, accuracy decreases as the computational timestep is increased.

The difficulties involved in implementing the Stream algorithm lie in defining a suitable continuous velocity field, and in integrating along streamlines to determine fluid volume fluxes occurring over each boundary. It is these two topics which we now examine.

### 6.2 Defining the Velocity Field

The velocity at any point in a given cell \(i, j\) is defined as

\[
V(x, y) = (\chi_b x + \chi_y) \mathbf{i} + (-\chi_b y - \chi_x) \mathbf{j}.
\]  

(6.1)
where
\[ \chi_b = \frac{u_{i+\frac{1}{2}} - u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} = -\frac{v_{j+\frac{1}{2}} - v_{j-\frac{1}{2}}}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}}, \] (6.2)

\[ \chi_x = -\chi_b y_{j-\frac{1}{2}} - v_{j-\frac{1}{2}} \] (6.3)

and
\[ \chi_y = -\chi_b x_{i-\frac{1}{2}} + u_{i-\frac{1}{2}}. \] (6.4)

Cell boundary velocities are defined using the notation of Figure 4.2. Note that the equality in equation (6.2) follows from the continuity equation applied to the examined cell, namely
\[ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = \frac{u_{i+\frac{1}{2}} - u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} + \frac{v_{j+\frac{1}{2}} - v_{j-\frac{1}{2}}}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}} = 0. \] (6.5)

The velocity field given in equation (6.1) can also be expressed in terms of the stream function,
\[ \Psi = \chi_b xy + \chi_x x + \chi_y y, \] (6.6)

where the standard stream function has been defined via
\[ u = \frac{\partial \Psi}{\partial y} \quad \text{and} \quad v = -\frac{\partial \Psi}{\partial x}, \] (6.7)

and we have assumed the boundary condition for integration, \( \Psi (0,0) = 0 \).

Equation (6.1) specifies a simple velocity field which varies linearly between cell boundaries, and satisfies the continuity equation everywhere. The normal components of the velocity field at cell boundaries are continuous between cells, but the tangential components are not. The velocity field is not continuous at cell vertex points. The velocity field discontinuities would be problematic if we were to integrate exactly along streamlines from cell vertices, but the integration method we present later avoids this inconvenience.
The stream function defined by equation (6.6) is continuous throughout the computational domain. Note that fluid fluxes over cell boundaries are constant along the length of each boundary, and consequently streamlines intersecting each boundary, which are separated by equal volume fluxes, are equally spaced.

Examples of various in cell velocity fields, represented as fluid streamlines, are shown in Figure 6.1. The cells shown in the figure have the non-dimensional dimensions of 1×1, and the corresponding velocity field constants are shown alongside each streamline trace. Cases (D) and (E) in Figure 6.1 are the only cases with \( \chi_b \neq 0 \), and consequently the streamlines in these two examples are curved.

### 6.3 The Fluid Boundary Flux Calculation

Ideally, we would like to integrate equation (6.1) exactly with respect to time to determine donating regions associated with each boundary, and then determine the intersection between these regions and the fluid regions to calculate fluid boundary fluxes. In practice however, such a procedure would be extremely computationally complex for two main reasons;

1. A fluid particle that flows through a particular boundary during a discrete timestep, and exists on a particular streamline, may pass through many cells on route to that boundary. It is the starting positions of such particles that define a donating region for a given cell boundary. The difficulty is that integrating along the path of a fluid particle, or streamline, becomes more and more complex the more cells the particle passes through, as each cell has a different velocity field. As a result, defining donating regions for boundaries which are fluxing fluid from
Figure 6.1: Example in cell velocity fields, represented by fluid streamlines. In each case 50 Streamlines are shown, equally spaced in the range $-1 \leq \Psi \leq 1$. 
many cells becomes impractically complex. Determining the intersection between these irregular donating regions and the reconstructed fluid regions would add further complications.

2. As previously discussed, the velocity field defined by equation (6.1) is continuous throughout the computational domain, except at cell vertices. Thus, trying to integrate away from vertex points to define the fluid donating regions may not be computationally possible.

It is for these reasons that an approximate method of integration has been developed to determine fluid boundary fluxes.

### 6.3.1 Approximate Integration Method

The theory behind this approximate integration method is simple. Each boundary flux is split into a discrete number of streamtubes or ‘channels’, each channel representing an equal flux of total fluid and void volume. Fluid streamlines, determined using the velocity stream equation (6.6), define the upper and lower boundaries of each channel. The length of each channel is determined by the length of the computational timestep. The number of channels that each boundary is split into, \( n_{\text{channel}} \), is determined by the user — the more channels the greater the accuracy of integration.

To illustrate this point, Figure 6.2 shows an example where the velocity at the right boundary of cell \( i, j \) is positive, and fluid is fluxing through this boundary from cells \( i, j \) and \( i, j + 1 \) during the solution timestep. This right boundary of cell \( i, j \) is referred to as the ‘final’ boundary, as it is the last boundary a fluid particle would pass through before entering cell \( i + 1, j \).

The volume of fluid in each channel which is fluxed through a particular
Figure 6.2: Example channel used to calculate the fluid flux over the right boundary of cell \(i, j\).

final boundary during a timestep \(\delta t\), is approximately

\[
V_{\text{fluid channel}} \approx \int_0^L w(l) F(l) \, dl, \quad (6.8)
\]

where a cell of unit depth has been assumed. In equation (6.8), \(F(l)\) is the VOF function evaluated along the central characteristic streamline of the channel, \(w\) is the non-constant width of the channel measured normal to the central channel streamline, and \(l\) is a streamwise coordinate along the length of the channel, directed upstream from the final boundary. The streamwise length of the channel, \(L\), is determined by the length of the timestep, \(\delta t\), as

\[
L = \int_0^{\delta t} |\mathbf{v}(l)| \, dt, \quad (6.9)
\]

where it has been noted that the fluid velocity within the channel is always directed parallel to the streamwise coordinate, \(l\), and as such

\[
|\mathbf{v}(l)| = -\frac{dl}{dt}, \quad (6.10)
\]
The variables $l$, $L$, and $w$ are shown in Figure 6.2.

Noting that no fluid can cross the fluid streamlines defining the upper and lower boundaries to the channel, the total volume flux of fluid and void at any point along the channel is given by, for the example of Figure 6.2,

$$V_{\text{total channel}} = \text{constant} = \frac{u_{i,j} \times \delta y_j}{n_{\text{channel}}}. \quad (6.11)$$

The channel flux can also be expressed as a function of the streamwise coordinate,

$$V_{\text{total channel}} \approx |v(l)| w(l) = -\frac{dl}{dt}(l) w(l), \quad (6.12)$$

giving when combined with equation (6.11),

$$w(l) \approx \left( \frac{u_{i,j} \delta y_j}{n_{\text{channel}}} \right) / \left( -\frac{dl}{dt}(l) \right). \quad (6.13)$$

Substituting into equation (6.8) gives

$$V_{\text{fluid channel}} = -\frac{u_{i,j} \delta y_j}{n_{\text{channel}}} \int_{0}^{L} F(l) \frac{dl}{dt}(l) dl = \frac{u_{i,j} \delta y_j}{n_{\text{channel}}} \int_{0}^{\delta t} F(t) dt \quad (6.14)$$

or

$$V_{\text{fluid channel}} = \frac{u_{i,j} \delta y_j}{n_{\text{channel}}} t_{\text{total fluid}}, \quad (6.15)$$

where $t_{\text{total fluid}}$ is defined as the total time a fluid particle would spend in fluid regions, when moving for time $\delta t$ along the central channel streamline towards the final cell boundary over a stationary fluid geometry.

Once individual channel fluxes have been determined, volume of fluid values are incremented using

$$F_A = F_A + \frac{V_{\text{all fluid channels on boundary}}}{(\delta x \times \delta y)_A} \quad (6.16)$$

and

$$F_D = F_D - \frac{V_{\text{all fluid channels on boundary}}}{(\delta x \times \delta y)_D}. \quad (6.17)$$
In equations (6.16) and (6.17), the subscript $A$ identifies the accepting cell, which is defined as the cell immediately downstream of the final boundary. The subscript $D$ identifies the donating cell, which is the cell immediately upstream of the final boundary. Note that although fluid volumes may be calculated using fluid geometries and velocity fields from other cells, during a boundary flux calculation only the VOF values for the accepting and donating cells are changed.

### 6.3.2 Fluid Particle Trajectory Calculation

In calculating $t_{\text{total fluid}}$ for each channel, we recognise that a channel may pass through more than one cell on route to the final cell boundary. To simplify computational calculations, we divide the total channel length into sections contained within individual computational cells. Thus, we can specify

$$t_{\text{total fluid}} = \sum_{\text{all cells along the central channel streamline}} t_{\text{fluid}}, \quad (6.18)$$

where $t_{\text{fluid}}$ is the time a fluid particle would spend in the fluid region of an individual cell, when moving along the central channel streamline over stationary fluid geometry. We also define $t_{\text{remain}}$ as the length of time elapsed when a fluid particle travels between the start of a central channel streamline and a particular cell. For example, at the final boundary of a channel, $t_{\text{remain}} = \delta t$.

Using these definitions, the integration problem now becomes one of travelling backwards along each central channel streamline, from each examined final boundary, calculating $t_{\text{fluid}}$ for each cell we pass through as we go. When we reach the start of the channel, $t_{\text{remain}} \to 0$, indicating that the channel integration is complete.
The numerical algorithm used to accomplish these calculations is detailed in Section 6.5, however, we first need to outline a few mathematical utilities on which the algorithm relies. In the following, we define $\Psi_c$ as the characteristic stream value for the central channel streamline being calculated. It is evaluated using the stream equation (6.6) at the centre of the channel at the final boundary.

**Length of Channel Remaining**

It is necessary to calculate the starting point for a channel, in order to ascertain whether the channel finishes within a particular cell or not. For the example of Figure 6.2, where fluid is flowing in the positive direction over the right vertical boundary of cell $i, j$, the starting $x$ coordinate of the channel, $x_{end}$, is given by,

$$
  x_{end} = \begin{cases} 
  x_{i+\frac{1}{2}} - \chi_y t_{\text{remain}} & \text{if } \chi_b = 0 \\
  \frac{1}{\chi_b} \left[ (\chi_b x_{i+\frac{1}{2}} + \chi_y) \exp(-\chi_b t_{\text{remain}}) - \chi_y \right] & \text{if } \chi_b \neq 0.
  \end{cases}
$$

(6.19)

**Intersection Between the Cell Boundaries and the Central Channel Streamline**

We also need to know whether a channel continues beyond the boundaries of an individual cell. Again using the example of Figure 6.2, the $x$ coordinate marking the intersection between the central streamline and the upper boundary of cell $i, j$ is given by,

$$
  x_{\text{top}} = \frac{\Psi_c - \chi_x y_{j+\frac{1}{2}}}{\chi_b y_{j+\frac{1}{2}} + \chi_x}
$$

(6.20)

if $\chi_b y_{j+\frac{1}{2}} + \chi_x \neq 0$, otherwise the intersection point is not defined.
Intersection Between a Fluid Interface and the Central Channel Streamline

We need to determine the intersection points between the free surface interfaces and central channel streamlines in each cell that the channel traverses. Free surface lines are defined using the left and right point method, as discussed for the Defined Donating Region method in Section 5.3.2. In calculating the streamline and free surface intersection points, we cast the interface curves into a different form, namely

\[ \lambda_x x + \lambda_y y = \lambda_c. \] (6.21)

The constants appearing in equation (6.21) are evaluated in each cell for the three different types of fluid interface:

1. Horizontal interface.

\[ \lambda_x = 0, \lambda_y = 1 \text{ and } \lambda_c = y_{\text{right or left}} \] (6.22)

2. Vertical interface.

\[ \lambda_x = 1, \lambda_y = 0 \text{ and } \lambda_c = x_{\text{right or left}} \] (6.23)

3. Interface has non-zero, finite gradient.

\[
\begin{align*}
\lambda_x &= \frac{y_{\text{right}} - y_{\text{left}}}{x_{\text{left}} y_{\text{right}} - x_{\text{right}} y_{\text{left}}}; \\
\lambda_y &= \frac{x_{\text{right}} - x_{\text{left}}}{x_{\text{right}} y_{\text{left}} - x_{\text{left}} y_{\text{right}}} \text{ and } \lambda_c = 1
\end{align*}
\] (6.24)

Finding the fluid interface and central streamline intersection points now becomes a question of solving the stream equation (6.6) and interface equation (6.21) simultaneously. The form of the solution to this problem depends on the form of the two curves. The different alternatives are listed below for the case of flux through a vertical cell boundary:
1. Horizontal interface, $\lambda_x = 0$. If $\chi_b \lambda_c + \chi_x \lambda_y \neq 0$ then

$$x_{\text{intersection}} = \frac{\Psi_c \lambda_y - \chi_x \lambda_c}{\chi_b \lambda_c + \chi_x \lambda_y},$$

(6.25)

otherwise no intersection takes place.

2. Vertical interface, $\lambda_y = 0$.

$$x_{\text{intersection}} = \frac{\lambda_c}{\lambda_x}.$$

(6.26)

3. Streamline is a straight line, $\chi_b = 0$. If $\lambda_x \chi_y - \lambda_y \chi_x \neq 0$ then

$$x_{\text{intersection}} = -\frac{\Psi_c \lambda_y - \chi_y \lambda_c}{\lambda_x \chi_y - \lambda_y \chi_x},$$

(6.27)

otherwise no intersection takes place.

4. Streamline has curvature and the interface has a finite, non-zero gradient. Define

$$b = \frac{\chi_y \lambda_x - \chi_x \lambda_y - \chi_b \lambda_c}{\chi_b \lambda_x}$$

(6.28)

and

$$c = \frac{\Psi_c \lambda_y - \chi_y \lambda_c}{\chi_b \lambda_x}.$$  

(6.29)

If the determinant of the solution to these equations, $\det = \sqrt{b^2 - 4c}$, is real, then at least one intersection point occurs. If $\det = 0$, then only one intersection point occurs, namely

$$x_{\text{intersection}} = -\frac{b}{2}.$$  

(6.30)

If the determinant is real and non-zero, then two intersection points occur, namely

$$x_{\text{intersection}} = -\frac{b + \sqrt{\det}}{2} \quad \text{and} \quad x_{\text{intersection}} = -\frac{b - \sqrt{\det}}{2}. $$

(6.31)

Similar solutions are used for intersections occurring in cells with horizontal flux boundaries.
Determining Whether a Region is Void or Fluid

The integration algorithm requires knowledge of which regions along the central channel streamline are fluid or void. Referring to Figure 6.3, we first define an angle between a point on the streamline, \((x, y)\), and the left \((x_l, y_l)\) point of the relevant free surface interface,

\[
\theta_l = \tan^{-1}\left(\frac{y_l - y}{x_l - x}\right) + s\pi,
\]

where

\[
s = \begin{cases} 
\text{an even integer if } x_l - x > 0 \\
\text{an odd integer otherwise.}
\end{cases}
\]

Here \(s\) is chosen to give \(-\pi < \theta_l \leq \pi\). An angle for the right point, \((x_r, y_r)\), is similarly defined, but this time the integer multiplier is chosen to give \(\theta_l - \pi < \theta_r \leq \theta_l + \pi\).

Comparing \(\theta_l\) and \(\theta_r\) determines whether the point lies in a fluid or void region. If \(\theta_r > \theta_l\), then the point lies in fluid, otherwise the point lies in void.

Time Taken to Traverse a Fluid Region

Once the fluid region has been defined in terms of intersection points on the characteristic channel streamline, \(t_{\text{fluid}}\) can be calculated. For example, say points \(x_1\) and \(x_2\) define the extremes of the fluid region within a particular streamline in a particular cell. The length of time a fluid particle takes to cross this region is given by

\[
t_{\text{fluid}} = \left| \int_{x_2}^{x_1} \frac{1}{u(x, y)} \, dx \right| = \begin{cases} 
\frac{1}{\chi_y} (x_1 - x_2) & \text{if } \chi_b = 0 \\
\frac{1}{\chi_b} \ln \left[ \frac{\chi_b x_2 + \chi_y}{\chi_b x_1 + \chi_y} \right] & \text{if } \chi_b \neq 0,
\end{cases}
\]

where we have assumed the particle exits the cell through a vertical boundary.

136
Figure 6.3: Angles are calculated to the left and right points of the free surface interface in order to determine whether the point lies in a fluid or void region.

6.4 Algorithm Accuracy

6.4.1 Fluid Volume Conservation

The accuracy of the Stream VOF convection algorithm in conserving volume is dependent on the accuracy of the integration algorithm used to calculate boundary fluid fluxes. As previously mentioned, the accuracy of the integration algorithm is dependent on the user specified variable, $n_{\text{channel}}$, the number of channels used in calculating each boundary flux. The greater the number of channels, the greater the accuracy.

Figure 6.4 shows an example of a ‘worst case scenario’ for integration accuracy. Here we see that fluid velocity streamlines are aligned with the free surface orientation, and the magnitude of error that we could expect
from a single boundary flux calculation could be as high as

$$\text{Error } (F) = O \left( \frac{u_{i,j}}{2n_{\text{channel}} \delta x_i} \delta t \right),$$

(6.35)

where the notation $O(z)$ specifies ‘of the order $z$’.

As detailed by equations (6.16) and (6.17), after each boundary fluid flux is calculated, fluid is added to the accepting cell, and the same amount is subtracted from the donating cell. As a result, fluid volume is rigorously conserved after advection. The difficulty is that integration inaccuracies can cause undershoot and overshoot $F$ values, which when brought back into the zero to one range, can cause net changes in fluid volume.

The solution is to correct $F$ value overshoots and undershoots using a local redistribution algorithm, after all boundary fluxes have been incremented, but before the $0 \leq F \leq 1$ check. The specific redistribution algorithm can be summarized by the schematic of Figure 6.5.
Figure 6.5: Schematic showing the algorithm used to locally redistribute non-real \( F \) values.

### 6.4.2 Fluid and Void Wisp Generation

Integration inaccuracies can also cause ‘wisp’ generation of either fluid in void regions, or alternatively void in fluid regions. Wisps are generated when a free surface interface moves across the computational domain. If, for example, a fluid interface is sweeping across a void region, integration inaccuracies at the interface cause small amounts of void to remain in the cells after the interface has passed, thus creating wisps of void within the fluid region. Similarly, wisps of fluid in void regions can be created when a void interface sweeps over an area of fluid.

Obviously the level of wisp generation depends on the accuracy of the integration scheme, and thus the magnitude of the user set variable \( n_{\text{channel}} \).
In general, the amount of fluid or void involved in wisps is small, even for small values of $n_{\text{channel}}$, but their presence is detrimental to computational efficiency, as they require the discretised Navier-Stokes equations to be solved in cells that should not contain fluid.

In the BOUNCE Stream method, wisps are eradicated using an algorithm similar to the undershoot and overshoot algorithm detailed above. For the case of a fluid wisp in a void region, a cell is considered to contain a wisp if the cell, and all of its four neighbours, contains less than a certain proportion of fluid. This critical proportion is specified by the user set variable, $F_{\text{wisp}}$. Once a fluid wisp is established, fluid is moved from the wisp cell to a neighbouring cell which is closer to the free surface interface. Exactly which cell is closer to the free surface interface is specified by the direction of the free surface interface normal in the wisp cell.

For the case of a void wisp in a fluid region, a cell is considered to contain a wisp if the cell and all four neighbours have VOF values greater than $1 - F_{\text{wisp}}$. As for the fluid wisp case, once a void wisp is established, fluid is moved from cells closer to the free surface interface to the examined cell, thus removing the void in fluid wisp.

Equation (6.35) gives the maximum expected error per boundary flux calculation. In calculating a suitable level for $F_{\text{wisp}}$, we note that;

1. each cell could be involved in up to four boundary flux calculations per timestep, corresponding to four cell boundaries, and

2. in real fluid calculations the computational timestep is limited by a stability criterion imposed on the solution to the discretised Navier-Stokes equations. This stability criterion can be expressed as the maximum distance a fluid particle can travel in one timestep is half of a cell dimension.
Considering these two points, a suitable level for the critical wisp VOF value is

\[ F_{\text{wisp}} = \frac{1}{n_{\text{channel}}}. \]  

(6.36)

Experimentation has shown that this level is sufficient in suppressing all wisps.

By destroying a wisp once it is generated, rather than using various criteria to limit fluid boundary fluxes, the Stream method wisp eradication technique has negligible effect on fluid geometry, and is capable of removing both wisps of fluid in void as well as wisps of void in fluid.

### 6.5 Numerical Procedure

Figure 6.7 shows the numerical procedure used to advect the VOF function using the Stream method. The subroutine \textit{VFSTREAM} contains three main loops. In the first loop, the velocity variables \( \chi_b, \chi_x \) and \( \chi_y \) are set for all cells possessing a conservative velocity field. The second loop calculates the free surface interface variables \( \lambda_c, \lambda_x \) and \( \lambda_y \) for all cells containing a free surface. The third and main loop, loops through all boundaries within the computational domain, calculating and incrementing fluid boundary fluxes. This is accomplished by looping along each cell boundary, calculating the fluid flux from each individual boundary channel. Each boundary channel calculation is performed by moving back along each characteristic streamline until the starting boundary of the channel is found. At the same time, \( t_{\text{fluid}} \) is calculated for each cell that is traversed.

The final sections of subroutine \textit{VFSTREAM} implement the undershoot and overshoot algorithm, the wisp eradicator algorithm, and finally, adjust the VOF values to mark fluid, void and free surface cells. The method used
SUBROUTINE VFSTREAM

Loop through all cells

Does cell contain fluid or is near a free surface interface?
  Yes
  Check that velocity field is conservative
  Yes
  Calculate velocity cell variables \( x_{\chi}, x_{\lambda}, \) and \( x_{\rho} \).
  No
  Loop through all real cells

Call subroutine LEFT_RIGHT to calculate the left and right points for the free surface interface

Calculate the interface variables \( \lambda_{\chi}, \lambda_{\rho}, \) and \( \lambda_{\lambda} \).

Loop through all boundaries within computational region

Determine which cell is a donating cell and which is an accepting cell over the x direction boundary

Loop through all channels along cell boundary

Determine starting point of channel on cell boundary

Use function TFLUID to calculate \( f_{\text{fluid}} \)
  for each cell we pass through and the time remaining in the integration, \( t_{\text{rem}} \).
  Yes
  \( t_{\text{rem}} > t_0 \)
  No
  Increment boundary fluxes to VOF values in the accepting and donating cells

Repeat VOF calculation for the y direction boundary

Implement undershoot/overshoot correction algorithm

Implement whispering gallery algorithm

Loop through all real cells

Reset new VOF value of cell to zero if less than \( \text{emf1} \)
Reset new VOF value of cell to one if greater than \( \text{emf1} \)

Return to main program

Figure 6.6: Schematic of subroutine VFSTREAM, used to advect the VOF function using the Stream method.
here follows that used in both the Hirt-Nichols and Defined Donating Region algorithms (see Sections 5.2.4 and 5.3.4, respectively).

Function *TFLUID* is used by subroutine *VFSTREAM* to calculate \( t_{\text{fluid}} \) for each cell traversed by each characteristic streamline. A schematic of function *TFLUID* is given in Figure 6.7 — more details can be found in the code listing given in Appendix E.

### 6.6 Translation Test - The Box Problem

Figure 6.8 shows the test box translation problem, as described in Section 5.2.1, now calculated using the Stream convection algorithm. In this example, the error minimisation technique was used to calculate free surface orientations, and the number of channels used per boundary flux was \( n_{\text{channel}} = 10 \). Also, the wisp eradication routine was implemented with a critical VOF wisp level of \( F_{\text{wisp}} = 0.1 \), in line with equation (6.36).

Comparing Figures 5.20 and 6.8 shows that less staggered velocity induced diffusion is occurring under the Stream algorithm method than the Defined Donating Region method. This is evidenced by the fact that the box translated by the Stream method has not stretched in a direction normal to the velocity field as the box translated by the Defined Donating Region method did. The Stream algorithm box still displays significant rounding of the free surface at box corners. As previously discussed however, this is a feature of the finite grid size and interface reconstruction method used in calculating boundary fluid fluxes.

Figure 6.9 shows the test box problem repeated using the Stream algorithm, but this time with the greater accuracy level of \( n_{\text{channel}} = 100 \), and the corresponding wisp eradication level of \( F_{\text{wisp}} = 0.01 \). The differences between
Figure 6.7: Schematic showing function TFLUID. This function is used by subroutine VFSTREAM to calculate $t_{\text{fluid}}$ for each cell when integrating boundary fluxes.
Figure 6.8: Test case calculated using the BOUNCE Stream algorithm. Three VOF contours shown; $\varepsilon_F$, 0.5 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. Here $n_{\text{channel}} = 10$ and $F_{\text{wisp}} = 0.1$.

the results of Figures 6.8 and 6.9 are minimal, although the width of the free surface transition zone appears slightly smaller in the latter case.

In both the examples shown, fluid volume was rigorously conserved under the Stream convection algorithm.

### 6.7 Cylindrical Coordinates

The Stream VOF advection algorithm has not been applied to cylindrical coordinates in this study. While the Stream method technique is equally applicable to Cartesian or cylindrical coordinates, implementation in cylindrical
Figure 6.9: Test case calculated using the BOUNCE Stream algorithm. Three VOF contours shown; $\varepsilon_F$, 0.5 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. Here $n_{\text{channel}} = 100$ and $F_{\text{wisp}} = 0.01$.

Coordinates would involve significant changes to the interface reconstruction method, chosen continuous velocity field and flux integration technique.
Chapter 7

VOF Advection Tests — A Comparison of VOF Advection Algorithm Accuracy

7.1 The Zalesak Slotted Disk Test

The Zalesak test has become a benchmark test for comparison of VOF advection algorithms. First devised by Zalesak [1979], the test involves rotating a slotted disk through one complete revolution within the computational domain. Advection algorithm accuracy can be quickly gauged by comparing the initial and final positions of the disk.

The form of the test used here is that originally presented [Zalesak 1979]. The computational domain has dimensions $1 \times 1$ m$^2$ and is composed of $100 \times 100$ equally sized square cells. The uniform vorticity velocity field has components

\[ u = -\Omega (y - y_0) \]  \[ v = \Omega (x - x_0) \]

where the constant angular velocity is chosen as $\Omega = 2\pi$ rad/s so that one
complete revolution of the disk takes exactly 1 s. The velocity field is centred at the centre of the computational domain, thus \( x_0 = 0.5 \) and \( y_0 = 0.5 \). The computational timestep is chosen so that one revolution takes 628 steps. The disk has a radius of 0.15 m, is centred at \((0.5, 0.75)\), and contains a slot of width 0.05 m. The length of the slot is such that the maximum distance between the end of the slot and the top of the cylinder is 0.05 m.

Figure 7.1 shows the Zalesak test problem computed using a variety of advection algorithms. In all the six cases shown, the plots were generated by plotting the \( F = 0.5 \) contour line — the dashed line indicates the initial disk position, and the solid line the disk position after one revolution.

Case (A) of Figure 7.1 shows results for the Hirt-Nichols algorithm. The correlation between the initial and final positions is poor, with both disk geometry and location varying significantly between the two times. Although not indicated in the plot, there was also significant generation of ‘wisps’ by the Hirt-Nichols algorithm.

Case (B) of Figure 7.1 shows results computed using the Defined Donating Region method and RIPPLE method for interface gradient calculation. Both the disk shape and location are predicted reasonably well after one revolution, although the disk slot has extended by approximately 0.02 m, there has been some stretching of the disk in the radial direction and the sharp corners around the slot have been rounded somewhat. While the extension of the slot and stretching of the disk are not of alarming magnitudes, both are caused by the staggered velocity dispersion, a previously described feature of the Defined Donating Region algorithm. Rounding of the slot corners is caused by the finite cell size and interface reconstruction method used to represent the free surface of the fluid. Again, this rounding is not of an alarming magnitude. There was no wisp generation in this calculation.
Figure 7.1: Zalesak slotted disk rotation test performed with the H-N, DDR and Stream VOF advection algorithms. Case (A) is the H-N algorithm, Case (B) the DDR algorithm using the RIPPLE method for free surface gradient calculation, Case (C) the DDR algorithm using the error minimisation method for free surface gradient calculation, Case (D) the Stream method using the RIPPLE method for free surface calculation and $n_{\text{channel}} = 10$, Case (E) the Stream method using the error minimisation method for free surface calculation and $n_{\text{channel}} = 10$ and Case (F) the Stream method using the error minimisation method for free surface calculation and $n_{\text{channel}} = 100$. 
Case (C) shows a repeat of case (B), but this time computed using the error minimisation method for interface gradient calculation. Any improvements made by the more complex interface reconstruction method are of marginal significance in this particular advection test.

Cases (D), (E) and (F) of Figure 7.1 were all calculated using the Stream advection algorithm. Case (D) was calculated using $n_{\text{channel}} = 10$ and the RIPPLE interface gradient calculation, Case (E) using $n_{\text{channel}} = 10$ and the error minimisation interface gradient calculation, and Case (F) using the higher number of integration channels $n_{\text{channel}} = 100$, again coupled to the error minimisation interface gradient calculation. All the results calculated using the Stream scheme appear similar.

Overall, the performance of the Stream scheme in the Zalesak advection test is superior to that of the other examined advection algorithms. Specifically, under the Stream scheme the slot in the disk has not extended significantly, and the perimeter of the disk after one revolution returns almost precisely to the starting location of the disk. Some rounding of the sharp corners of the fluid geometry has occurred, a feature of the surface reconstruction method discussed previously. Again, there was no wisp generation in this calculation.

Comparing results achieved using the different variations of the Stream scheme in Figure 7.1, we see that either the added complexity of the error minimisation method or the added computational expense required by the greater accuracy $n_{\text{channel}} = 100$ test do little to improve advection performance. In all the Stream algorithm cases, the wisp eradicator was successful in eliminating all wisps.

Table 7.1 shows the computational times required to perform the Zalesak rotation tests. The computer used for these tests was a Pentium-Pro PC run-
Table 7.1: Calculation times for the Zalesak slotted disk problem.

<table>
<thead>
<tr>
<th>VOF advection Method</th>
<th>Calc. Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A) Hirt-Nichols</td>
<td>54</td>
</tr>
<tr>
<td>(B) Defined Donating Region, RIPPLE interface gradient</td>
<td>59</td>
</tr>
<tr>
<td>(C) Defined Donating Region, error minimisation gradient</td>
<td>231</td>
</tr>
<tr>
<td>(D) Stream, $n_{\text{channel}} = 10$, RIPPLE interface gradient</td>
<td>124</td>
</tr>
<tr>
<td>(E) Stream, $n_{\text{channel}} = 10$, error minimisation gradient</td>
<td>344</td>
</tr>
<tr>
<td>(F) Stream, $n_{\text{channel}} = 100$, error minimisation gradient</td>
<td>826</td>
</tr>
</tbody>
</table>

The calculation times for the Zalesak slotted disk problem were measured at 200 MHz. As shown, the fastest method was the H-N algorithm (A), followed closely by the DDR method using the RIPPLE interface gradient technique (B), and then the Stream method using $n_{\text{channel}} = 10$ and again the RIPPLE interface gradient technique (D). Out of these three cases, the H-N and DDR algorithms took approximately the same computational time, but as discussed above, results achieved using the H-N algorithm were not acceptable. The Stream scheme of case (D) took approximately twice the computational time of the case (B) DDR algorithm, but as also discussed above, provided a greater level of accuracy.

All tests run using the error minimisation gradient method in Table 7.1 required substantially longer to complete than equivalent tests run using the RIPPLE interface gradient technique. The error minimisation gradient algorithm is very expensive because it is iterative, and because for each iteration, free surface constructions have to be made in a total of nine cells. In the Zalesak problem its use is probably not justified. Similarly, the Stream scheme run with a greater accuracy criteria of $n_{\text{channel}} = 100$ took over twice as long as the calculation performed with $n_{\text{channel}} = 10$. Again, in the Zalesak problem, this greater accuracy criteria is probably not justified.
While the coding of both the Defined Donating Region and Stream VOF algorithms could be further optimised, in practice the computational expense of solving the discretised Navier-Stokes equations using large scale matrix inversions far outweighs time spent in calculating VOF fluxes, so the use of these more complex VOF advection algorithms is well justified. Also, the added accuracy that these schemes deliver often means that larger timesteps can be taken, thus reducing the frequency between the computationally expensive matrix inversions, further increasing code efficiency.

7.2 Non-Uniform Vorticity Tests

More thorough VOF advection algorithm tests are those where free surface interfaces are sheared and deformed in non-uniform vorticity velocity fields. Two such tests, performed here, are taken from the work of Rider & Kothe [1998].

7.2.1 Single Vortex

In the first test, a cylinder of fluid, of radius 0.15 m and centred at (0.5,0.75), is deformed in a velocity field specified by the stream function,

\[ \Psi = \frac{1}{\pi} \sin^2 (\pi x) \sin^2 (\pi y). \]  

(7.2)

The velocities are defined normally as,

\[ u = -\frac{\partial \Psi}{\partial y} \text{ and } v = \frac{\partial \Psi}{\partial x}. \]  

(7.3)

To visualize this flow field, we have inserted marker particles in the flow, originating from inside the cylinder volume. Approximately 45,000 evenly
spaced particles were used to create each trace, their paths explicitly integrated using a timestep of 1 ms and the exact velocity field. Similar marker particle traces were presented in Rider & Kothe [1998].

Figure 7.2 shows the effect of the single vortex velocity field on the marker particles. As shown, the fluid form is stretched and sheared as it is rotated around itself, forming a spiral type pattern.

Figure 7.3 shows the single vortex problem calculated using the Defined Donating Region method and the RIPPLE method for free surface gradient estimation. The grid was composed of 32×32 square cells, and the computational timestep used was 10 ms. Comparing Figures 7.3 with 7.2, we see that the VOF advection algorithm reproduces the deforming fluid well, until approximately 1.5 s, when breakup of the ‘tail’ of the spiral starts to occur. Beyond this time, the breakup of the fluid becomes more and more extreme, so that by 3 s, the spiral has transformed into a series of discrete fluid ‘globs’.

Breakup of the spiral occurs because the width of the fluid form becomes less than the computational cell dimension. In these cases, the interface reconstruction technique tends to arrange the small amounts of fluid in each cell as close together as possible, causing the fluid ‘glob’. This process can be described as numerical surface tension. As shown, its effect is most pronounced when the dimensions of the fluid region are similar to or smaller than the dimensions of the computational cells.

In Figures 7.4 and 7.5 we see the same single vortex test repeated with the Defined Donating Region algorithm and the Stream algorithm, respectively. In both cases, the error minimisation technique was used to calculate free surface orientations. Comparing Figures 7.4 and 7.3, we see that using the error minimisation technique has resulted in slightly less numerical surface tension, and consequently, a slightly better prediction of fluid location. From
**Figure 7.2:** Single vortex flow field deforming a cylinder. Marker particle positions shown at indicated times approximate exact solution.
Figure 7.3: Single vortex VOF advection test calculated with the Defined Donating Region scheme using the Ripple method for free surface gradient calculation. Five contour levels shown; $\varepsilon_F$, 0.25, 0.5, 0.75 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. 
Figure 7.4: Single vortex VOF advection test calculated with the Defined Donating Region scheme using the error minimisation method for free surface gradient calculation. Five contour levels shown; $\varepsilon_F$, 0.25, 0.5, 0.75 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. 
Figure 7.5: Single vortex VOF advection test calculated with the Stream scheme using the error minimisation method for free surface gradient calculation. Here $n_{\text{channel}} = 100$, and $F_{\text{wisp}} = 0.01$. Five contour levels shown: $\varepsilon_F$, 0.25, 0.5, 0.75 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. 

157
Figure 7.5, we see that there is further improvement in using the Stream method, but the level of improvement is not pronounced.

An attractive method for testing VOF advection algorithm performance is given by time reversing the velocity field, so that the fluid returns to its starting position after a set period [Rider & Kothe 1998]. In this manner, the initial and final positions of the fluid can be compared directly, revealing the level of VOF advection error present. To accomplish the time reversal, we use a modified stream function [Rider & Kothe 1998] to define a new fluid velocity field,

\[ \Psi = \frac{1}{\pi} \sin^2 (\pi x) \sin^2 (\pi y) \cos \left( \frac{\pi t}{T} \right), \tag{7.4} \]

where \( T \) is the time required for the fluid to advance from and return to its starting position.

Figure 7.6 shows the time reversed single vortex advection test performed using the Defined Donating Region and Stream schemes. In both cases the error minimisation method was used to calculate free surface orientations, the grid was composed of 32×32 cells and the computational timestep used was 10 ms. The Stream method was calculated using \( n_{\text{channel}} = 100 \) and \( F_{\text{wisp}} = 0.01 \). For the tests with \( T = 0.5 \text{ s} \) and \( T = 2.0 \text{ s} \), the advection algorithms perform well, returning the fluid close to its original position and form. For the longer tests with \( T = 8.0 \text{ s} \), the performance is less satisfactory, with the Stream method showing marginally better accuracy than the Defined Donating Region method. The difficulty with the longer test is that the fluid spiral becomes quite thin at \( t = \frac{T}{2} \), and as seen in the previous non-time reversed tests, this causes the spiral to degenerate into discrete ‘globs’ of fluid.
Figure 7.6: Time reversed single vortex VOF advection test. Cases (A)–(C) are calculated using the Defined Donating Region method, while cases (D)–(F) are calculated using the Stream method. The initial fluid position is indicated by the dashed contour line, and the final ($t = T$) position by the dark contour line. Cases (A) and (D) use $T = 0.5s$, cases (B) and (E) use $T = 2.0s$, and cases (C) and (F) use $T = 8.0s$.  

159
7.2.2 A Deformation Field Test

The second non-uniform vorticity test performed here, also devised by Rider & Kothe [1998], again involves deforming a cylinder of fluid, the dimensions and position of which are unchanged from the single vortex test. The velocity field in this test is more complex however, and is described by the stream function,

\[ \Psi = \frac{1}{4\pi} \sin \left(4\pi \left( x + \frac{1}{2} \right) \right) \cos \left(4\pi \left( y + \frac{1}{2} \right) \right). \tag{7.5} \]

We have again used marker particles to visualize this flow field, as shown in Figure 7.7.

Figures 7.8 and 7.9 show the deformation field test calculated using the Defined Donating Region and Stream methods, respectively. Both utilised the error minimisation gradient method, a computational timestep of 10 ms and grid dimensions of 32×32 cells. The Stream method used \( n_{\text{channel}} = 100 \) and \( F_{\text{wisp}} = 0.01 \).

Comparing the VOF solutions to the marker particle solution of Figure 7.7 shows that the VOF solution is quite accurate at \( t = 0.5 \) s, but beyond this, the resemblance between the VOF solutions and ‘exact’ solution deteriorates. This is hardly surprising however, as the computational cell dimensions used are quite large relative to the scale of the observed feature. Interestingly, the Stream method computes the small amount of fluid existing between the two major vortices at \( t = 0.5 \) s, whereas the Defined Donating Region method does not.

It is arguable that the two vortex tests we have performed test the free surface interface reconstruction technique rather than the VOF cell boundary flux algorithm. This is certainly true for smaller timesteps. In all of these non-uniform vorticity tests, the computational timestep used was 10 ms. More accurate results could have been achieved using a smaller computational
Figure 7.7: Deformation field deforming a cylinder. Marker particle positions shown at indicated times approximate exact solution.
Figure 7.8: Deformation field VOF advection test calculated with the Defined Donating Region scheme. The six VOF contour shades shown are separated by the five contour levels; $\varepsilon_F$, 0.25, 0.5, 0.75 and $1 - \varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$. 

162
Figure 7.9: Deformation field VOF advection test calculated with the Stream scheme. The six VOF contour shades shown are separated by the five contour levels: \( \varepsilon_F, 0.25, 0.5, 0.75 \) and \( 1 - \varepsilon_F \), where \( \varepsilon_F = 1 \times 10^{-6} \).
timestep, but at the expense of blurring any differences between the Defined Donating Region and Stream VOF advection algorithms.

7.3 Other VOF advection Algorithms

7.3.1 Youngs Algorithm

The Youngs VOF algorithm [Youngs 1982] was one of the first to reconstruct the free surface interface in each cell with an infinitely variable gradient. This scheme is popular and delivers better accuracy than the Hirt-Nichols algorithm. The difficulty with Youngs algorithm is that it is an operator split method, meaning that fluxes are first calculated and incremented in one coordinate direction, the free surfaces are reconstructed, and the fluxes then calculated and incremented in the other coordinate direction. The disadvantages of this technique lie in computational expense — fluid free surfaces have to be constructed twice per computational timestep — and that fluxes are not calculated symmetrically.

A comparison of the Youngs VOF algorithm against a comparable unsplit VOF algorithm is given in [Rider & Kothe 1998].

7.3.2 Multidimensional VOF Advection Schemes

The Defined Donating Region and Stream schemes developed in this study are two of the four currently available fully multidimensional VOF advection algorithms. The existing two algorithms, as outlined in Chapter 5, Section 5.1, are those developed by [Rider & Kothe 1998] and Puckett et al. [1997]. In this section we present a brief comparison of the four schemes, in order to determine the strengths and weaknesses of each one.
The four multidimensional VOF advection algorithms under discussion all calculate cell boundary fluid fluxes using a conceptually similar method. Fluid interfaces are first reconstructed in each computational cell, so that the location of fluid regions is determined. Donating regions are then defined for each cell boundary, and finally, fluid fluxes over each boundary are calculated as the intersection between these donating regions and the fluid regions. The primary differences between the four schemes lie in the method used for defining cell boundary donating regions. It is these primary differences which characterise the operation and performance of each of the multidimensional VOF advection algorithms.

Figure 7.10 shows example donating regions for a right cell boundary, as calculated using each of the four schemes. In all cases the horizontal component of the velocity at the boundary is assumed to be directed towards the right, and the local velocity field may contain non-uniform vorticity.

Case (A) of Figure 7.10 shows a donating region calculated using the Defined Donating Region scheme. The fluid crossing the boundary can only originate from the one adjacent cell. This property ensures that the Defined Donating Region scheme conserves fluid volume without requiring any fluid redistribution algorithms, and produces no fluid wisps or other debris.

Cases (B) and (C) of Figure 7.10 represent donating regions defined using the Rider & Kothe [1998] and Puckett et al. [1997] schemes, respectively. As shown, a single donating region defined under these schemes can traverse one, two or even three cells. This may produce a greater level of accuracy than the Defined Donating Region scheme, however, this is at the expense of fluid conservation. As the donating regions are defined independently for each cell boundary, when the velocity field is spatially varying, donating regions can overlap, or not include some fluid regions [Rider & Kothe 1998]. Also, under
these conditions donating regions may not contain the total volume fluxed through the associated boundary. Consequently, both the Rider-Kothe and Puckett et al. schemes require a local fluid redistribution algorithm in order to achieve fluid conservation.

The final case (D) in Figure 7.10 shows a donating region constructed by the Stream algorithm. Donating region boundaries are now defined using an effectively continuous velocity field, and may traverse as many computational cells as the computational timestep allows. A local fluid redistribution algorithm is required by the Stream scheme to ensure fluid conservation, but
it is used only to correct inaccuracies caused by the approximate boundary flux integration procedure. As the user has control over the accuracy of this integration via the number of channels used per boundary cell calculation, \( n_{\text{channel}} \), the dependence on the local redistribution algorithm is not inherent to the advection algorithm, but rather, is controlled. In the limiting case of exact boundary flux integrations, the Stream scheme would conserve volume precisely and require no fluid redistribution algorithm.

An added advantage of being able to control the precision of the integrations performed under the Stream scheme is that the user has control over the computational time spent calculating VOF advection. Thus, if fluid free surface interface behaviour is relatively unimportant in a particular calculation, \( n_{\text{channel}} \) can be set at a low value, and the simulation produced quickly.

The Defined Donating Region scheme is a simple and robust scheme suitable for most VOF fluid calculations. It has the advantage of rigorously conserving fluid volume, without any iterations or local redistribution schemes, and is capable of calculating complex geometries reasonably accurately with no wisp generation. This is a particularly attractive feature of the scheme, appreciated in fluid problems such as the impacting droplet problem, where the distinct nature of free surfaces must be maintained. The Defined Donating Region scheme has also been formulated for cylindrical coordinates, an essential option for the impacting droplet problem.

The Stream scheme is more computationally expensive than the Defined Donating Region scheme, but is slightly more accurate. Coupled with the local fluid redistribution algorithm and de-wisper algorithm, it conserves volume rigorously and maintains reasonably sharp free surface interfaces. A nice feature of the Stream scheme is that the user has control over the level of accuracy used in calculating fluid cell boundary fluxes.
The Stream scheme has only been applied to two-dimensional Cartesian coordinates in this study, however, application to other coordinate systems would be simple. Indeed, in three-dimensional Cartesian coordinates, few modifications to the existing code would be required, making the Stream scheme easier to apply than other multidimensional schemes.

The ultimate VOF advection scheme could possibly be the Stream scheme, but with two major modifications. The velocity field presently used is continuous, except at cell vertex points. A better velocity field would be one that is continuous everywhere, although this is sure to be at the expense of analytical and computational complexity. The second improvement to the Stream scheme would be to calculate cell boundary flux integrations exactly. To accomplish this, the first major improvement of a fully continuous velocity field would be required, and even then, exact integrations may be complex to compute.

Of the two free surface interface reconstruction techniques examined, the error minimisation technique is slightly more accurate, but is considerably more expensive. As previously discussed, however, its use may be justified in some problems.

An improved free surface reconstruction technique could involve curved rather than straight free surface interfaces in each cell, thus better representing real fluid geometries. As shown in the above tests, however, representing the surface as a straight line in each cell gives good results for fluid geometries that are larger than cell dimensions — it is only when fluid forms have features which are of the same size or smaller than cell dimensions that poor surface reconstruction occurs. As velocity field solutions obtained by solving the discretised Navier-Stokes equations are only available in spacings equal to the cell dimensions, a scheme involving curved free surfaces within each
cell is not justified.

7.4 Real Calculation Divergence Correction

Under actual fluid flow calculations, where the fluid velocity field is calculated using the discretised Navier-Stokes equations, total fluid volumes typically fluctuate with a small amplitude around the actual fluid volume. These fluctuations are a result of the accuracy of the PPE velocity solution, which is limited by the residual comparator $\varepsilon$. As the level at which $\varepsilon$ is set determines the convergence of the PPE solution, it also determines the accuracy to which the continuity equation is satisfied throughout the flow field, and within each cell. Thus, to achieve absolute conservation of fluid volume, $\varepsilon$ must be set to the computationally impossible value of zero.

In practice the level of fluid volume fluctuations resulting from computational solutions to the PPE are very low — for a typical value of $\varepsilon = 10^{-4}$ used in a droplet impact calculation, the fluctuation in fluid volume is of the order $10^{-3} \%$. In the RIPPLE code [Kothe et al. 1994], VOF values are adjusted after each advection calculation to correct for these errors, however, as the level of fluctuations in the studied droplet impact tests is low, this correction was not implemented in the BOUNCE code.
Chapter 8

The Viscous Vapour Layer Model — Theory and Assumptions

The viscous vapour layer refers to the region of vapour between the lower droplet surface and solid surface. The vapour layer is responsible for cushioning and possibly repelling the droplet from the solid surface. Consequently, accurate calculation of pressures generated within this layer is imperative in calculating the dynamics of droplet impacts.

Flow within the vapour layer is calculated by BOUNCE using a code separate to the main droplet dynamics VOF code. Specifically, this is because:

1. The height of the vapour layer can be several orders of magnitude smaller than the dimensions of the droplet. If the droplet and vapour layer were to utilize the same computational mesh, the computational cell size required to describe the vapour layer flow would necessitate an unworkably large number of cells to describe the movement of the droplet.
2. A droplet impacting a hot surface is a stiff problem — stiff in that it involves processes which operate on different timescales. The vapour within the layer moves rapidly through a small channel, the dimensions of which can change rapidly as the droplet nears or rebounds from the surface. Conversely, the dynamics of the body of the droplet are largely governed by surface tension forces, which deform the droplet with relatively low internal fluid velocities. As these different flow phenomena operate on radically different timescales, the problem is stiff.

A consequence of this stiffness is that droplet simulations are computationally expensive, requiring a large number of timesteps to predict a single impact. Thus, in the interests of minimising computational expense, it is imperative that the viscous vapour layer code be as efficient as possible. Using a separate code that is specific to the viscous vapour layer is more efficient than using a two phase VOF code, because simplifications can be used in the development of the separate code which are specific to the vapour layer. An example here is the assumption of one-dimensional flow within the layer.

3. Using a code which is specific to the viscous vapour layer allows certain features to be modelled, which are specific to the vapour layer. Examples here include the modelling of heat transfer into both the solid surface and liquid droplet, and the molecular treatment of both the liquid-vapour and solid-vapour interfaces. These features would not be easily incorporated into a two phase VOF code.

In this chapter we outline the theory and assumptions used in the treatment of the viscous vapour layer. The computational method used by the vapour layer code to solve the vapour layer equations is given in Chapter 9.
8.1 Problem Definition

Figure 8.1 shows the basic variables used to analyse the viscous vapour layer. The problem is defined in two dimensional cylindrical coordinates, with \( r \) representing the radial direction and \( z \) the vertical or axial direction. The droplet is separated from the solid surface by a distance \( \delta \), which is a function of \( r \). The vapour mixture, so called because it can be composed of both air and vapour resulting from droplet vaporisation, has velocity components \( u \) and \( v \) in the \( r \) and \( z \) directions, respectively.

Both in the analysis and the computational solution procedure, the vapour layer problem is divided into two sections:

1. Viscous vapour flow. Here we solve for the vapour velocities beneath the droplet, and find the resulting pressures within the vapour layer. The vapour layer pressures are responsible for repelling the droplet from the surface. Required for this solution is a knowledge of the position of the droplet, provided by the internal droplet dynamics VOF algorithm, and a knowledge of the droplet vaporisation velocity, \( \dot{w}(r) \), which is
provided by the second section of the vapour layer analysis.

2. Droplet vaporisation velocity, \( \dot{w}(r) \). The droplet vaporisation velocity is the velocity at which vapour is being produced at the lower surface of the droplet. The vaporisation velocity is determined by heat transfer within the solid, vapour mixture and liquid mediums.

In this chapter we describe the viscous vapour flow solution, followed by the droplet vaporisation velocity solution.

### 8.2 Viscous Flow Solution

#### 8.2.1 Momentum Conservation

**Navier-Stokes Equations**

In two dimensional cylindrical coordinates, the Navier-Stokes momentum equations for incompressible Newtonian flow with gravitational terms neglected reduce to

\[
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial r} + v \frac{\partial u}{\partial z} = -\frac{1}{\rho_m} \frac{\partial P}{\partial r} + \nu_m \left( \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\partial u}{\partial r} \right) - \frac{u}{r^2} + \frac{\partial^2 u}{\partial z^2} \right),
\]

(8.1)

in the radial direction, and in the vertical direction,

\[
\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial r} + v \frac{\partial v}{\partial z} = -\frac{1}{\rho_m} \frac{\partial P}{\partial z} + \nu_m \left( \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\partial v}{\partial r} \right) + \frac{\partial^2 v}{\partial z^2} \right),
\]

(8.2)

where a constant kinematic viscosity of the vapour mixture, \( \nu_m \), has been assumed. The subscript \( m \) in these equations refers to properties of the vapour mixture.

Incompressible flow within the vapour layer has been assumed because velocities generated within the layer are several orders of magnitude smaller than the speed of sound within the vapour. Also, pressures generated within
the layer in excess of atmospheric pressure are usually small compared with atmospheric pressure, further supporting the incompressible flow assumption. Vapour layer pressures calculated during a typical droplet impact are examined in Chapter 10.

In obtaining a flow solution to the Navier-Stokes equations, we note that the most significant processes, in terms of repelling the droplet from the solid surface, occur when the droplet is close to the solid surface. Under such conditions, we can employ the following three assumptions:

1. the height of the vapour layer is small compared with the radius of the droplet or vapour layer,

2. the height of the layer changes slowly in the radial direction, and

3. the height of the layer changes at a velocity lower than the rate at which the vapour moves.

Using these assumptions we find that the magnitude of the vertical vapour velocity is much lower than that of the radial velocity, and equation (8.2) simplifies to,

$$-\frac{1}{\rho_m} \frac{\partial P}{\partial z} = 0.$$  (8.3)

Thus, the pressure is constant across the thickness of the vapour layer. Also, in the radial direction we find that the magnitudes of the inertial, momentum advection and radial viscous terms are all small, and the Navier-Stokes equation (8.1) reduces to the viscous flow equation,

$$\frac{\partial^2 u}{\partial z^2} (r, z) = \mu_m \frac{dP}{dr} (r).$$  (8.4)

An order of magnitude analysis that was performed on equations (8.1) and (8.2) during an actual droplet simulation confirmed the validity of equations (8.3) and (8.4).
Vapour Velocity Boundary Conditions

To integrate equation (8.4) we require boundary conditions for the vapour velocity at the upper and lower surfaces of the viscous layer.

In this study, the Knudsen number (Kn) is defined as the ratio of the average mean free path of the vapour mixture to the height of the vapour layer. As shown in Chapter 10, this number can approach values of the order 0.1 during droplet impact simulations. As discussed in Appendix A at these levels, the continuum assumption for the gas is justified at several mean free paths away from any adjoining medium, but a kinetic theory treatment must be used to model gas behaviour in regions close to any vapour interface.

A full kinetic momentum analysis of molecule behaviour at vapour interfaces is given in Appendix A. The conclusion of this analysis is that a slip velocity exists between the vapour and each adjacent medium, and that its magnitude is given by equation (A.10).

As the solid surface below the droplet is stationary, equation (A.10) gives the velocity slip boundary condition at the solid-vapour interface to be

\[ u(r, 0) = \frac{2 - \sigma_{v,s}}{\sigma_{v,s}} \lambda_{m,s} \frac{\partial u}{\partial z}(r, 0). \tag{8.5} \]

The average mean free path of the vapour mixture, \( \lambda_{m,s} \), is evaluated at the temperature of the gas adjacent to the solid surface (\( T_{sv} \)), and the constant \( \sigma_{v,s} \) is the Specular Reflection Coefficient for the combination of the vapour mixture and solid interface surface.

At the liquid-vapour interface, equation (A.10) gives the slip boundary condition as

\[ u(r, \delta) - u_l(r) = -\frac{2 - \sigma_{v,l}}{\sigma_{v,l}} \lambda_{m,l} \frac{\partial u}{\partial z}(r, \delta), \tag{8.6} \]

where \( u_l \) is the liquid velocity on the lower surface of the droplet, and \( \delta \) is the height of the vapour layer. The variable \( \lambda_{m,l} \) is the mean free path of the
vapour mixture evaluated at the temperature of the gas adjacent to the liquid surface \((T_{lv})\), and the constant \(\sigma_{v,l}\) is the Specular Reflection Coefficient for the combination of the vapour mixture and liquid interface surface.

Defining the variables

\[
\theta_s = \frac{2 - \sigma_{v,s}}{\sigma_{v,s}} \lambda_{m,s}
\]  
(8.7)

and

\[
\theta_l = \frac{2 - \sigma_{v,l}}{\sigma_{v,l}} \lambda_{m,l}
\]  
(8.8)

equations (8.5) and (8.6) become, respectively,

\[
u (r, 0) = \theta_s \frac{\partial u}{\partial z} (r, 0),
\]  
(8.9)

and

\[
u (r, \delta) - u_l (r) = -\theta_l \frac{\partial u}{\partial z} (r, \delta).
\]  
(8.10)

Integrating the viscous flow equation (8.4) twice in the vertical direction, noting the two boundary condition equations (8.9) and (8.10), gives

\[
u (r, z) = \frac{1}{\mu_m} \frac{dP}{dr} (r) \frac{z^2}{2} + \left( \frac{z + \theta_s}{\delta + \theta_s + \theta_l} \right) \left[ u_l (r) - \frac{\delta}{\mu_m} \frac{dP}{dr} (r) \left( \frac{\theta_l + \delta}{2} \right) \right].
\]  
(8.11)

The average velocity within the viscous vapour layer is given by,

\[
\bar{u} (r) = \frac{1}{\delta} \int_0^\delta u (r, z) dz.
\]  
(8.12)

Combining the equations (8.11) and (8.12) and integrating gives a relationship between the average vapour layer velocity and the vapour layer pressure,

\[
\bar{u} (r) = \frac{1}{\mu_m} \frac{dP}{dr} (r) \left[ \frac{\delta^2}{6} - \frac{\delta (\theta_l + \delta)}{2} \left( \frac{\theta_s + \delta}{2} \right) \right] + \frac{\delta (\theta_s + \delta)}{2} u_l (r).
\]  
(8.13)
For computational convenience, it is preferable to work with volume flowrates rather than average velocities within the vapour layer. Consequently, defining \( S(r) \) to be the volume flow rate of vapour mixture within the layer,

\[
S(r) = 2\pi r \delta (r) \bar{u}(r),
\]

equation (8.13) becomes

\[
S(r) = 2\pi r \delta \left\{ \frac{1}{\mu_m} \frac{dP}{dr}(r) \left[ \frac{\delta^2}{6} - \delta \left( \frac{\theta_l + \frac{\delta}{2}}{\delta + \theta_l + \theta_s} \right) \right] + \frac{\left( \theta_s + \frac{\delta}{2} \right)}{\left( \delta + \theta_l + \theta_s \right)} u_l(r) \right\}.
\]

(8.15)

Note that if we assume that the mean free path of the vapour mixture is large compared with the height of the vapour layer, a no slip velocity boundary condition results and equation (8.15) becomes,

\[
S(r) = 2\pi r \delta \left\{ - \frac{\delta^2}{12\mu_m} \frac{dP}{dr}(r) + \frac{u_l(r)}{2} \right\}.
\]

(8.16)

Furthermore, if we assume that the velocity of the fluid on the underside of the droplet is zero, then equation (8.16) simplifies to

\[
\bar{u}(r) = - \frac{\delta^2}{12\mu_m} \frac{\partial P}{\partial r}(r).
\]

(8.17)

This equation was used by Gottfried et al. [1966] in their work on quasi-steady droplet evaporation, and Buyevich & Mankevich [1995] in their work on droplet impacts.

### 8.2.2 Mass Conservation

Continuity requirements within the vapour layer are solved by considering conservation of volume on a cylinder of radius \( r' \), contained within the layer, as shown in Figure 8.2. Note that as the vapour mixture is assumed incompressible, conservation of vapour volume implies conservation of vapour mass.
Conservation of volume on the cylinder dictates

\[ S_{in} - S_{out} = \frac{dV}{dt}, \]  

(8.18)

where \( S \) represents a volume flow rate \([\text{m}^3/\text{s}]\) and \( V \) the volume of the cylinder \([\text{m}^3]\). The volume of fluid entering this cylinder from vaporisation of liquid is given by

\[ S_{in} = \int_0^{r'} 2\pi r \dot{w}(r) dr, \]  

(8.19)

where \( \dot{w}(r) \) is the vaporisation velocity \([\text{m/s}]\) at the lower surface of the droplet. The volume of fluid leaving this cylinder is equal to the volume flow rate within the vapour layer at radius \( r = r' \),

\[ S_{out} = S(r'), \]  

(8.20)

and the volume of vapour within the cylinder is

\[ V = \int_0^{r'} 2\pi r \delta(r) dr. \]  

(8.21)
Differentiating equation (8.21) with respect to time gives,

\[
\frac{dV}{dt} = 2\pi \int_0^{r'} \delta'(r) \frac{\partial \delta(r)}{\partial t} rdr.
\]  

(8.22)

Substituting equations (8.19), (8.20) and (8.22) into (8.18) gives the volume flowrate within the vapour layer in terms of the geometry of the layer and the droplet vaporisation velocity,

\[
S(r') = 2\pi \int_0^{r'} \left( \dot{w}(r) - \frac{\partial \delta}{\partial t}(r) \right) rdr.
\]  

(8.23)

The differential term contained within the integral of equation (8.23) is responsible for calculating vapour volume flowrates resulting from changes in the vapour layer height. This term was not included in the analysis of Buyevich et al. [1986], despite the studied problem being fully transient. The term can significantly affect droplet dynamics by increasing vapour layer pressures as the droplet nears the solid surface, and decreasing vapour layer pressures as the droplet moves away.

Once vapour flowrates have been determined using equation (8.23), vapour layer pressures can be calculated by integrating the previously discussed equation (8.15), noting that the pressure at the outer extremity of the vapour layer is atmospheric. Thus, a vapour flow solution is achieved.

### 8.3 Vapour Mixture Solution

In simulating a droplet impact, BOUNCE can assume that the fluid contained within the vapour layer is composed solely of vapour vaporised from the droplet, or alternatively that the fluid is a mixture of the droplet vapour and air. If the latter assumption is used, the total vapour pressure is the addition of the partial droplet vapour pressure, \( P_v \), and the partial air pressure, \( P_a \).
Thus,

\[ P = P_v + P_a. \] (8.24)

### 8.3.1 Vapour Mixture Properties

Utilizing equation (8.24), and assuming that the gases are well mixed across the small height of the vapour layer, the volume fractions of air and droplet vapour within the layer can be expressed as, respectively,

\[ y_a(r) = \frac{P_a(r)}{P(r)} \quad \text{and} \quad y_v(r) = \frac{P_v(r)}{P(r)}. \] (8.25)

Combining equations (8.24) and (8.25) gives the volume ratio conservation,

\[ y_a(r) + y_v(r) = 1, \] (8.26)

or alternatively,

\[ y_v(r) = 1 - y_a(r). \] (8.27)

The vapour layer density is given by,

\[ \rho_m(r) = y_a(r)\rho_a + (1 - y_a(r))\rho_v, \] (8.28)

where the air density, \( \rho_a \), and droplet vapour density, \( \rho_v \), are determined at atmospheric pressure and at the average vapour layer temperature of

\[ T_v = \frac{T_{s,i} + T_{sat}(P_{atmos})}{2}. \] (8.29)

The vapour layer thermal conductivity, \( k_m \) [W/m.K], and vapour layer dynamic viscosity, \( \mu_v \) [N.s/m²] are similarly defined as,

\[ k_m(r) = y_a(r)k_a + (1 - y_a(r))k_v, \] (8.30)

and

\[ \mu_m(r) = y_a(r)\mu_a + (1 - y_a(r))\mu_v. \] (8.31)
The constant air and droplet vapour properties are also determined at atmospheric pressure and at the average vapour layer temperature $T_v$.

A harmonic mean is often used in calculating the viscosity of a mixture of fluids, rather than the arithmetic mean employed in Equation (8.31). It was found that the variation in viscosity between air and droplet vapours used in this study was low, so that Equation (8.31) would represent the viscosity of a binary mixture of gases with sufficient accuracy.

Note that if the vapour layer is assumed to contain only droplet vapour, then $y_a(r) = 0$, and all vapour layer properties reduce to constant droplet vapour properties.

### 8.3.2 Vapour Mixture Calculation

In calculating the volume fraction of air within the layer, volume conservation of air is considered over a cylinder of radius $r'$, in an analogous fashion to the total vapour volume conservation examination of Section 8.2.2.

Referring again to Figure 8.2, conservation of air volume requires

\[ S_{in,a} - S_{out,a} = \frac{dV_a}{dt}, \quad (8.32) \]

where the subscript $a$ has been used to denote air volumes only. No air enters the cylinder from the lower droplet surface, so

\[ S_{in,a} = 0. \quad (8.33) \]

The volume of air exiting the cylinder is,

\[ S_{out,a} = y_a (r') S (r'), \quad (8.34) \]

and the volume of air contained within the cylinder is,

\[ V_a = \int_0^{r'} 2\pi r \delta (r) y_a (r) \, dr. \quad (8.35) \]
Differentiating equation (8.35) with respect to time, and substituting into equation (8.32) gives the air volume fraction as,

\[ y_a(r') = -\frac{2\pi}{S(r')} \int_0^{r'} \left( y_a(r) \frac{d\delta(r)}{dt} + \delta(r) \frac{dy_a(r)}{dt} \right) r dr. \]  

(8.36)

Thus, a solution to equation (8.36) for the vapour layer air fraction requires a solution to the vapour layer volume flow rates.

### 8.4 Droplet Vaporisation Velocity

The droplet vaporisation velocity, \( \dot{w} \), is defined as the velocity at which vapour is produced at the lower surface of the droplet [m/s]. In order to calculate this velocity, heat transfer rates within the solid, vapour mixture and liquid phases must be calculated.

Figure 8.3 shows temperature variables defined for the vaporisation velocity calculation. Note that generally temperatures are dependent on \( r \) as well as \( z \) — this figure illustrates temperatures at only one radial location. The temperatures of the solid and liquid phases are represented by \( T_s(r, z) \) and \( T_l(r, z) \), respectively, and the initial temperatures of the solid and liquid phases are \( T_{s,i} \) and \( T_{l,i} \), respectively. The temperature of the solid at the solid-vapour interface is \( T_{ss}(r) \), and the temperature of the vapour at the solid-vapour interface is \( T_{sv}(r) \). Similarly, the temperature of the liquid at the liquid-vapour interface is \( T_{ll}(r) \), and the temperature of the vapour at the liquid-vapour interface is \( T_{lv}(r) \).

Also shown in Figure 8.3 are heat transfer rates [W/m²] existing at both the liquid-vapour and solid-vapour interfaces. The heat transfer rate out of the solid is \( q_s \), the heat transfer rate into the bulk of the liquid is \( q_l \), and the heat transfer rate across the vapour region is \( q_v \). These heat transfer rates are a function of \( r \).
Figure 8.3: Liquid, vapour mixture and solid temperatures at radius $r$ used in the calculation of the vaporisation velocity, $\dot{w}$.

8.4.1 Solid Phase Heat Transfer

In two dimensional cylindrical coordinates, the heat diffusion equation describing conduction within the solid is [Incropera & DeWitt 1996],

$$\frac{\partial T}{\partial t} = \alpha_s \left[ \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\partial T}{\partial r} \right) + \frac{\partial^2 T}{\partial z^2} \right],$$

(8.37)

where a constant thermal conductivity, $k_s$, for the solid has been assumed [W/m.K]. The thermal diffusivity [$m^2/s$] for the solid is $\alpha_s = \frac{k_s}{c_s \rho_s}$, where $c_s$ is the specific heat [J/kg.K] and $\rho_s$ is the density [kg/m$^3$] of the solid.

During hot surface droplet impacts, solid temperature variations in the radial direction are typically small compared with variations in the axial direction. To illustrate this point, for solids which are examined in this study, the thermal diffusivity is typically of the order $1 \times 10^{-5} \text{ m}^2/\text{s}$. For a droplet with a diameter of the order 2 mm, simulations indicate that the droplet is
in residence with the surface for a time period of the order of 5 ms. Thus the thermal diffusion length, conservatively defined as $\sqrt{\alpha t}$, is of the order 0.2 mm. This length is considerably less than the diameter of the droplet. Consequently, heat transfer within the solid is dominated by conduction in the axial rather than radial direction, and the heat diffusion equation (8.37) can be simplified to give,

$$\frac{\partial T_s}{\partial t} = \alpha_s \frac{\partial^2 T_s}{\partial z^2}. \quad (8.38)$$

In order to integrate equation (8.38) to find the temperature distribution within the solid, boundary conditions are required to describe conditions at the solid-vapour interface, and within the body of the solid.

At the solid-vapour interface, the rate of heat loss to the vapour phase $[\text{W/m}^2]$ is given by Fourier’s Law as

$$q_s(r) = -k_s \frac{\partial T_s}{\partial z} (r,0). \quad (8.39)$$

For the internal solid boundary condition, we assume that the solid has a thickness which is large compared with the thermal diffusion length, so that the solid may be approximated as a semi-infinite medium. Thus, the second boundary condition is,

$$T_s(r,-\infty) = T_{s,i}. \quad (8.40)$$

Equation (8.38), combined with equations (8.39), (8.40), and a knowledge of the heat loss rate from the solid, $q_s(r)$, provide a solution for the temperature distribution within the solid phase.

### 8.4.2 Vapour Phase Heat Transfer

In two dimensional cylindrical coordinates, the equation for conservation of thermal energy in an incompressible fluid, assuming constant thermal con-
ductivity, is given as [Bird, Stewart & Lightfoot 1960],

\[
\frac{\partial T}{\partial t} + u \frac{\partial T}{\partial r} + v \frac{\partial T}{\partial z} = \frac{k}{c_p \rho} \left[ \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\partial T}{\partial r} \right) + \frac{\partial^2 T}{\partial z^2} \right] + 2 \frac{\mu}{c_p \rho} \left[ \left( \frac{\partial u}{\partial r} \right)^2 + \left( \frac{u}{r} \right)^2 + \left( \frac{\partial v}{\partial z} \right)^2 + \frac{1}{2} \left( \frac{\partial v}{\partial r} + \frac{\partial u}{\partial z} \right)^2 \right].
\]

Equation (8.41) can be greatly simplified to represent heat transfer within the vapour layer, by examining the magnitude of each of the constituent terms:

1. Using the assumptions that were made in Section 8.2.1 regarding the geometry of the vapour layer, we conclude that all terms in equation (8.41) which represent changes in the radial direction are small compared with terms representing changes in the axial direction, as the height of the vapour layer is considerably smaller than the radius of the vapour layer.

2. Comparing the convection and conduction terms in equation (8.41), the non-dimensional ratio \( \frac{v \delta}{\alpha_m} \) is small, and thus, the axial convective term in equation (8.41) can be neglected.

3. Similarly, comparing the conductive and viscous generation terms in equation (8.41), we note that the thermal conductivity, \( k \), for gases of interest in this study is several orders of magnitude larger than the absolute viscosity, \( \mu \), for those same gases. Thus, the non-dimensional ratio \( \frac{2 \mu u^2}{kT} \) is small, and consequently, the viscous dissipation terms in equation (8.41) may be neglected.

4. Finally, droplet and solid surface temperature variations in time are small compared with temperature variations across the vapour layer in
the axial direction, and consequently, the temporal term in equation (8.41) can be neglected.

Utilizing these assumptions, equation (8.41) reduces to the one dimensional steady-state conduction equation,

\[
\frac{\partial^2 T_v}{\partial z^2} = 0. \quad (8.42)
\]

Integrating twice in the axial direction, noting the solid surface and liquid surface boundary conditions

\[
T_v(r, 0) = T_{sv}(r), \quad (8.43)
\]

and

\[
T_v(r, \delta) = T_{lv}(r), \quad (8.44)
\]

respectively, gives the temperature distribution within the vapour layer as

\[
T_v = T_{sv} + \frac{z}{\delta} (T_{lv} - T_{sv}). \quad (8.45)
\]

As a linear temperature distribution has been obtained, the heat transfer rate across the layer is constant, and given by

\[
q_v(r) = \frac{k_m}{\delta} (T_{sv}(r) - T_{lv}(r)), \quad (8.46)
\]

where \(k_m\) is the thermal conductivity of the vapour mixture.

Note that heat transfer by radiation across the vapour layer has been neglected in this study. The maximum possible radiative heat loss rate by the solid surface is given by the heat loss rate from a black-body at the temperature of the surface [Incropera & DeWitt 1996]. This heat transfer rate [W/m²] is given by

\[
q_{rad} = \sigma_{\text{S.B.}} T^4, \quad (8.47)
\]
where $\sigma_{S.B.} = 5.670 \times 10^{-8}$ W/m$^2$K$^4$ is the Stefan-Boltzmann constant. For radiative heat transfer across the vapour layer to be important in the droplet impact simulation, the radiative heat transfer rate given by equation (8.47) must be of the same order of magnitude as the conductive heat transfer rate given by Equation (8.46).

Considering that the thermal conductivity of gases within the vapour layer is of the order 0.04 W/mK, and that the average vapour layer height during an impact is generally less than 0.1 mm, radiative heat transfer can be neglected below an initial solid surface temperature of approximately 700°C. Thus, 700°C represents the upper initial solid temperature under which the impact simulation results of BOUNCE are valid.

### 8.4.3 Liquid Phase Heat Transfer

Heat transfer within the liquid phase is described by the thermal energy transport equation (B.6), as developed in Appendix B for incompressible fluids represented using a VOF function in two dimensional cylindrical coordinates. This equation applied to the liquid phase yields,

$$F \frac{\partial T_l}{\partial t} + uF \frac{\partial T_l}{\partial r} + vF \frac{\partial T_l}{\partial z} = \alpha_l \left[ \frac{1}{r} \frac{\partial}{\partial r} \left( rF \frac{\partial T_l}{\partial r} \right) + \frac{\partial}{\partial z} \left( F \frac{\partial T_l}{\partial z} \right) \right], \quad (8.48)$$

where $F$ is the local volume of fluid fraction. Note that for liquids commonly involved in hot surface droplet impacts, thermal energy generation by viscous dissipation is negligible, as the dimensionless parameter $\frac{2\mu u^2}{kT}$ is small.

As in the solid conduction analysis of Section [8.4.1](#) we evaluate the characteristic thermal diffusion time, $\sqrt{\alpha t}$, for a typical droplet liquid. The thermal diffusivity of liquids used in droplet impacts is typically of the order $2 \times 10^{-7}$ m$^2$/s. For an approximate impact time of 5 ms for a 2 mm diameter droplet, this gives a thermal diffusion length of the order 0.03 mm. As this
diffusion length is considerably smaller than the dimensions of the droplet, we can assume that the effect of conduction in the radial direction is negligible compared with conduction in the axial direction, and equation (8.48) can be simplified to give

\[
F \frac{\partial T_l}{\partial t} + u F \frac{\partial T_l}{\partial r} + v F \frac{\partial T_l}{\partial z} = \alpha_l \frac{\partial}{\partial z} \left( F \frac{\partial T_l}{\partial z} \right). \tag{8.49}
\]

Equation (8.49) provides a solution for the temperature distribution within the droplet in terms of absolute coordinates — coordinates that are relative to the solid surface. For the computational solution, it is more convenient to work with a combination of absolute coordinates, and coordinates which are relative to the lower surface of the droplet.

Defining a relative coordinate system using dashed variables,

\[
t' = t, \quad r' = r \quad \text{and} \quad z' = z - \delta(t,r), \tag{8.50}
\]

the temporal and axial differential operators from equation (8.49) can be transformed to,

\[
\frac{\partial T_l}{\partial t'} = \frac{\partial T_l}{\partial t} - \frac{\partial \delta}{\partial z'} \frac{\partial T_l}{\partial z'}, \tag{8.51}
\]

and

\[
\frac{\partial T_l}{\partial z'} = \frac{\partial T_l}{\partial z}, \tag{8.52}
\]

respectively. The radial differential operator is not redefined in the relative coordinate system for reasons of computational simplicity. The axial fluid velocity in the relative coordinate system can be defined from equation (8.50) as

\[
v = v' + \frac{\partial \delta}{\partial t}. \tag{8.53}
\]

Substituting equations (8.51)–(8.53) into (8.49) yields

\[
F \left( \frac{\partial T_l}{\partial t'} - \frac{\partial \delta}{\partial t'} \frac{\partial T_l}{\partial z'} \right) + u F \frac{\partial T_l}{\partial r'} + \left( v' + \frac{\partial \delta}{\partial t'} \right) F \frac{\partial T_l}{\partial z'} = \alpha_l \frac{\partial}{\partial z'} \left( F \frac{\partial T_l}{\partial z'} \right). \tag{8.54}
\]
Cancelling like terms and expanding the right side of equation (8.54) yields,

$$F \frac{\partial T_i}{\partial t'} + u F \frac{\partial T_i}{\partial r} + v' F \frac{\partial T_i}{\partial z'} = \alpha_l \left[ F \frac{\partial^2 T_i}{\partial z'^2} + \frac{\partial F}{\partial z'} \frac{\partial T_i}{\partial z'} \right]. \quad (8.55)$$

The term $\frac{\partial F}{\partial z'}$ in equation (8.55) represents the gradient of the VOF function, in coordinates relative to the lower droplet surface. However, as the thermal diffusion length $\sqrt{\alpha t}$ was shown to be significantly smaller than the dimensions of the droplet, the VOF function can be assumed to be axially constant within the droplet volume in which we wish to predict liquid heat transfer rates. Thus, we can assume

$$\frac{\partial F}{\partial z'} = 0, \quad (8.56)$$

and equation (8.55) simplifies to give

$$\frac{\partial T_i}{\partial t'} + u \frac{\partial T_i}{\partial r} + v' \frac{\partial T_i}{\partial z'} = \alpha_l \frac{\partial^2 T_i}{\partial z'^2}. \quad (8.57)$$

Note that dashed terms in this equation must be evaluated in relative coordinates, and non-dashed terms in absolute coordinates.

In order to integrate equation (8.57), boundary conditions for heat conduction at the liquid-vapour interface and within the body of the droplet are required. At the liquid-vapour interface, Fourier’s law gives the rate of heat loss into the droplet as

$$q_i(r) = -k_l \frac{\partial T_i}{\partial z'}(r, 0). \quad (8.58)$$

As in the solid conduction analysis, the depth of the droplet is two orders of magnitude larger than the heat diffusion length into the droplet. Thus, the liquid can be approximated as a semi-infinite medium, and the second boundary condition is

$$T_i(r, \infty) = T_{i,l}. \quad (8.59)$$
Two forms of heat convection within the droplet have been neglected in this study; natural convection and Marangoni convection.

Natural convection is convection driven by buoyant forces within the fluid. The effects of natural convection must be considered in a heat transfer analysis when the inequality

$$\frac{Gr}{Re^2} \ll 1 \quad (8.60)$$

is true [Incropera & DeWitt 1996]. In Equation (8.60), $Re$ is the Reynolds number, and $Gr$ is the Grashof number, defined as

$$Gr = \frac{g\beta \Delta T d^3}{\nu^2}, \quad (8.61)$$

where $g$ is the acceleration due to gravity, $\beta$ is the coefficient of volumetric expansion for the liquid, $\Delta T$ is the change in temperature across the diameter of the droplet, $d$ is the diameter of the droplet, and $\nu$ is the kinematic viscosity of the fluid.

Using typical values for water droplets examined in this study, we find that the droplet Reynolds number is approximately $Re = 14,300$, and the Grashof number is approximately $Gr = 454,000$. Consequently, the ratio in the inequality Equation (8.60) is 0.002, and we are justified in neglecting the effects of natural convection.

Marangoni convection is convection driven by changes in the magnitude of surface tension along a fluid interface. These changes may be caused by changes in the liquid temperature along a fluid interface, as the magnitude of surface tension is dependent on the temperature of a fluid [Legros, Dupont, Queeckers, Van Vaerenbergh & Schwabe 1990].

The importance of Marangoni convection is gauged using the dimensionless thermal Marangoni number [Legros et al. 1990],

$$Ma = -\frac{\delta \sigma \Delta T d}{\delta T \mu k}, \quad (8.62)$$
where $\sigma$ is the surface tension at the fluid interface, $\Delta T$ is the temperature change along the interface, $d$ is the length of the interface, $\mu$ is the absolute viscosity of the fluid and $k$ is the thermal conductivity of the fluid. Substituting in typical values for water droplets examined in this study we find that $Ma \approx 1 \times 10^{-4}$. At this level of the Marangoni number, surface tension variations could not cause any fluid velocities to develop within a stationary droplet [Legros et al. 1990]. Consequently, our neglect of Marangoni convection is justified.

We now turn our attention to both vapour layer interfaces, where an energy balance and an examination of the conditions existing at those boundaries is made.

### 8.4.4 Solid-Vapour Interface

#### Energy Balance

An energy balance at the solid-vapour interface reveals

$$q_s (r) = q_v (r).$$

Substituting equation (8.39) for the heat loss rate from the solid, and equation (8.46) for the energy transfer rate across the vapour layer, yields

$$k_s \frac{\partial T_s}{\partial z} (r, 0) = \frac{k_m}{\delta} (T_{lv} (r) - T_{sv} (r)).$$

#### Interface Conditions

Following the analysis presented in Appendix A a molecular treatment is required to describe conditions existing at both the solid-vapour and liquid-vapour interfaces. When integrating the Navier-Stokes equations to provide the velocity distribution within the layer, this molecular treatment resulted in
a slip velocity existing at both interfaces. In a similar fashion, the analysis of Appendix A concludes that a temperature discontinuity may occur at both the solid and liquid interfaces, and its magnitude is specified by equation (A.23), for the case of a solid-liquid interface.

Thus, equation (A.23) gives a relationship between the temperatures on either side of the solid-liquid interface as

\[
T_{ss}(r) - T_{sv}(r) = \lambda_{m,s} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,s}} - 1 \right) \frac{dT_v}{dz} (r, 0), \tag{8.65}
\]

where \(\lambda_{m,s}\) is the average mean free path of the vapour mixture, evaluated at the temperature of the vapour at the solid-vapour interface \(T_{sv}\), and \(\sigma_{t,s}\) is the Thermal Accommodation Coefficient for the combination of solid surface and vapour mixture. Noting that the temperature profile within the vapour is linear, as given by equation (8.45), equation (8.65) becomes

\[
T_{sv}(r) - T_{ss}(r) = \lambda_{m,s} \delta \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,s}} - 1 \right) \left( T_{lv}(r) - T_{sv}(r) \right) \tag{8.66}
\]

and the conditions at the solid-vapour interface are specified.

### 8.4.5 Liquid-Vapour Interface

#### Energy Balance

In a similar manner to the solid-vapour interface, an energy balance at the liquid-vapour interface reveals that

\[
q_v(r) = q_l(r) + \dot{w} p_v H_{fg,c}. \tag{8.67}
\]

In this equation, the latent heat of vaporisation is corrected to account for the sensible heat contained within the vapour layer. Thus

\[
H_{fg,c} = H_{fg} (T_{sat} (P_{atmos})) + c_{p,v} \left( \frac{T_{s,i} - T_{sat} (P_{atmos})}{2} \right). \tag{8.68}
\]
Substituting equations (8.46) and (8.58) into equation (8.67) gives the energy balance at the interface as
\[
\frac{k_m}{\delta} (T_{sv}(r) - T_{lv}(r)) = -k_l \frac{\partial T_l}{\partial z'} (r, 0) + \dot{w} \rho_v H_{fg,c}.
\] (8.69)

**Mass Balance**

In addition to the equations specified for the solid-vapour interface, at the liquid-vapour interface a mass balance equation is required. Following the molecular treatment given in Section A.3.1 for evaporation rates and condensation rates existing at a liquid-vapour boundary, the droplet vaporisation rate is specified by equation (A.31) as
\[
\dot{w} = \frac{\sigma_{t,l}}{\rho_v} \sqrt{\frac{M_v}{2\pi R}} \left( \frac{P_{sat}(T_{li}(r))}{\sqrt{T_{li}(r)}} - \frac{P_v}{\sqrt{T_{lv}(r)}} \right).
\] (8.70)

Here \(\sigma_{t,l}\) is the Thermal Accommodation Coefficient for the combination of liquid and vapour mixture, \(M_v\) is the molecular weight of the droplet vapour [kg/kmol] and \(R\) is the universal gas constant [J/kmol.K]. The partial vapour pressure, \(P_v\), was defined in Section 8.3.

**Interface Conditions**

For the same reasons as outlined for the solid-vapour interface, a temperature discontinuity exists at the liquid-vapour interface. Following equation (A.47) of Section A.3.3, the magnitude of this discontinuity at the solid-vapour interface is given as
\[
T_{lv}(r) - T_{li}(r) = -\frac{\lambda_{m,li}}{\phi} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,li}} - 1 \right) \frac{dT_v}{dz} (r, \delta).
\] (8.71)

Noting that the temperature profile within the vapour is linear, as given by equation (8.45), equation (8.71) becomes
\[
T_{lv}(r) - T_{li}(r) = -\frac{\lambda_{m,li}}{\phi \delta} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,li}} - 1 \right) (T_{lv}(r) - T_{sv}(r)).
\] (8.72)
The variable $\phi$ was defined in Appendix A, Section A.3.3 to be the ratio of the molecular evaporation rate to the molecular condensation rate, and $\lambda_{m,l}$ is the average mean free path of the vapour mixture, evaluated at the temperature of the vapour at the liquid-vapour interface, $T_{lv}$. Utilising equations (A.26) and (A.28), this ratio can be expressed as

$$
\phi = \frac{J_e}{J_c} = \frac{\dot{w}_e \sqrt{T_{lv}(r)}}{\sigma_{lt} P_v} \sqrt{\frac{2\pi R}{M_v}} + 1, \quad (8.73)
$$

where the molecular variables are as defined in equation (8.70).

8.4.6 Heat Transfer Simplifications

Equations (8.38), (8.40), (8.57), (8.59), (8.64), (8.65), (8.69), (8.70) and (8.71) can be combined to yield the temperature distributions throughout the liquid, vapour and solid phases, as well as the droplet vaporisation velocity and heat transfer rates. This system of equations is quite complex, and can be computationally expensive to solve.

Under certain droplet impact conditions, simplifications to the above set of equations can be made, thus reducing the complexity of the problem, and in so doing, increasing the efficiency of the solution procedure. In this section, we outline some of the simplifications which can be used in predicting the droplet vaporisation rate.

Interface Molecular Treatment

As detailed in Section A.1, a molecular treatment is required at both vapour layer interfaces, as the height of the vapour layer during an impact can become comparable with the mean free path of the vapour mixture. If however, a droplet does not approach such small distances from the solid surface, the total volume of vapour within the layer may be regarded as a continuous
medium, and the molecular treatment neglected. Following the analysis of Section A.1, this assumption is valid if the maximum Knudsen number remains below 0.01 throughout a droplet impact.

If the molecular treatment is neglected, heat transfer within the solid, liquid and vapour phases is unaltered from the above, but conditions change at each vapour layer interface. Equation (8.65), which describes the magnitude of the temperature discontinuity at the solid-vapour interface, is replaced by

\[ T_{ss}(r) = T_{sv}(r), \quad (8.74) \]

as there is now no temperature discontinuity at the interface. Similarly, equation (8.71), which describes the magnitude of the temperature discontinuity at the liquid-vapour interface, is replaced by

\[ T_{ll}(r) = T_{lv}(r). \quad (8.75) \]

Also, the molecular mass balance at the liquid-vapour interface is no longer valid, and equation (8.70) is discarded. To close the heat transfer problem, we assume that the temperature of the liquid at the liquid-vapour interface is equal to the saturation temperature of the liquid at the vapour pressure within the layer. Thus

\[ T_{ll}(r) = T_{sat}(P_{v}(r)). \quad (8.76) \]

Calculating temperature distributions and heat transfer rates now requires solution to equations (8.38), (8.40), (8.57), (8.59), (8.64), (8.59), (8.69), (8.76) and (8.75). While this set of equations numbers the same as the set required by the molecular treatment case, two of the new set, equations (8.74) and (8.75) are trivial, and the exclusion of equation (8.70) means that the problem is no longer so highly non-linear. Consequently,
solution to the non-molecular treatment case is computationally easier and more efficient.

**Solid Heat Transfer**

If the thermal conductivity of the solid is high, then the conductive resistance within the solid is low and the temperature of the solid may be approximated as constant and uniform. Thus, equations (8.38) and (8.40) of the previous solution set are replaced by the single equality

$$T_{ss}(r) = T_{s,i}.$$  

(8.77)

Also, the energy balance over the solid-vapour interface, equation (8.64), now has no relevance, and is subsequently discarded.
Chapter 9

The Viscous Vapour Layer Model — Numerical Solution Procedure

In this chapter we outline the procedure used by BOUNCE to solve the viscous vapour layer fluid flow and heat transfer equations which were developed in the previous chapter.

9.1 Numerical Solution Overview

9.1.1 Computational Mesh

The solution to flow within the viscous vapour layer is accomplished on a computational mesh composed of a single row of ‘subcells’, so termed because an integer multiple of subcells exists within each column of VOF computational cells. The number of subcells contained within each column of VOF cells is represented by the variable $o_{\text{max}}$. Thus, the total number of subcells
The height of subcell \( o \) is determined from VOF data as \( \hat{\delta}_o \) [m], and is contained within the computational domain is \((i_{\text{max}} - 2) \times o_{\text{max}}\), where as previously defined, \( i_{\text{max}} - 2 \) is the number of VOF columns contained within the computational domain.

Two dummy subcells, located beyond the bounds of the computational domain, are used to impose boundary conditions on the flow solution. One dummy cell is located to the left of the centreline, the other dummy cell to the right of the right-hand computational boundary. The subcells are numbered sequentially from the left dummy cell outwards. Thus, the left most cell contained within the computational domain is \( o = 2 \), and the right most cell contained within the computational domain is \( o = ((i_{\text{max}} - 2) \times o_{\text{max}}) + 1 \).

Variable positioning within the vapour layer subcells is displayed schematically in Figure 9.1, where in this example, \( o_{\text{max}} = 2 \). Integer subscripts indicate that the property is either averaged across a subcell, or located at the centre of a subcell. Non-integer subscripts indicate that a property lies on a subcell boundary.
uniform for all subcells contained within the single VOF cell. The average gauge pressure acting on the lower surface of the droplet within subcell \( o \) is \( P_o \) [Pa], while the average vapour mixture density and air volume fraction within the same subcell are represented by \( \rho_{m,o} \) [kg/m\(^3\)] and \( y_{a,o} \), respectively. Variables located on subcell borders are the radial location, \( r_{o+\frac{1}{2}} \) [m], and the vapour volume flow rate, \( S_{o+\frac{1}{2}} \) [m\(^3\)/s]. The average droplet vaporisation velocity within subcell \( o \) is represented by \( \dot{w}_o \) [m/s].

9.1.2 Computational Solution Structure

Figure 9.2 shows the computational procedure used by subroutine VAPOR-LAYER to calculate the flow conditions existing within the vapour layer. Firstly, subroutine VAPORHEIGHT is called to determine the height of the vapour layer. This height is calculated from the internal droplet dynamics VOF function, and the method used to accomplish this calculation is outlined in Section 9.2.

The subroutine called to solve for the vapour layer pressures and flowrates in VAPORLAYER is dependent on the vapour layer assumptions used in the particular droplet impact simulation. If no heat transfer into the liquid phase is modelled, and if no molecular treatment of vapour layer interfaces is modelled, then subroutine VAPORNORM is called to calculate vapour layer pressures and flowrates. If however, either liquid phase heat transfer or layer interface molecular treatments are included in the simulation, then subroutine VAPORSUB is called to calculate vapour layer pressures and flowrates.

Subroutine VAPORSUB is a more complex and computationally expensive subroutine than VAPORNORM, because the equations which must be solved by subroutine VAPORSUB are non-linear functions of the vapour
layer pressure. As a result, an iterative method must be used by the more complex subroutine to determine the vapour layer pressures and flowrates. The procedure used by subroutine \textit{VAPORSUB} to calculate these vapour layer flow conditions is the subject of the remainder of this chapter.

As subroutine \textit{VAPORNORM} is a simplification of subroutine \textit{VAPORSUB}, and as it can be used in simulating only a limited number of real droplet impacts, the procedure used by subroutine \textit{VAPORNORM} will not be detailed in this text. However, a full listing of both subroutines is included in the code listing, Appendix E.

The computational procedure used by subroutine \textit{VAPORSUB} is shown schematically in Figure 9.3. As shown, the subroutine consists primarily of two nested iterative loops, and contained within these loops is the coding used to calculate the vapour layer pressure distribution.
Figure 9.3: Schematic showing the procedure used by subroutine VAPORSUB to calculate vapour layer pressures.
The coding used to calculate the vapour layer pressures is composed of two sections. In the first, the droplet vaporisation velocity constants $\alpha_o$ and $\beta_o$ are found for each vapour layer subcell, such that the vaporisation velocity within each subcell can be expressed as a function of the vapour layer pressure,

$$\dot{w}_o = \alpha_o P_o + \beta_o. \quad (9.1)$$

The droplet vaporisation velocity constants, $\alpha_o$ and $\beta_o$, are weakly dependent on the previous vapour layer pressure, air volume fraction and vapour layer interface temperature distributions. The method used to calculate these constants is detailed in Section 9.3 and utilises the heat transfer equations developed in the previous chapter, Section 8.4.

In the second section of the vapour pressure coding, the momentum and mass conservation equations developed in the previous chapter, Section 8.2, are solved to determine the pressure distribution and vapour mixture flowrates within the vapour layer. This calculation requires values for the droplet vaporisation velocity constants, $\alpha_o$ and $\beta_o$, as provided by the droplet vaporisation coding. The method used to solve the mass and momentum conservation equations within the vapour layer is outlined in Section 9.4.

An iterative vapour layer solution method is required in BOUNCE because of the dependence between the droplet vaporisation constants and the previously calculated vapour layer pressure distribution, interface temperatures and air fraction distribution. The nested inner and outer loops shown in Figure 9.3 are used to perform these iterations.

The inner iterative loop is responsible for converging a solution to the vapour layer pressure distribution, given a constant air volume fraction distribution. The outer iterative loop is used to converge a solution to the air
volume fraction distribution. Thus, only once the inner loop has converged on a vapour layer pressure distribution is a new air volume fraction calculated by the outer loop, and the iterations returned to the inner loop with the new air volume fraction distribution. This two stage convergence method was chosen because of the highly non-linear nature of the viscous vapour layer problem.

The convergence of both inner and outer loop solutions is gauged by comparing the maximum change in vapour layer pressures between the start and finish of each loop cycle against a vapour layer pressure comparator, $\varepsilon_V$. This comparator is related to the internal droplet dynamics VOF algorithm implicit iteration pressure comparator, $\varepsilon_P$, using

$$\varepsilon_V = \frac{\max (P_{i,j}) \varepsilon_P}{10},$$

(9.2)

where $\max (P_{i,j})$ is the maximum pressure calculated during the last VOF code iteration. The purpose of equation (9.2) is to ensure that the accuracy of pressures calculated within the vapour layer is always higher than the accuracy required for the convergence of the internal droplet VOF algorithm.

A consequence of the highly non-linear nature of the vapour layer problem is that the vapour layer pressure and air volume fraction distributions must be under-relaxed after each iterative loop cycle. Under-relaxation in these calculations is dynamic, and a separate under-relaxation factor is used for each of the two iterative loops.

Several subroutines are called by VAPORSUB. Subroutines HEATLIQUID, HEATSETUP, COOLSOLID, and COOLSETUP are responsible for initialising and solving the temperature distribution calculations within the liquid and solid phases. Subroutine YASOLVER convects the air volume fraction solution, using the equations developed later in Section 9.5. Subroutine VCOURANT calculates the maximum stable timestep for the air
volume fraction calculation, as detailed in Section 9.5.3.

In the BOUNCE vapour layer code, considerable use is made of a tridiagonal matrix inversion algorithm, taken from Press et al. [1992]. Tridiagonal matrices often result from sets of linear equations which model one-dimensional physical problems with a diffusion operator. In the vapour layer code, the vapour pressure equations, axial liquid and solid temperature distribution equations, and air volume fraction equations each model one-dimensional physical problems, and each set of equations is expressed as a tridiagonal matrix. The inversion algorithm used in this study uses LU decomposition [Press et al. 1992]. The algorithm is very stable and extremely efficient, both of which are important factors given the stiffness and non-linearity of the droplet impact problem.

9.2 Vapour Layer Height

9.2.1 The Height Calculation Method

The vapour layer height is calculated for each VOF column of cells which contains liquid. It is defined as the distance between the solid surface and the lower droplet surface. The solid surface is located at the top border of the VOF cell \( j = 3 \), so the vapour layer height is approximated from the VOF data using

\[
\delta_i = y_{j_{\text{top}} + \frac{1}{2}} - y_{3 + \frac{1}{2}} - \sum_{j=2}^{j_{\text{top}}} F_{i,j} \delta y_j, \tag{9.3}
\]

where \( y_{j + \frac{1}{2}} \), as previously defined in Chapter 4, locates the upper border of the VOF cells in row \( j \). The integer \( j_{\text{top}} \) is determined by looping upwards through each column of cells. It is defined as the \( j \) index of the first full cell, or alternatively the \( j \) index of the first fluid cell whose upper neighbour
contains less VOF fraction than it does. If the column contains no fluid, the height is set to $-1$.

To demonstrate, consider the simple droplet placed above the solid surface, as shown in Figure 9.4. In this figure the shaded grey area represents the amount of liquid contained in each cell, as determined from the VOF data.

Column $i = 2$ of Figure 9.4 demonstrates the simplest case of vapour layer height calculation. In this case $j_{\text{top}} = 6$, the first full fluid cell to be
encountered when moving upwards along column $i = 2$, and from equation (9.3), the vapour layer height becomes,

$$\delta_2 = y_{6+\frac{1}{2}} - y_{3+\frac{1}{2}} - \sum_{j=2}^{6} F_{2,j} \delta y_j = y_{6+\frac{1}{2}} - y_{3+\frac{1}{2}} - F_{2,5} \delta y_5 - \delta y_6. \quad (9.4)$$

Column $i = 5$ of Figure 9.4 demonstrates an example where more than one partially full cell is used in the calculation of the vapour layer height. In this case, $j_{\text{top}} = 7$. At the periphery of the example droplet, that is within column $i = 6$, the proportion of fluid in the cell $j = 8$ is less than that contained in $j = 7$, and consequently $j_{\text{top}} = 7$. Referring to equation (9.3), the vapour layer height for this column is,

$$\delta_6 = y_{7+\frac{1}{2}} - y_{3+\frac{1}{2}} - \sum_{j=2}^{7} F_{6,j} \delta y_j = y_{7+\frac{1}{2}} - y_{3+\frac{1}{2}} - F_{6,6} \delta y_6 - F_{6,7} \delta y_7. \quad (9.5)$$

When the determination of $j_{\text{top}}$ is governed by the second requirement, namely that the cell above has a smaller proportion of fluid volume than the examined cell, discrete jumps in vapour layer heights can occur between timesteps. In the example of Figure 9.4 for example, if the VOF value of the cell specified by $i = 6$ and $j = 8$ became greater than the VOF value of the cell specified by $i = 6$ and $j = 7$, a discrete jump in the layer height would occur as $j_{\text{top}}$ changed from 7 to 8. In practice however, these heights are usually located at the periphery of the droplet geometry where the vapour layer height is large and generated vapour layer pressures are low. Thus, discrete jumps in layer height have little effect on droplet calculations.

A vapour layer height, $\hat{\delta}_o$, is also defined for each individual vapour layer subcell. The magnitude of this height is equal to the magnitude of the VOF cell vapour layer height, $\delta_i$, in which the subcell is contained.

The dynamics of droplet impingement are critically dependent on the thickness of the vapour layer. Equation (9.3) represents the most satisfactory
method that was found for calculating this thickness, but several additional features were used in the interests of algorithm stability.

9.2.2 The Accuracy and Stability of the Vapour Layer Height Calculation

While the position of the solid surface is located at the top of cell \( j = 3 \), small amounts of fluid can enter the VOF cells at heights \( j = 3 \) and \( j = 2 \) and the vapour layer height still remain real and positive. This usually presents little problem during droplet calculations, but it can cause difficulties if droplet breakup is prevalent. In such cases, rapidly changing surface curvatures generated when droplets are separating or coalescing can cause small amounts of fluid to separate from the bulk volumes. If such fluid debris moves below the solid surface level, and the computational column contains no other fluid, an unrealistic value for the vapour layer height may result.

A repacking algorithm is included in BOUNCE to prevent such problems occurring. This algorithm moves any fluid located below the solid surface back above the surface. In effect, this feature repacks the fluid VOF data in each column from \( j = j_{\text{top}} \) downwards, filling the computational cells as it goes, until the calculated vapour layer height is reached. The feature only operates if the vapour layer height is less than \( \delta y_4 \), or alternatively if fluid has entered cell \( j = 2 \).

The limit of accuracy achieved in calculating the height of the vapour layer is dependent on the accuracy of the VOF data. If the PPE residual comparator is set to a typical value of \( \varepsilon_P = 10^{-5} \), the corresponding error in the VOF values may be of the order \( 10^{-5} \). Assuming cell dimensions of the order 0.1 mm, the limit of accuracy for vapour layer height calculations is 1 nm. If vapour layer heights of this magnitude are developed in computa-
tions, many iterations are required to achieve solution convergence, and as a result, progress effectively stops as the computational timestep tends to zero.

In practice, the limit of accuracy of the vapour layer height calculations does not produce difficulties for the computational algorithm because at this height, certain assumptions used in the development of the viscous vapour layer code are not applicable. For example, 1 nm is well below the surface roughness limit of practically all solid surfaces. A surface which has been very smoothly finished by fine honing, lapping or buffing has a minimum surface roughness grade of approximately 25 nm [Boundy 1986]. This is well above the vapour layer height accuracy limit.

Also, as is discussed in Appendix A.1 at this distance between droplet and solid surface, properties of the vapour mixture within the layer do not obey continuum theories, and the viscous vapour layer flow treatment is not valid. Thus, the accuracy of the internal droplet dynamics VOF code does not restrict the ability of BOUNCE to model actual droplet impacts.

If the situation arises when the pressure generated by the vapour layer is insufficient to repel the liquid, and the vapour layer height becomes less than a specified minimum height, say \( \delta_{\text{min}} \), then we assume that accurate calculation of vapour flow within the viscous layer is no longer being achieved. We can allow computations to continue, bearing in mind that the resulting simulation may not be physical, by assuming that direct contact between the solid and liquid occurs. Under such circumstances, the vapour layer height for all heat transfer and vapour layer calculations is given by,

\[
\delta_i = \delta_{\text{min}}. \tag{9.6}
\]

and additional pressure at the lower surface of the droplet, above that generated by the vapour layer, is used to stop the lower surface of the droplet.
9.2.3 The Velocity of the Underside of the Droplet

Different routines within the vapour layer model require a knowledge of the velocity of the underside of the droplet. This velocity is calculated using

\[
\frac{\partial \delta_o}{\partial t} = v(x, y) + \frac{\dot{w}_o \rho_f}{\rho_v},
\]

(9.7)

where \( v(x, y) \) is the velocity of the droplet fluid at the absolute location \( x = r_o \) and \( y = y_{3+\frac{1}{2}} + \delta_o \), interpolated from the main VOF internal droplet dynamics algorithm velocities. The velocity of the droplet underside surface is greater than the velocity of the droplet fluid at the underside of the droplet, as liquid is vaporising from the lower surface of the droplet. In equation (9.7), \( \dot{w}_o \) is the most recently calculated average vaporisation velocity associated with subcell \( o \).

An alternative method for computing the droplet underside velocity is by directly differencing the calculated vapour heights between timesteps. This method, however, leads to unrealistic values of vapour height velocities if discrete jumps in layer height are experienced, and was consequently not used.

9.3 Vaporisation Velocity Calculation

In this section we detail the numerical procedure used to calculate the droplet vaporisation velocity as a function of the vapour layer pressure. The heat transfer equations which define the vaporisation velocity were detailed in the previous chapter, Section 8.4.
Figure 9.5: The temperature distribution within the solid phase is represented on a variable mesh size grid. This figure shows the temperature variables at the radial location corresponding to \( r = r_o \).

### 9.3.1 Solid Phase Heat Transfer

Heat transfer within the solid phase is governed by the heat diffusion equation (8.38) and boundary condition equations (8.40) and (8.64) of the previous chapter. A heat transfer solution is found by discretising these equations over a variable size grid within the solid phase.

#### Solid Temperature Mesh

Figure 9.5 shows the computational mesh used to represent the temperature distribution within the solid. The nodes in this figure represent the temperature within the solid at only one radial location, \( r = r_o \) — a solid phase temperature distribution is calculated for each vapour layer subcell. A variable grid size mesh is used within the solid because the heat transfer rates at the surface of the solid can be large, necessitating a fine temperature node spacing in order to accurately represent the steep temperature gradients.

The solid phase mesh spacing is calculated using a geometric series. Three
equidistant nodes near the solid-vapour interface are first located using

\[ z_{k_{\text{max}}} = \frac{\delta z_{s,\text{min}}}{2}, \quad (9.8) \]

\[ z_{k_{\text{max}}-1} = -\frac{\delta z_{s,\text{min}}}{2} \quad (9.9) \]

and

\[ z_{k_{\text{max}}-2} = -\frac{3\delta z_{s,\text{min}}}{2}, \quad (9.10) \]

where \( \delta z_{s,\text{min}} \) is the user defined minimum solid phase mesh size. The remainder of the temperature nodes are then located using the recursion series

\[ \delta z_k = \chi_s \delta z_{k+1}, \quad (9.11) \]

where \( \chi_s \) is the user defined solid mesh stretch rate, and the distance between the nodes is defined as

\[ \delta z_k = z_{k+1} - z_k. \quad (9.12) \]

**Solid Thermal Energy Transport Equation**

As the droplet impact problem is critically dependent on the heat transfer rate from the solid, an implicit solution procedure has been implemented to discretise equation (8.38). Using the variable grid size difference equation

\[ \frac{\partial^2 T_k}{\partial z^2} = \frac{1}{(\delta z_k + \delta z_{k-1})} \left[ \frac{T_{k+1}}{\delta z_k} - \left( \frac{1}{\delta z_{k-1}} + \frac{1}{\delta z_k} \right) T_k + \frac{T_{k-1}}{\delta z_{k-1}} \right], \quad (9.13) \]

equation (8.38) becomes

\[ \frac{T_{o,k} - T_{o,k}}{\delta t} = \alpha_s \left\{ \frac{1}{(\delta z_k + \delta z_{k-1})} \left[ \frac{T_{o,k+1}}{\delta z_k} - \left( \frac{1}{\delta z_{k-1}} + \frac{1}{\delta z_k} \right) T_{o,k} + \frac{T_{o,k-1}}{\delta z_{k-1}} \right] \right\}. \quad (9.14) \]
Here a first order discretisation in time has been used, and $T_{o,k}^*$ represents the solid temperature from the previous timestep. Note that the space discretisation equation (9.13) is second order accurate on a uniformly sized mesh, reducing to first order accurate on a variable sized mesh.

Equation (9.14) can be rearranged to give a linear relationship between three adjacent temperatures,

$$\alpha_k T_{o,k-1} + \beta_k T_{o,k} + \gamma_k T_{o,k+1} = \theta_k,$$  \hspace{1cm} (9.15)

where

$$\alpha_k = -\frac{\alpha_s \delta t}{\delta z_{k-\frac{1}{2}} \delta z_{k-1}},$$ \hspace{1cm} (9.16)

$$\beta_k = 1 + \frac{\alpha_s \delta t}{\delta z_{k-\frac{1}{2}}^2} \left( \frac{1}{\delta z_{k-1}} + \frac{1}{\delta z_k} \right),$$ \hspace{1cm} (9.17)

$$\gamma_k = -\frac{\alpha_s \delta t}{\delta z_{k-\frac{1}{2}} \delta z_k},$$ \hspace{1cm} (9.18)

$$\theta_k = T_{o,k}^*$$  \hspace{1cm} (9.19)

and

$$\delta z_{k-\frac{1}{2}} = \frac{\delta z_{k-1} + \delta z_k}{2}.$$ \hspace{1cm} (9.20)

Note that equations (9.16)–(9.19) are only valid within the range $2 \leq k \leq k_{max} - 1$, where $k_{max}$ is the total number of solid mesh temperature nodes at each radial location.

**Vapour Surface and Internal Boundary Conditions**

Boundary condition equations must also be discretised to define a heat transfer solution within the solid. Within the volume of the solid, the constant
temperature boundary condition of equation (8.40) is represented by

\[ T_{o,1} = T_{s,i}, \quad (9.21) \]

where \( T_{s,i} \) is the initial temperature of the solid. In terms of the solid heat transfer coefficients defined by equation (9.15), this internal boundary condition can be expressed as

\[ \alpha_1 = 0, \quad \beta_1 = 1, \quad \gamma_1 = 0 \quad \text{and} \quad \theta_1 = T_{s,i}. \quad (9.22) \]

At the surface of the solid, the temperature nodes are equally spaced, and the gradient of the solid phase temperature is given by

\[ \frac{\partial T_s}{\partial z} (r_o, 0) = \frac{T_{0,k_{\max}} - T_{0,k_{\max}-1}}{\delta z_{s,\min}}. \quad (9.23) \]

Also, the temperature of the solid surface is defined as the average of the two equally spaced temperature nodes,

\[ T_{ss,o} = T_s (r_o, 0) = \frac{T_{0,k_{\max}} + T_{0,k_{\max}-1}}{2}. \quad (9.24) \]

The front surface boundary condition equation (8.64) could be discretised using equations (9.23) and (9.24). However, in the interests of computational efficiency, this discretisation is not directly implemented in the BOUNCE code.

**Solid Phase Solution Procedure**

Equation (9.15), coupled to a front surface boundary condition and the internal solid boundary condition of equation (9.21), provides a set of equations defining the temperature distribution within the solid. Such a set can be solved using a tridiagonal matrix inverter, which as previously discussed, is an efficient and stable method for inverting such systems of equations [Press et al. 1992].
In BOUNCE, the heat transfer rate from the solid is dependent on the relationship between the temperature and temperature gradient of the solid at the solid-vapour interface. This relationship is expressed by the boundary condition equation (8.64).

A method of determining the heat transfer rate from the solid could be to calculate the temperature distribution within the solid, while applying equation (8.64). However, the iterative method used by BOUNCE to solve the viscous vapour layer flow equations requires the heat transfer rate from the solid to be calculated numerous times during each timestep. Thus, the solid phase temperature distribution, calculated by solving the thermal energy transport equations using the tridiagonal matrix inversion algorithm, would be evaluated numerous times during each timestep. As matrix inversions are computationally expensive, the vapour layer calculation algorithm would be prohibitively expensive and inefficient. Consequently, a less direct method of calculating the solid phase heat transfer rate is employed in BOUNCE.

The discretised heat diffusion equation (9.15) is a linear function of the solid temperatures at the computational nodes. As a result, the relationship between the temperature and temperature gradient at the solid surface can also be expressed as a linear relationship,

\[
\frac{\partial T_s}{\partial z} (r_o, 0) = I_{s,o} (T_{ss,o} + J_{s,o}). \tag{9.25}
\]

Equation (9.25) is valid for the duration of each timestep, \( I_{s,o} \) and \( J_{s,o} \) are solid phase heat transfer constants within subcell \( o \), and \( T_{ss,o} = T_s (r_o, 0) \) is the temperature of the solid within subcell \( o \) at the solid-vapour interface.

By using equation (9.25) to calculate the heat transfer rate from the solid, rather than solving for the complete temperature distribution within the solid at each vapour layer iteration, the number of computationally expensive matrix inversions are minimised during each timestep calculation.
In determining the solid phase heat transfer constants \( I_{s,o} \) and \( J_{s,o} \), we note that the solid heat transfer coefficients \( \alpha_k, \beta_k \) and \( \gamma_k \) are dependent only on the geometry of the solid phase temperature mesh, and the constant thermal properties of the solid. Thus, differentiating equation (9.15) with respect to the temperature of the solid surface, \( T_{ss,o} \), gives

\[
\alpha_k \frac{\partial T_{o,k-1}}{\partial T_{ss,o}} + \beta_k \frac{\partial T_{o,k}}{\partial T_{ss,o}} + \gamma_k \frac{\partial T_{o,k+1}}{\partial T_{ss,o}} = 0, \tag{9.26}
\]

as \( \theta_k \) is independent of the solid surface temperature. Defining

\[
\xi_{s,k} = \frac{\partial T_{o,k}}{\partial T_{ss,o}} / \frac{\partial T_{o,k+1}}{\partial T_{ss,o}}, \tag{9.27}
\]

equation (9.26) can be rearranged to give

\[
\xi_{s,k} = -\frac{\gamma_k}{\beta_k + \alpha_k \xi_{s,k-1}}. \tag{9.28}
\]

At the rear surface of the solid, the boundary condition equation (9.21) can be differentiated to give

\[
\xi_{s,1} = 0. \tag{9.29}
\]

Using a recursion analysis of equation (9.28), combined with the initial condition equation (9.29), all \( \xi_{s,k} \) variables within the range \( 1 \leq k \leq k_{\text{max}} - 1 \) may be calculated.

Defining the subcell constant

\[
Z_{s,o} = \xi_{s,k_{\text{max}}-1}, \tag{9.30}
\]

we have from (9.27),

\[
\frac{\partial T_{o,k_{\text{max}}}}{\partial T_{ss,o}} Z_{s,o} = \frac{\partial T_{o,k_{\text{max}}-1}}{\partial T_{ss,o}}. \tag{9.31}
\]

Also, differentiating equation (9.24) with respect to \( T_{ss,o} \) gives

\[
2 = \frac{\partial T_{o,k_{\text{max}}}}{\partial T_{ss,o}} + \frac{\partial T_{o,k_{\text{max}}-1}}{\partial T_{ss,o}}. \tag{9.32}
\]
Combining equation (9.31) with equation (9.32) gives
\[
\frac{\partial T_{o,k_{\text{max}}}}{\partial T_{ss,o}} = \frac{2}{1 + Z_{s,o}}
\] (9.33)
and
\[
\frac{\partial T_{o,k_{\text{max}} - 1}}{\partial T_{ss,o}} = \frac{2Z_{s,o}}{1 + Z_{s,o}}.
\] (9.34)

Equation (9.23) gives the solid surface temperature gradient in terms of the discretised temperatures. Taking the derivative of this gradient with respect to \(T_{ss,o}\) gives
\[
\frac{\partial}{\partial T_{ss,o}} \left( \frac{\partial T_s}{\partial z} (r_o, 0) \right) = \frac{1}{\delta z_{s,\text{min}}} \left( \frac{\partial T_{o,k_{\text{max}}}}{\partial T_{ss,o}} - \frac{\partial T_{o,k_{\text{max}} - 1}}{\partial T_{ss,o}} \right).
\] (9.35)
Substituting in equations (9.33) and (9.34) yields,
\[
\frac{\partial}{\partial T_{ss,o}} \left( \frac{\partial T_s}{\partial z} (r_o, 0) \right) = \frac{2}{\delta z_{s,\text{min}}} \left( \frac{1 - Z_{s,o}}{1 + Z_{s,o}} \right).
\] (9.36)
Also, differentiating equation (9.25) with respect to \(T_{ss,o}\) gives
\[
\frac{\partial}{\partial T_{ss,o}} \left( \frac{\partial T_s}{\partial z} (r_o, 0) \right) = I_{s,o}.
\] (9.37)
Comparing equations (9.36) and (9.37) gives the solid phase heat transfer constant,
\[
I_{s,o} = \frac{2}{\delta z_{s,\text{min}}} \left( \frac{1 - Z_{s,o}}{1 + Z_{s,o}} \right).
\] (9.38)
Note that \(I_{s,o}\) is only dependent on the properties of the solid, the magnitude of the timestep and geometry of the temperature mesh. Thus, \(I_{s,o}\) is constant over the duration of a timestep.

The second solid phase heat transfer constant, \(J_{s,o}\), is calculated by evaluating the temperature distribution within the solid phase using equations (9.15) and (9.21), while assuming
\[
\frac{\partial T_s}{\partial z} (r_o, 0) = 0.
\] (9.39)
From equation (9.25), this yields

\[ J_{s,o} = -T_{ss,o}. \] (9.40)

Substituting equation (9.39) into equation (9.23) gives the front surface boundary condition required for this procedure,

\[ T_{o,k_{\text{max}}} - T_{o,k_{\text{max}}-1} = 0. \] (9.41)

In terms of the solid heat transfer coefficients defined by equation (9.15), this front boundary condition can be expressed as

\[ \alpha_{k_{\text{max}}} = -1, \beta_{k_{\text{max}}} = 1, \gamma_{k_{\text{max}}} = 0 \text{ and } \theta_{k_{\text{max}}} = 0. \] (9.42)

Note that the solid surface temperature, required for the evaluation of \( J_{s,o} \), is given by equation (9.24) as a function of the computational node temperatures.

**Final Solid Phase Temperature Distribution**

When the vapour layer solution has converged and the solid surface temperatures are known, the new timestep solid temperature distribution is calculated by again inverting the system of equations given by (9.15) and internal boundary condition equation (9.21), this time with the front surface boundary condition given directly by equation (9.24). In terms of the solid heat transfer coefficients, this front boundary condition can be expressed as

\[ \alpha_{k_{\text{max}}} = 1, \beta_{k_{\text{max}}} = 1, \gamma_{k_{\text{max}}} = 0 \text{ and } \theta_{k_{\text{max}}} = 2T_{ss,o}. \] (9.43)

The total number of solid phase temperature matrix inversions required for each implicit timestep iteration using the above method is two — one for the evaluation of the solid phase constant \( J_{s,o} \), the other to determine
the final solid phase temperature distribution. This number of inversions is considerably less than if the entire temperature distribution were solved at each vapour layer iteration, as numerous vapour layer iterations are typically required to advance the simulation one timestep. An integrity test of the solid phase heat transfer algorithm is presented in Appendix D.

9.3.2 Liquid Phase Heat Transfer

The calculation of heat transfer within the liquid phase is achieved using a similar procedure to that used within the solid phase, however, in the liquid phase convection as well as conduction is included in the analysis. Heat transfer within the liquid phase is governed by equation (8.57), coupled to the boundary condition equations (8.58) and (8.59) of the previous chapter. Similar to the solid phase analysis, a heat transfer solution is found by discretising these equations over a variable size grid within the liquid phase.

Liquid Temperature Mesh

Figure 9.6 shows the computational mesh used to represent the temperature distribution within the liquid. As for the solid phase, the liquid phase mesh spacing is calculated using a geometric series. Three equidistant nodes near the liquid surface are first located using

\[ z'_{k_{\text{max}} + 1} = -\frac{\delta z_{l,\text{min}}}{2}, \]  

(9.44)

\[ z'_{k_{\text{max}} + 2} = \frac{\delta z_{l,\text{min}}}{2} \]  

(9.45)

and

\[ z'_{k_{\text{max}} + 3} = \frac{3\delta z_{l,\text{min}}}{2}, \]  

(9.46)
Figure 9.6: The temperature distribution within the liquid phase is represented on a variable mesh size grid. This figure shows the temperature variables at the radial location corresponding to \( r = r_o \).

where \( \delta z_{l,\min} \) is the user defined minimum liquid phase mesh size. The remainder of the temperature nodes are then located using the recursion series

\[
\delta z_k = \chi_l \delta z_{k+1}, \tag{9.47}
\]

where \( \chi_l \) is the user defined liquid mesh stretch rate, and the distance between the nodes is defined as

\[
\delta z_k = z'_{k+1} - z'_k. \tag{9.48}
\]

Note that equations (9.47) and (9.48) are only valid within the range \( k_{\max} + 3 \leq k \leq k_{\max} + l_{\max} - 1 \), where \( l_{\max} \) is the total number of liquid mesh temperature nodes at each radial location, and that the displacement \( z' \) is defined relative to the lower droplet surface, \( \hat{\delta}_o \).

Liquid Thermal Energy Transport Equation

Equation (8.57) of the previous chapter is used to model heat transfer rates within the liquid phase. In discretising this equation, we note that the axial
conduction term is usually of far greater magnitude than either convective term, and subsequently the convective terms can be modelled using an explicit discretisation, while the conductive term is modelled using an implicit discretisation.

Substituting the variable grid size difference equation (9.13) of the previous section into equation (8.57) we find

\[
\frac{T_{o,k} - T_{o,k}^*}{\delta t} + u_{o,k} \frac{\partial T_l}{\partial r}_{o,k} + v'_{o,k} \frac{\partial T_l}{\partial z'}_{o,k} = \alpha_l \left\{ \frac{1}{(\delta z_k + \delta z_{k-1})} \left[ \frac{T_{o,k+1}}{\delta z_k} - \left( \frac{1}{\delta z_{k-1}} + \frac{1}{\delta z_k} \right) \frac{T_{o,k} + T_{o,k-1}}{\delta z_{k-1}} \right] \right\},
\]

(9.49)

where the explicit discretisation of the convective terms is deferred to the next section. In a similar manner to the solid equation, equation (9.46) can be rearranged into the form

\[
a_k T_{o,k-1} + \beta_k T_{o,k} + \gamma_k T_{o,k+1} = \theta_k,
\]

(9.50)

where

\[
a_k = -\frac{\alpha_l \delta t}{\delta z_{k-1/2} \delta z_{k-1}},
\]

(9.51)

\[
\beta_k = 1 + \frac{\alpha_l \delta t}{\delta z_{k-1/2}} \left( \frac{1}{\delta z_{k-1}} + \frac{1}{\delta z_k} \right),
\]

(9.52)

\[
\gamma_k = -\frac{\alpha_l \delta t}{\delta z_{k-1/2} \delta z_k},
\]

(9.53)

and

\[
\theta_k = T_{o,k}^* - \delta t \left\{ u_{o,k} \frac{\partial T_l}{\partial r}_{o,k} + v'_{o,k} \frac{\partial T_l}{\partial z'}_{o,k} \right\}.
\]

(9.54)

Equations (9.50)–(9.54) are valid only within the range \(k_{\text{max}} + 1 \leq k \leq k_{\text{max}} + l_{\text{max}} - 1\).
Thermal Energy Convective Transport Terms

Evaluating each convective term in equation (9.54) involves evaluating a fluid velocity and a temperature gradient. The relative axial velocity, \( v'_{o,k} \), is defined, using equation (8.53), as

\[
v'_{o,k} = v_{o,k} - \frac{\partial \delta_o}{\partial t}.
\]  
(9.55)

The absolute axial velocity, \( v_{o,k} \), is interpolated from internal droplet VOF velocities at the absolute location defined by

\[
x_{o,k} = r_o \text{ and } y_{o,k} = y_{3+\frac{1}{2}} + \delta_o + z'_{k}.
\]  
(9.56)

Calculation of the rate of change of the droplet vapour layer height was detailed in Section [9.2.3]. In a similar manner to the axial velocity, the absolute radial velocity, \( u_{o,k} \), is interpolated from internal droplet VOF velocities at the same absolute location defined by equation (9.56).

Both temperature gradients required for the convective heat transfer terms are discretised using a first order upwind scheme. For the axial term, the gradient is discretised using

\[
\frac{\partial T_l}{\partial z'}_{o,k} = \begin{cases} 
T_{o,k} - T_{o,k-1} \frac{\delta z_{k-1}}{\delta z_k} & \text{if } v'_{o,k} \geq 0 \\
T_{o,k+1} - T_{o,k} \frac{\delta z_k}{\delta z_{k-1}} & \text{if } v'_{o,k} < 0,
\end{cases}
\]  
(9.57)

The discretisation of the radial term is slightly more complex, because this term must be evaluated in absolute coordinates. Thus, we define

\[
\frac{\partial T_l}{\partial r}_{o,k} = \begin{cases} 
T_{o,k} - T_{o-1,o,k} \frac{\delta r_{o-1}}{\delta r_{o+\frac{1}{2}}} & \text{if } u_{o,k} \geq 0 \\
\overline{T}_{o+1,o,k} - T_{o,k} \frac{\delta r_{o+\frac{1}{2}}}{\delta r_{o-\frac{1}{2}}} & \text{if } u_{o,k} < 0,
\end{cases}
\]  
(9.58)

where

\[
\delta r_{m+\frac{1}{2}} = \frac{\delta r_{m+1} + \delta r_m}{2}.
\]
The average temperature of the adjacent temperature cell is evaluated using

$$T_{n,o,k} = \frac{T_{a,k} + T_{o,k} \left( \delta z_{k - \frac{1}{2}} - z_{a,k} \right)}{\delta z_{k - \frac{1}{2}}}, \quad (9.59)$$

where

$$z_{a,k} = \sum_{l=k_{\text{max}}+1}^{k_{\text{max}}+l_{\text{max}}} \max \left[ \min \left( z_{k + \frac{1}{2}} + \delta_o, z_{l + \frac{1}{2}} + \delta_n \right) 
- \max \left( z_{k - \frac{1}{2}} + \delta_o, z_{l - \frac{1}{2}} + \delta_n \right), 0 \right] \quad (9.60)$$

and

$$T_{a,k} = \sum_{l=k_{\text{max}}+1}^{k_{\text{max}}+l_{\text{max}}} T_{n,l} \max \left[ \min \left( z_{k + \frac{1}{2}} + \delta_o, z_{l + \frac{1}{2}} + \delta_n \right) 
- \max \left( z_{k - \frac{1}{2}} + \delta_o, z_{l - \frac{1}{2}} + \delta_n \right), 0 \right]. \quad (9.61)$$

Note that equation (9.59) correctly evaluates the temperature of the adjacent temperature cell as equal to the temperature of the examined cell if the adjacent temperature mesh does not extend to the absolute axial location of the examined cell.

### Vapour Surface and Internal Boundary Conditions

In solving the thermal energy transport equation (9.50), the vapour surface and internal boundary condition equations must also be discretised. The internal boundary condition equation (8.59) is discretised as

$$T_{o,k_{\text{max}}+l_{\text{max}}} = T_{l,i}, \quad (9.62)$$

where $T_{l,i}$ is the initial temperature of the liquid. In terms of the liquid heat transfer coefficients defined by equation (9.50), this internal boundary condition can be expressed as

$$a_{k_{\text{max}}+l_{\text{max}}} = 0, \quad \beta_{k_{\text{max}}+l_{\text{max}}} = 1, \quad \gamma_{k_{\text{max}}+l_{\text{max}}} = 0 \quad \text{and} \quad \theta_{k_{\text{max}}+l_{\text{max}}} = T_{l,i}. \quad (9.63)$$
For the vapour surface boundary condition, we use the same indirect method of solution that was implemented within the solid phase. Thus, we first define the linear relationship
\[ \frac{\partial T_l}{\partial z'} (r_o, 0) = I_{l,o} \left( T_{ll,o} + J_{l,o} \right), \] (9.64)
where \( I_{l,o} \) and \( J_{l,o} \) are liquid phase heat transfer constants within subcell \( o \), and \( T_{ll,o} = T_l (r_o, 0) \) is the temperature of the liquid within subcell \( o \) at the liquid-vapour interface.

Defining
\[ \xi_{l,k} = \frac{\partial T_{o,k}}{\partial T_{ll,o}} \frac{\partial T_{o,k-1}}{\partial T_{ll,o}} - 1 \frac{\partial T_{ll,o}}{\partial T_{o,k}}, \] (9.65)
we can determine the magnitude of
\[ Z_{l,o} = \xi_{l,k_{\text{max}}+2} = \frac{\partial T_{o,k_{\text{max}}+2}}{\partial T_{ll,o}} \frac{\partial T_{o,k_{\text{max}}+1}}{\partial T_{ll,o}} \] (9.66)
by performing a recursion analysis on
\[ \xi_{l,k} = \frac{-\alpha_k}{\beta_k + \gamma_k \xi_{s,k+1}}, \] (9.67)
noting that the boundary condition equation (9.62) specifies
\[ \xi_{l,l_{\text{max}}} = 0. \] (9.68)
Equation (9.67) was derived from the liquid thermal energy transport equation (9.50), and as such, the thermal energy equation coefficients in this equation refer to coefficients evaluated for the liquid phase.

The temperature of the liquid at the vapour interface is defined as,
\[ T_{ll,o} = T_l (r_o, 0) = \frac{T_{o,k_{\text{max}}+1} + T_{o,k_{\text{max}}+2}}{2}. \] (9.69)
Differentiating equation (9.69) with respect to \( T_{ll,o} \), and substituting \( Z_{l,o} \) from equation (9.66) yields
\[ \frac{\partial T_{o,k_{\text{max}}+1}}{\partial T_{ll,o}} = \frac{2}{1 + Z_{l,o}} \] (9.70)
and

\[
\frac{\partial T_{o,k_{\text{max}}+2}}{\partial T_{i,o}} = \frac{2Z_{l,o}}{1 + Z_{l,o}}. \tag{9.71}
\]

Also, the temperature gradient of the liquid at the vapour interface is defined as,

\[
\frac{\partial T_{l}}{\partial z}(r_o, 0) = \frac{T_{o,k_{\text{max}}+2} - T_{o,k_{\text{max}}+1}}{\delta z_{l,{\text{min}}}}. \tag{9.72}
\]

Differentiating this equation with respect to \( T_{l,o} \), and substituting in equations (9.70) and (9.71), gives the first liquid phase heat transfer constant as

\[
J_{l,o} = \frac{2}{\delta z_{l,{\text{min}}}} \left( \frac{Z_{l,o} - 1}{Z_{l,o} + 1} \right). \tag{9.73}
\]

The second liquid phase heat transfer constant, \( J_{l,o} \), is calculated using a similar method used to calculate the second solid phase heat transfer constant. By evaluating the temperature distribution within the liquid phase using equations (9.50) and (9.63), while assuming

\[
\frac{\partial T_{l}}{\partial z'}(r_o, 0) = 0, \tag{9.74}
\]

we have from equation (9.64),

\[
J_{l,o} = -T_{l,o}. \tag{9.75}
\]

Substituting equation (9.74) into equation (9.72) gives the front surface boundary condition required for this procedure,

\[
T_{o,k_{\text{max}}+2} - T_{o,k_{\text{max}}+1} = 0. \tag{9.76}
\]

In terms of the solid heat transfer coefficients defined by equation (9.50), this front boundary condition can be expressed as

\[
a_{k_{\text{max}}+1} = 0, \quad \beta_{k_{\text{max}}+1} = -1, \quad \gamma_{k_{\text{max}}+1} = 1 \quad \text{and} \quad \theta_{k_{\text{max}}+1} = 0. \tag{9.77}
\]

Note that the liquid surface temperature, required for the evaluation of \( J_{l,o} \), is given by equation (9.69) as a function of the computational node temperatures.
Radial Boundary Conditions

In BOUNCE, it is assumed that no heat enters or leaves the droplet in the radial direction. Consequently, the radial temperature gradient within the droplet, as defined by equation \( \text{eqn} \), is set to zero if

1. a temperature node which is adjacent to an examined node is not contained within a fluid region, and

2. the fluid velocity at the examined node is directed from the adjacent void region node towards the examined node.

As the radius of a simulated droplet may change throughout an impact, a column of liquid temperature nodes may or may not be contained within a fluid region at any given instant. The thermal energy transport equation is only solved on columns of temperature nodes which are contained within fluid regions of the droplet. Consequently, when the region in which a column of temperature nodes exists changes from a fluid to a void region, the column of temperature nodes no longer takes part in the droplet temperature calculation, and the temperatures of that column remain constant until fluid again surrounds the nodes.

Note that typically when a column of temperature nodes enters or exits a fluid region, the vapour layer height corresponding to that column is large, and axial conduction ensures that the temperatures of the nodes are close to the ambient temperature. Thus, the total thermal energy conservation error caused by changing the number of temperature nodes involved in the thermal energy calculation is small.
Final Liquid Phase Temperature Distribution

Once the vapour layer iterations are complete and the temperature of the liquid is known for each subcell of the vapour layer grid, the thermal energy equations specified by equation (9.50), combined with the boundary condition equations (9.62) and (9.69), are solved using a tridiagonal matrix inverter to find the final temperature distribution within the liquid phase. Integrity tests of the liquid phase heat transfer algorithm are presented in Appendix D.

9.3.3 Solid-Vapour Interface

In this section we outline the procedure used by BOUNCE to solve the equations developed in the previous chapter, Section 8.4.4, for the solid-vapour interface conditions.

Energy Balance

Equation (8.64) can be discretised to yield

$$k_s \frac{\partial T_s}{\partial z} (r_o, 0) = \frac{k_{m,o}}{\delta_o} (T_{lv,o} - T_{sv,o}) , \quad (9.78)$$

where $k_{m,o}$ is the average thermal conductivity of the vapour mixture within vapour layer subcell $o$, and $T_{lv,o}$ and $T_{sv,o}$ are the average vapour mixture temperatures within subcell $o$ at the liquid and solid interfaces, respectively. Equation (9.25) can be used to define the solid temperature gradient at the solid-vapour interface, thus giving the energy balance at the solid-vapour interface as

$$k_s I_{s,o} (T_{ss,o} + J_{s,o}) = \frac{k_{m,o}}{\delta_o} (T_{lv,o} - T_{sv,o}) . \quad (9.79)$$
Interface Conditions

In a similar manner, equation (8.66) can be discretised to yield the interface condition equation at the solid-vapour interface as

\[
T_{sv,o} - T_{ss,o} = \lambda_{m,s,o} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{l,s}} - 1 \right) (T_{lv,o} - T_{sv,o}).
\]

(9.80)

The average mean free path for the vapour mixture in the vicinity of the solid-vapour interface, \( \lambda_{m,s,o} \), is calculated using equation (A.49) from Appendix A. The temperature used in this calculation is that of the vapour at the solid-vapour interface, \( T_{sv,o}^* \), the asterisk indicating that this value is taken from the most recent vapour layer iteration.

9.3.4 Liquid-Vapour Interface

In this section we outline the procedure used by BOUNCE to solve the equations developed in the previous chapter, Section 8.4.5, for the liquid-vapour interface conditions.

Energy Balance

Equation (8.69) is used to perform an energy balance at the liquid vapour interface. It may be discretised to yield

\[
\frac{k_{m,o}}{\delta_o} (T_{sv,o} - T_{lv,o}) = -k_l \frac{\partial T_l}{\partial z'} (r_o, 0) + \dot{w}_o \rho_v H_{fg,c},
\]

(9.81)

where the latent heat of vaporisation corrected for sensible heat, \( H_{fg,c} \), is defined by equation (8.68). The gradient of the liquid at the vapour interface can be substituted into equation (9.81) from equation (9.64) to give

\[
\frac{k_{m,o}}{\delta_o} (T_{sv,o} - T_{lv,o}) = -k_l J_{l,o} (T_{lv,o} + J_{l,o}) + \dot{w}_o \rho_v H_{fg,c}.
\]

(9.82)
Mass Balance

Equation (8.70) expresses the droplet vaporisation velocity at the liquid-vapour interface in terms of molecular fluxes to and from the liquid surface. This equation can be discretised as

$$\dot{w}_o = \frac{\sigma_{tl}}{\rho_v} \sqrt{\frac{M_v}{2\pi R}} \left( \frac{P_{sat}(T_{ll,o})}{\sqrt{T_{ll,o}}} - \frac{P_{v,o}}{\sqrt{T_{lv,o}}} \right),$$  \hspace{1cm} (9.83)

where $P_{v,o}$ is the absolute partial vapour pressure within subcell $o$ of the vapour layer, and the values of variables marked with an asterisk are taken from the most recent vapour layer iteration. Since vapour layer pressures are stored as gauge pressures, the absolute vapour pressure is defined using equations (8.25) and (8.27) as

$$P_{v,o} = (1 - y_{a,o}) (P_o + P_{atm})$$  \hspace{1cm} (9.84)

where $P_{atm}$ is atmospheric pressure.

As discussed in Appendix C, the saturation curve for liquids relevant to this study can be reasonably well approximated over a wide range of pressures by the Antoine equation

$$T_{sat}(P) = A_{sat} + \frac{B_{sat}}{\ln P + C_{sat}},$$  \hspace{1cm} (9.85)

where $A_{sat}$, $B_{sat}$ and $C_{sat}$ are constants specific to the liquid being modelled. Equation (9.85) can be rearranged to give the saturation pressure as a function of temperature,

$$P_{sat}(T) = \exp \left( \frac{B_{sat}}{T - A_{sat} - C_{sat}} \right).$$  \hspace{1cm} (9.86)

The saturation pressure from equation (9.86) could be substituted directly into equation (9.83) to give the droplet vaporisation rate as a function of the liquid temperature at the vapour interface. However, such a procedure
would give rise to a non-linear equation for the vaporisation velocity, which is not easily solved using matrix inversion methods. Consequently, before substitution into equation (9.83), equation (9.86) is linearised into the form

$$P_{\text{sat}}(T) = \frac{T - L_{\text{sat}}}{K_{\text{sat}}}$$  (9.87)

where the saturation constants $K_{\text{sat}}$ and $L_{\text{sat}}$ are dependent on the liquid temperature at the vapour interface from the last vapour layer iteration, $T_{ll,o}^*$. 

In obtaining values for the saturation constants $K_{\text{sat}}$ and $L_{\text{sat}}$, we employ the two requirements

$$\left. \frac{dP_{\text{sat}}}{dT} \right|_{\text{non-linear}} = \left. \frac{dP_{\text{sat}}}{dT} \right|_{\text{linear}}$$  (9.88)

and

$$\left. P_{\text{sat}}(T_{ll,o}^*) \right|_{\text{non-linear}} = \left. P_{\text{sat}}(T_{ll,o}^*) \right|_{\text{linear}}.$$  (9.89)

Substituting equations (9.86) and (9.87) into equation (9.88) and differentiating yields

$$K_{\text{sat}} = \frac{- (T_{ll,o}^* - A_{\text{sat}})^2}{B_{\text{sat}} \exp \left( \frac{B_{\text{sat}}}{T_{ll,o}^* - A_{\text{sat}} - C_{\text{sat}}} \right)}.$$  (9.90)

Similarly, substituting equations (9.86) and (9.87) into equation (9.89) yields

$$L_{\text{sat}} = T_{ll,o}^* - K_{\text{sat}} \exp \left( \frac{B_{\text{sat}}}{T_{ll,o}^* - A_{\text{sat}} - C_{\text{sat}}} \right).$$  (9.91)

Substituting equations (9.84) and (9.87) into equation (9.83) gives the final linear vaporisation velocity equation in terms of the subcell pressure and liquid temperature at the vapour interface,

$$\dot{w}_o = \frac{\sigma_{l,t}}{\rho_v} \sqrt{\frac{M_v}{2\pi R}} \left( \frac{T_{ll,o} - L_{\text{sat}}}{K_{\text{sat}} \sqrt{T_{ll,o}}} - \frac{(1 - y_{n,o}) (P_v + P_{\text{atm}})}{\sqrt{T_{lv,o}}} \right).$$  (9.92)
Interface Conditions

Equation \((8.71)\) of the previous chapter gives the relationship between temperatures on either side of the liquid-vapour interface. Applying this equation to a vapour layer subcell yields

\[
T_{lv,o} - T_{ll,o} = -\frac{\lambda_{m,l,o}}{\phi_o \delta_o} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,l}} - 1 \right) (T_{lv,o} - T_{sv,o}).
\] (9.93)

The average mean free path for the vapour mixture in the vicinity of the liquid-vapour interface, \(\lambda_{m,l,o}\), is calculated using equation \((A.49)\) from Appendix \(A\). The temperature used in this calculation is that of the vapour at the solid-vapour interface, \(T_{lv,o}^*\).

The ratio of condensation and evaporation rates, \(\phi_o\), as used in equation \((9.93)\) is defined using equation \((8.73)\). In discretised form, this equation becomes

\[
\phi_o = \frac{\dot{w}_o * \rho_v \sqrt{T_{lv}^*}}{\sigma_{t,l} (1 - y_{a,o}^*) (P_o^* + P_{atm}) \sqrt{2\pi R M_v} + 1},
\] (9.94)

where as previously, variables marked with an asterisk are taken from the most recent vapour layer iteration.

9.3.5 Heat Transfer Simplifications

In this section the heat transfer simplifications outlined in Section \(8.4.6\) are applied to the vapour layer computational procedure.

Interface Molecular Treatment

If the molecular treatment at both liquid-vapour and solid-vapour interfaces is neglected, then equation \((9.80)\) is replaced by the discretised form of equation \((8.74)\),

\[
T_{ss,o} = T_{sv,o}.
\] (9.95)
and equation (9.93) is replaced by the discretised form of equation (8.75),

\[ T_{ll,o} = T_{lv,o}. \quad (9.96) \]

Also, the vaporisation velocity is no longer specified by equation (9.92). Instead, the discretised form of equation (8.76), utilising equations (9.87) and (9.84),

\[ T_{ll,o} = T_{sat} (P_{v,o}) = K_{sat} (1 - y_{a,o}) (P_o + P_{atm}) + L_{sat}, \quad (9.97) \]

is used to close the vaporisation heat transfer solution.

**Solid Heat Transfer**

If the temperature of the solid is to be approximated as uniform and constant, then equation (9.79) is replaced by the discretised form of equation (8.77),

\[ T_{ss,o} = T_{s,i}. \quad (9.98) \]

**9.3.6 Vaporisation Velocity Solution Procedure**

The droplet vaporisation velocity can be calculated by simultaneously solving equations (9.79), (9.80), (9.82), (9.92) and (9.93) when the full vapour layer heat transfer model is specified. If heat transfer simplifications are employed, then the equations to be solved are modified, as detailed in the previous section. To aid in the solution of this complex set of equations, we define five general equations, which, depending on the values chosen for their coefficients, can represent the vapour layer heat transfer model with some, none, or all of the heat transfer simplifications employed.

The energy conservation equation at the solid-vapour interface is defined as

\[ T_{sv,o} = c_1 T_{ss,o} + c_2 T_{lv,o} + c_3, \quad (9.99) \]
where from equation (9.79) for the full heat transfer model, the coefficients are defined as

\[ c_1 = -\frac{k_s L_{s,o} \delta_o}{k_{m,o}}, \]  
\[ (9.100) \]

\[ c_2 = 1 \]  
\[ (9.101) \]

and

\[ c_3 = c_1 L_{s,o}. \]  
\[ (9.102) \]

If the temperature of the solid is assumed constant and uniform, then the coefficients are specified by the simplified equation (9.98) as

\[ c_1 = 0, \]  
\[ (9.103) \]

\[ c_2 = 0 \]  
\[ (9.104) \]

and

\[ c_3 = T_{s,i}. \]  
\[ (9.105) \]

The second general heat transfer equation is the solid-vapour interface condition equation, defined as

\[ T_{sv,o} - T_{ss,o} = c_4 (T_{lv,o} - T_{sv,o}). \]  
\[ (9.106) \]

For the full heat transfer model, the coefficient for this equation is defined from equation (9.80) as

\[ c_4 = \frac{\lambda_{m,s,o}}{\delta_o} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,s}} - 1 \right). \]  
\[ (9.107) \]

If the molecular interface treatment is neglected, then \( c_4 \) is defined from equation (9.95) as

\[ c_4 = 0. \]  
\[ (9.108) \]
The general equation for energy conservation at the liquid-vapour interface is defined as

\[ T_{sv,o} - T_{lv,o} = c_5 T_{ll,o} + c_6 \dot{w}_o + c_7. \]  

(9.109)

The coefficients for this equation are defined from equation (9.82) as

\[ c_5 = -\frac{k_l l_d \hat{\delta}_o}{k_{m,o}}, \]  

(9.110)

\[ c_6 = \frac{H_{fg,c} \rho_v \hat{\delta}_o}{k_{m,o}} \]  

(9.111)

and

\[ c_7 = c_5 J_{l,o}. \]  

(9.112)

Similarly, the general heat transfer equation representing conditions at the liquid-vapour interface is defined as

\[ T_{lv,o} - T_{ll,o} = c_8 (T_{lv,o} - T_{sv,o}), \]  

(9.113)

where the coefficient is defined using equation (9.93) as

\[ c_8 = -\frac{\lambda_{m,l,o}}{\phi_o \delta_o} \left( \frac{9 \gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_{t,l}} - 1 \right). \]  

(9.114)

Alternatively, if the interface molecular treatment is neglected, \( c_8 \) is given by equation (9.96) as

\[ c_8 = 0. \]  

(9.115)

Finally, the general mass balance equation at the liquid-vapour interface is defined as

\[ T_{ll,o} = c_9 \dot{w}_o + c_{10} P_o + c_{11}. \]  

(9.116)

Using equation (9.92), the coefficients in equation (9.116) are given by

\[ c_9 = K_{sat} \sqrt{T_{ll,o} \rho_v} \frac{\sqrt{2 \pi R}}{M_v}, \]  

(9.117)
\[ c_{10} = K_{\text{sat}} \sqrt{T_{\text{ll,o}}^*} \frac{(1 - y_{a,o})}{T_{\text{lv,o}}^*} \]  

(9.118)

and

\[ c_{11} = c_{10} P_{\text{atm}} + L_{\text{sat}}. \]  

(9.119)

If the molecular interface treatment is neglected, then the mass balance coefficients are instead given by

\[ c_9 = 0 \]  

(9.120)

and

\[ c_{10} = K_{\text{sat}} (1 - y_{a,o}), \]  

(9.121)

while \( c_{11} \) remains unchanged.

Once the general heat transfer coefficients defined above have been determined, the vaporisation velocity is calculated by the simultaneous solution of the five general heat transfer equations, \([9.99], [9.106], [9.109], [9.113]\) and \([9.116]\). This results in an expression for the vaporisation velocity which is a linear function of the vapour layer pressure, or equation \([9.1]\). This equation is repeated here,

\[ \dot{w}_o = \alpha_o P_o + \beta_o, \]  

(9.122)

and the coefficients are defined from the solution to the five general heat transfer equations as

\[ \alpha_o = \frac{c_{10}}{c_{14}} \]  

(9.123)

and

\[ \beta_o = \frac{c_{11} - c_{15}}{c_{14}}. \]  

(9.124)

The new coefficients in equations \([9.123]\) and \([9.124]\) are defined as

\[ c_{12} = 1 - c_1 - c_1 c_4, \]  

(9.125)
\[ c_{13} = \frac{c_2 - c_1 c_4 - c_{12}}{c_{12} (1 - c_8) + c_8 (c_2 - c_1 c_4)}, \quad (9.126) \]

\[ c_{14} = \frac{c_6}{c_{13} - c_5} - c_9 \quad (9.127) \]

and

\[ c_{15} = \frac{c_8 c_3 c_{13} + c_{12} c_7 - c_3}{c_{12} (c_{13} - c_5)}. \quad (9.128) \]

Back substitution into the general heat transfer equations gives the other vapour layer subcell variables. The temperature of the liquid at the liquid-vapour interface is given by

\[ T_{ll,o} = K_o \dot{w} + L_o, \quad (9.129) \]

where

\[ K_o = \frac{c_6}{c_{13} - c_5} \quad (9.130) \]

and

\[ L_o = c_{15}. \quad (9.131) \]

The temperature of the vapour at the liquid-vapour interface is given by

\[ T_{lv,o} = M_o T_{ll,o} + N_o, \quad (9.132) \]

where

\[ M_o = \frac{c_{12}}{c_{12} (1 - c_8) + c_8 (c_2 - c_1 c_4)} \quad (9.133) \]

and

\[ M_o = -\frac{c_8 c_3}{c_{12} N_o}. \quad (9.134) \]

Similarly, the temperature of the vapour at the solid-vapour interface is given by

\[ T_{sv,o} = Q_o T_{lv,o} + P_o, \quad (9.135) \]
where
\[ Q_o = \frac{c_2 - c_1c_4}{c_{12}} \]  
(9.136)
and
\[ P_o = \frac{c_3}{c_{12}} \]  
(9.137)
and the temperature of the solid at the solid-vapour interface is given by
\[ T_{ss,o} = Q_o T_{sv,o} + R_o T_{lv,o} \]  
(9.138)
where
\[ Q_o = 1 + c_4 \]  
(9.139)
and
\[ R_o = -c_4. \]  
(9.140)

9.4 Vapour Layer Flow Solution

In this section we outline the computational method used to calculate the pressures and flowrates existing within the vapour layer. The calculation is performed by solving the vapour layer momentum and mass balances, which were developed in Section 8.2 of the previous chapter. It is assumed here that the droplet vaporisation velocity is specified as a linear function of the vapour layer pressure, as given by equation (9.1).

The procedure used to solve the vapour flow is noteworthy in that equation (8.23) for the vapour mixture mass balance, and equation (8.15) for the momentum balance, are combined to give a series of linear expressions which relate average subcell pressures. The coefficients of these linear expressions can be expressed in the form of a tridiagonal matrix, and inverted using the algorithm discussed in Section 9.1.2.
9.4.1 Linear Pressure Relationship

A general expression relating the average pressures of three adjacent subcells is now developed.

Mass Conservation

We first note that the mass conservation equation (8.23) can be expressed as

\[ S(r' + \Delta r') = S(r') + 2\pi \int_{r'}^{r'+\delta r'} \left( \dot{\psi}(r) - \frac{\partial \delta}{\partial t}(r) \right) rdr, \quad (9.141) \]

where \( \delta r' \) is a finite radial increment.

Integrating equation (9.141) over subcell \( o \), assuming a constant vaporisation and vapour layer height velocity within the subcell gives,

\[ S(r) = S_{o-\frac{1}{2}} + \pi C_o \left( r^2 - r_{o-\frac{1}{2}}^2 \right), \quad (9.142) \]

where \( S(r) \) is the vapour mixture flow rate at location \( r \) within subcell \( o \), the boundary condition \( S\left(r_{o-\frac{1}{2}}\right) = S_{o-\frac{1}{2}} \) has been used, and the constant

\[ C_n = \dot{w}_n - \frac{\partial \delta}{\partial t} \quad (9.143) \]

has been defined as the absolute vapour velocity at the lower surface of the droplet. Repeating the integration of equation (9.141) over subcell \( o - 1 \), using the boundary condition, \( S\left(r_{o-\frac{1}{2}}\right) = S_{o-\frac{1}{2}} \), gives

\[ S(r) = S_{o-\frac{1}{2}} + \pi C_{o-1} \left( r^2 - r_{o-\frac{1}{2}}^2 \right), \quad (9.144) \]

which is valid within subcell \( o - 1 \). The same integration over subcell \( o + 1 \), using the boundary condition \( S\left(r_{o+\frac{1}{2}}\right) = S_{o+\frac{1}{2}} \), gives

\[ S(r) = S_{o+\frac{1}{2}} + \pi C_{o+1} \left( r^2 - r_{o+\frac{1}{2}}^2 \right), \quad (9.145) \]

237
which is valid within subcell \( o+1 \). An expression relating the vapour flowrates on both sides of subcell \( o \) is found by evaluating equation \((9.142)\) at the location \( r = r_{o+\frac{1}{2}} \). Thus,

\[
S_{o+\frac{1}{2}} = S_{o-\frac{1}{2}} + \pi C_o \left( r_{o+\frac{1}{2}}^2 - r_{o-\frac{1}{2}}^2 \right). \tag{9.146}
\]

**Momentum Conservation**

Equations \((9.142)\)–\((9.146)\) provide a relationship between the vapour flowrates within three adjacent subcells, \( o-1, o \) and \( o+1 \), and the vapour flowrates at the two included borders between these computational subcells. In a similar fashion, we can develop expressions between the average pressures and border pressures of the three adjacent subcells, \( o-1, o \) and \( o+1 \), by integrating the momentum equation \((8.15)\).

The momentum equation \((8.15)\) can be expressed as

\[
\frac{dP}{dr}(r) = -\frac{A_o}{\pi r} S(r) + G_o \tag{9.147}
\]

If the vapour layer height, \( \delta \), droplet lower surface fluid velocity, \( u_l \), and molecular interface variables are assumed constant within each subcell, then equation \((9.147)\) defines the pressure within subcell \( o \) as,

\[
\frac{dP}{dr}(r) = -\frac{A_n}{\pi r} S(r) + G_o \tag{9.148}
\]

where the above constants are defined by

\[
A_n = -\mu_{m,n} \left\{ \frac{2\hat{\delta}_n}{\delta_n} \left[ \frac{\hat{\theta}_{l,n}}{6} - \frac{\hat{\delta}_n}{2} \left( \frac{\hat{\theta}_{l,n} + \hat{\delta}_n}{2} \right) \left( \frac{\hat{\theta}_{s,n} + \hat{\delta}_n}{2} \right) \right] \right\}, \tag{9.149}
\]

238
and

\[ G_n = \frac{2A_n \delta_n u_{t,n} \left( \theta_{s,n} + \frac{\delta_n}{2} \right)}{\left( \theta_{s,n} + \theta_{t,n} + \frac{\delta_n}{2} \right)}. \] (9.150)

In equations (9.149) and (9.150), the subscript \( n \) specifies properties that are averaged over the dimensions of subcell \( n \). Substituting equation (9.142) for the vapour flowrate into equation (9.148) and integrating, noting the subcell border boundary condition

\[ P \left( r_{o-\frac{1}{2}} \right) = P_{o-\frac{1}{2}}, \] (9.151)

gives

\[
P(r) = P_{o-\frac{1}{2}} - A_o \left( \frac{S_{o-\frac{1}{2}}}{\pi} - C_o \frac{r^2}{2} \right) \ln \left( \frac{r}{r_{o-\frac{1}{2}}} \right) - \frac{A_o C_o}{2} \left( r^2 - r_{o-\frac{1}{2}}^2 \right) + G_o \left( r - r_{o-\frac{1}{2}} \right), \] (9.152)

which is valid within subcell \( o \).

The average pressure acting on the lower surface of the droplet within subcell \( o \) is defined as

\[ P_o = \frac{1}{\pi \left( \frac{r_{o+\frac{1}{2}}^2 - r_{o-\frac{1}{2}}^2}{2} \right)} \int_{r_{o-\frac{1}{2}}}^{r_{o+\frac{1}{2}}} 2\pi P(r) dr. \] (9.153)

Substituting equation (9.152) into equation (9.153), and integrating, gives the average pressure within subcell \( o \) as,

\[ P_o = P_{o-\frac{1}{2}} + A_o \left[ \left( \frac{S_{o-\frac{1}{2}}}{\pi} - C_o \right) \left( \ln r_{o-\frac{1}{2}} - \frac{a_o}{b_o} \right) \right] + \frac{C_o}{2} \left( r_{o-\frac{1}{2}}^2 - \frac{c_o}{2b_o} \right) + G_o \left( \frac{2d_o}{3b_o} - r_{o-\frac{1}{2}} \right), \] (9.154)

where the geometric constants within subcell \( n \) have been defined as

\[ a_n = \left( \ln r_{n+\frac{1}{2}} - \frac{1}{2} \right) r_{n+\frac{1}{2}} - \left( \ln r_{n-\frac{1}{2}} - \frac{1}{2} \right) r_{n-\frac{1}{2}}^2. \] (9.155)
\[ b_n = r_{n+\frac{1}{2}}^2 - r_{n-\frac{1}{2}}^2, \quad (9.156) \]

\[ c_n = r_{n+\frac{1}{2}}^4 - r_{n-\frac{1}{2}}^4, \quad (9.157) \]

and

\[ d_n = r_{n+\frac{1}{2}}^3 - r_{n-\frac{1}{2}}^3, \quad (9.158) \]

Equation (9.154) expresses the average vapour layer pressure within subcell \( o \) as a function of the vapour layer pressure existing at the left border of the subcell, \( P_{o-\frac{1}{2}} \), the vapour layer flowrate at the left border of the subcell, \( S_{o-\frac{1}{2}} \), and the geometry and properties of the examined subcell.

By using a similar analysis to the above, but by using the boundary condition at the right rather than left border of subcell \( o \) when integrating equation (9.148),

\[ P\left(r_{o+\frac{1}{2}}\right) = P_{o+\frac{1}{2}}, \quad (9.159) \]

the average pressure within subcell \( o \) can be expressed as,

\[
P_o = P_{o+\frac{1}{2}} + A_o \left[ \left( \frac{S_{o-\frac{1}{2}}}{2} - r_{o-\frac{1}{2}} C_o \right) \ln r_{o+\frac{1}{2}} - \frac{a_o}{b_o} \right]
+ \frac{C_o}{2} \left( r_{o+\frac{1}{2}} - \frac{c_o}{2b_o} \right) + G_o \left( \frac{2d_o}{3b_o} - r_{o+\frac{1}{2}} \right). \quad (9.160)
\]

Unlike equation (9.154), this expression involves the vapour layer pressure at the right border of subcell \( o \).

The analysis given by equations (9.148)–(9.158) can be repeated in the subcells adjacent to subcell \( o \). In subcell \( o-1 \), the vapour mixture flowrate of equation (9.144) can be combined with the momentum equation (9.148) evaluated in subcell \( o-1 \). Integrating, using the boundary condition

\[ P\left(r_{o-\frac{1}{2}}\right) = P_{o-\frac{1}{2}}, \quad (9.161) \]
gives the pressure within subcell $o - 1$ as

$$P(r) = P_{o-\frac{1}{2}} - A_{o-1} \left( \frac{S_{o-\frac{1}{2}}}{\pi} - C_{o-1} r_{o-\frac{1}{2}}^2 \right) \ln \left( \frac{r}{r_{o-\frac{1}{2}}} \right)$$

$$- \frac{A_{o-1} C_{o-1}}{2} \left( r^2 - r_{o-\frac{1}{2}}^2 \right) + C_{o-1} \left( r - r_{o-\frac{1}{2}} \right).$$

(9.162)

Substituting equation (9.162) into equation (9.153) and integrating gives the average pressure within subcell $o - 1$ as,

$$P_{o-1} = P_{o-\frac{1}{2}} + A_{o-1} \left[ \left( \frac{S_{o-\frac{1}{2}}}{\pi} - r_{o-\frac{1}{2}}^2 C_{o-1} \right) \left( \ln r_{o-\frac{1}{2}} - \frac{a_{o-1}}{b_{o-1}} \right) \right]$$

$$+ \frac{C_{o-1}}{2} \left( r_{o-\frac{1}{2}} - \frac{c_{o-1}}{2b_{o-1}} \right) + C_{o-1} \left( 2d_{o-1} - 3b_{o-1} - r_{o-\frac{1}{2}} \right).$$

(9.163)

Similarly, in subcell $o + 1$, the vapour mixture flowrate of equation (9.145) can be combined with the momentum equation (9.148) evaluated in subcell $o + 1$. Integrating, using the boundary condition

$$P \left( r_{o+\frac{1}{2}} \right) = P_{o+\frac{1}{2}},$$

(9.164)

gives the pressure within subcell $o + 1$ as

$$P(r) = P_{o+\frac{1}{2}} - A_{o+1} \left( \frac{S_{o+\frac{1}{2}}}{\pi} - r_{o+\frac{1}{2}}^2 C_{o+1} \right) \ln \left( \frac{r}{r_{o+\frac{1}{2}}} \right)$$

$$- \frac{A_{o+1} C_{o+1}}{2} \left( r^2 - r_{o+\frac{1}{2}}^2 \right) + C_{o+1} \left( r - r_{o+\frac{1}{2}} \right).$$

(9.165)

Substituting equation (9.162) into equation (9.153) and integrating gives the average pressure within subcell $o + 1$ as,

$$P_{o+1} = P_{o+\frac{1}{2}} + A_{o+1} \left[ \left( \frac{S_{o+\frac{1}{2}}}{\pi} - r_{o+\frac{1}{2}}^2 C_{o+1} \right) \left( \ln r_{o+\frac{1}{2}} - \frac{a_{o+1}}{b_{o+1}} \right) \right]$$

$$+ \frac{C_{o+1}}{2} \left( r_{o+\frac{1}{2}} - \frac{c_{o+1}}{2b_{o+1}} \right) + C_{o+1} \left( 2d_{o+1} - 3b_{o+1} - r_{o+\frac{1}{2}} \right).$$

(9.166)

The pressure equations (9.154), (9.160), (9.163) and (9.166), combined with the vapour flowrate equation (9.146), provide a relationship between
the seven variables, \( P_{o-1} \), \( P_{o-\frac{1}{2}} \), \( P_o \), \( P_{o+\frac{1}{2}} \), \( P_{o+1} \), \( S_{o-\frac{1}{2}} \) and \( S_{o+\frac{1}{2}} \). The equations can be solved simultaneously to find an expression relating the average pressures of the three adjacent subcells, \( o - 1 \), \( o \) and \( o + 1 \), to the absolute lower droplet surface vapour velocity constants \( C_o \), giving,

\[
B_{1,o}P_{o-1} - (B_{2,o} + B_{1,o}) P_o + B_{2,o}P_{o+1} = C_{o-1}D_{1,o} + C_oD_{2,o} + C_{o+1}D_{3,o} + D_{8,o}
\]  

(9.167)

where the following variables are defined as

\[
B_{1,n} = A_{n-1}e^{-\frac{1}{2},n-1} - A_{n}e^{-\frac{1}{2},n};
\]  

(9.168)

\[
B_{2,n} = A_{n}e^{\frac{1}{2},n} - A_{n+1}e^{\frac{1}{2},n+1};
\]  

(9.169)

\[
D_{1,n} = A_{n-1}B_{2,n} \left( \frac{1}{2}f_{n-\frac{1}{2},n-1} - g_{n-\frac{1}{2},n-1} \right);
\]  

(9.170)

\[
D_{2,n} = A_{n} \left[ B_{2,n} \left( g_{n-\frac{1}{2},n} - \frac{1}{2}f_{n-\frac{1}{2},n} \right) + B_{1,n} \left( g_{n+\frac{1}{2},n} - \frac{1}{2}f_{n+\frac{1}{2},n} \right) \right]
\]  

\[- B_{1,n}B_{2,n}b_{n},
\]  

(9.171)

\[
D_{3,n} = A_{n+1}B_{1,n} \left( \frac{1}{2}f_{n+\frac{1}{2},n+1} - g_{n+\frac{1}{2},n+1} \right)
\]  

(9.172)

and

\[
D_{8,n} = B_{2,n} \left( G_{n-1}h_{n-\frac{1}{2},n-1} - G_{n}h_{n-\frac{1}{2},n} \right) - B_{1,n} \left( G_{n}h_{n+\frac{1}{2},n} - G_{n+1}h_{n+\frac{1}{2},n+1} \right)
\]  

(9.173)

The new geometric constants in equations (9.168)–(9.172) are defined as

\[
f_{n,m} = r_{n} - \frac{c_{m}}{2b_{m}^{*}};
\]  

(9.174)
\[ g_{n,m} = r_n^2 \left( \ln r_n + \frac{a_m}{b_m} \right) \quad (9.175) \]

and

\[ h_{n,m} = \frac{2d_m}{3b_m} - r_n. \quad (9.176) \]

Combining equation (9.143) with equation (9.1) gives the absolute lower surface vapour velocity as

\[ C_n = \alpha_n P_n + \beta_n - \frac{\partial \bar{\delta}_n}{\partial t}. \quad (9.177) \]

Substitution into equation (9.167) gives the final expression,

\[ (B_{1,o} - \alpha_{o-1} D_{1,o}) P_{o-1} - (B_{2,o} + B_{1,o} + \alpha_o D_{2,o}) P_o + (B_{2,o} - \alpha_{o+1} D_{3,o}) P_{o+1} = \left( \beta_{o-1} - \frac{\partial \bar{\delta}_{o-1}}{\partial t} \right) D_{1,o} + \left( \beta_o - \frac{\partial \bar{\delta}_o}{\partial t} \right) D_{2,o} + \left( \beta_{o+1} - \frac{\partial \bar{\delta}_{o+1}}{\partial t} \right) D_{3,o} + D_{8,o}, \quad (9.178) \]

where the constants were as defined for equation (9.167).

### 9.4.2 Pressure Boundary Conditions

Equation (9.178) is a linear expression relating the average vapour layer pressure of a subcell, to those of its two adjacent neighbours. In order to calculate the absolute magnitude of these pressures throughout the vapour layer, boundary conditions on the average subcell pressures must be imposed at both the centreline and peripheral boundaries of the viscous vapour layer.

**Vapour Layer Centreline Boundary Condition**

At the centre of the viscous vapour layer, the vapour mixture flowrate is zero. Thus, noting that the first cell within the vapour layer computational
domain is \( o = 2 \),

\[
S_{\frac{3}{2}} = 0,
\]

(9.179)

and equation (9.146) gives the vapour mixture flowrate at \( r = r_{\frac{3}{2}} \) as

\[
S_{\frac{5}{2}} = \pi C_{2} \left( r_{\frac{5}{2}}^{2} - r_{\frac{3}{2}}^{2} \right).
\]

(9.180)

Equation (9.160), evaluated at subcell \( o = 2 \), gives the average vapour layer pressure within this cell as

\[
P_{2} = P_{\frac{5}{2}} + A_{2} \left[ -r_{\frac{3}{2}}^{2} C_{2} \left( \ln r_{\frac{3}{2}} - \frac{a_{2}}{b_{2}} \right) \right.
\]

\[
+ \frac{C_{2}^{2}}{2} \left( r_{\frac{5}{2}}^{2} - \frac{c_{2}}{2b_{2}} \right) \left. \right] + G_{2} \left( \frac{2d_{2}}{3b_{2}} - r_{\frac{5}{2}} \right),
\]

(9.181)

where use has been made of equation (9.179). Similarly, equation (9.166), evaluated at subcell \( o = 2 \), gives the average vapour layer pressure of subcell \( o = 3 \) as

\[
P_{3} = P_{\frac{5}{2}} + A_{3} \left[ \left( \frac{S_{\frac{5}{2}}^{2}}{\pi} - r_{\frac{3}{2}}^{2} C_{3} \right) \left( \ln r_{\frac{3}{2}} - \frac{a_{3}}{b_{3}} \right) \right.
\]

\[
+ \frac{C_{3}^{2}}{2} \left( r_{\frac{5}{2}}^{2} - \frac{c_{3}}{2b_{3}} \right) \left. \right] + G_{3} \left( \frac{2d_{3}}{3b_{3}} - r_{\frac{5}{2}} \right).
\]

(9.182)

Combining equations (9.180)–(9.182) gives the relationship between the average subcell pressures of cell \( o = 2 \) and \( o = 3 \) as

\[
(1 - \alpha_{2} D_{4}) P_{2} + (-1 - \alpha_{3} D_{5}) P_{3} =
\]

\[
\left( \beta_{2} - \frac{\partial \tilde{\delta}_{2}}{\partial t} \right) D_{4} + \left( \beta_{3} - \frac{\partial \tilde{\delta}_{3}}{\partial t} \right) D_{5} + D_{9},
\]

(9.183)

where

\[
D_{4} = \frac{1}{2} A_{2} f_{\frac{5}{2}, 2} - g_{\frac{5}{2}, 3} A_{3},
\]

(9.184)

\[
D_{5} = A_{3} B_{1, 3} \left( \frac{1}{2} f_{\frac{5}{2}, 3} - g_{\frac{5}{2}, 3} \right)
\]

(9.185)
and
\[ D_9 = G_2 h_{2.2} - G_3 h_{2.3}. \]  
(9.186)

Thus, the boundary condition at the centre of the vapour layer is specified.

Note that as the left boundary of the computational domain is the centreline of the vapour layer,
\[ P_1 = P_2. \]  
(9.187)

**Vapour Layer Periphery Boundary Condition**

The rightmost subcell contained within the viscous vapour layer is defined as the rightmost subcell within the computational domain which has a defined, non-zero vapour layer height. This peripheral subcell is numbered \( o = o_R \), as \( R \) represents the outer radius of the droplet.

The boundary condition applied at the vapour layer periphery is that the vapour layer pressure at the outer radius of the droplet is atmospheric. This assumption is justified as the vapour velocities within the vapour layer are small compared with the speed of sound in air. As vapour layer pressures are gauge pressures, this boundary condition can be expressed as
\[ P_{o_R + \frac{1}{2}} = 0, \]  
(9.188)

since \( R \approx r_{o_R + \frac{1}{2}} \).

Applying the pressure equation \( (9.160) \) to subcell \( o = o_R \), noting the boundary condition \( (9.188) \), gives
\[ P_{o_R} = A_{o_R} \left[ \left( \frac{S_{o_R - \frac{1}{2}}}{\pi} - r_{o_R - \frac{1}{2}}^2 \right) C_{o_R} \left( \ln r_{o_R + \frac{1}{2}} - \frac{a_{o_R}}{b_{o_R}} \right) \right. \\
+ \frac{C_{o_R}}{2} \left( r_{o_R + \frac{1}{2}} - \frac{c_{o_R}}{2b_{o_R}} \right) \left. + G_{o_R} \left( \frac{2d_{o_R}}{3b_{o_R}} - r_{o_R + \frac{1}{2}} \right) \right]. \]  
(9.189)
Similarly, applying equations (9.154) and (9.163) to subcell \( o = o_R \) gives, respectively,

\[
P_{oR} = P_{oR-\frac{1}{2}} + A_{oR} \left[ \left( \frac{S_{oR-\frac{1}{2}}}{\pi} - r_{oR-\frac{1}{2}}^2 C_{oR} \right) \left( \ln r_{oR-\frac{1}{2}} - \frac{a_{oR}}{b_{oR}} \right) \right]
+ \frac{C_{oR}}{2} \left( r_{oR-\frac{1}{2}} - \frac{c_{oR}}{2b_{oR}} \right) + G_{oR} \left( \frac{2d_{oR}}{3b_{oR}} - r_{oR-\frac{1}{2}} \right)
\] (9.190)

and

\[
P_{oR-1} = P_{oR-\frac{1}{2}} + A_{oR-1} \left[ \left( \frac{S_{oR-\frac{1}{2}}}{\pi} - r_{oR-\frac{1}{2}}^2 C_{oR-1} \right) \left( \ln r_{oR-\frac{1}{2}} - \frac{a_{oR-1}}{b_{oR-1}} \right) \right]
+ \frac{C_{oR-1}}{2} \left( r_{oR-\frac{1}{2}} - \frac{c_{oR-1}}{2b_{oR-1}} \right) + G_{oR-1} \left( \frac{2d_{oR-1}}{3b_{oR-1}} - r_{oR-\frac{1}{2}} \right). \] (9.191)

Eliminating the vapour flowrate term from equations (9.189)–(9.191) gives the vapour layer periphery boundary condition,

\[
(1 - \alpha_{oR-1} D_6) P_{oR-1} + \left( -1 - \frac{B_{1, oR}}{A_{oR} e_{oR+\frac{1}{2}, oR}} - \alpha_{oR} D_7 \right) P_{oR} =
\left( \beta_{oR-1} - \frac{\partial \hat{\delta}_{oR-1}}{\partial t} \right) D_6 + \left( \beta_{oR} - \frac{\partial \hat{\delta}_{oR}}{\partial t} \right) D_7 + D_{10}, \] (9.192)

where

\[
D_6 = A_{oR-1} \left( \frac{1}{2} f_{oR-\frac{1}{2}, oR-1} - \frac{r_{oR-\frac{1}{2}}^2}{2} e_{oR-\frac{1}{2}, oR-1} \right), \] (9.193)

\[
D_7 = -\frac{B_{oR-1} f_{oR+\frac{1}{2}, oR} r_{oR+\frac{1}{2}}^2}{2 e_{oR+\frac{1}{2}, oR}} + \frac{B_{oR-1} r_{oR-\frac{1}{2}}^2}{2} - \frac{A_{oR} f_{oR-\frac{1}{2}, oR}}{2} + A_{oR} e_{oR-\frac{1}{2}, oR}, \] (9.194)

and

\[
D_{10} = -\frac{B_{oR-1} G_{oR} h_{oR+\frac{1}{2}, oR+\frac{1}{2}}}{A_{oR} e_{oR+\frac{1}{2}, oR+1}}. \] (9.195)
9.4.3 Vapour Layer Flow Solution Technique

Equation (9.178), combined with the two boundary condition equations (9.183) and (9.193) forms a set of linear equations which specify the pressure distribution within the vapour layer. As previously discussed, this set of equations is solved in BOUNCE using a tridiagonal matrix inversion algorithm.

In the preceding analysis, the lower case variable constants, \( a - h \), are dependent only on the subcell mesh spacings. Thus, these constants can be calculated prior to a droplet impact simulation commencing. Other constants generally have to be determined during each vapour layer iteration.

Under certain circumstances during an impact simulation, a droplet may separate into two or more discrete volumes of fluid. If such separation occurs, the vapour layer will be split into a number of discrete sections, each bounded by a region at atmospheric pressure.

In BOUNCE, it is assumed that the vapour layer is continuous from the centreline of the droplet to the periphery of the rightmost cell within the vapour layer. This assumption is necessary to avoid large pressure discontinuities occurring in the vapour layer when two fluid volumes join or separate. Such pressure discontinuities are capable of destroying the iterative implicit VOF calculation. If a column of VOF cells contains no fluid, and is between the centreline and rightmost periphery of the vapour layer, then the vapour height is set equal to the height of the top boundary within the computational domain. In practice, the error introduced in the droplet dynamics calculation caused by using this procedure is small, as the extreme vapour layer height imposed within these void regions ensures that the local vapour layer pressure is close to atmospheric.

Once the vapour layer subcell pressures have been determined, the droplet vaporisation velocities can be found from equation (9.1). By successively sub-
stituting these vaporisation velocities into equation (9.146), noting the central flowrate boundary condition equation (9.179), the vapour layer flowrates at the borders of each subcell can be determined. Integrity tests of the vapour layer flow solution algorithm are presented in Appendix D.

9.5 Air Volume Fraction Solution

In this section the computational procedure used to solve the air fraction transport equation, as developed in Section 8.3 of the previous chapter, is outlined.

9.5.1 Linear Air Volume Fraction Relationship

Equation (8.36) of the previous chapter gives the volume flowrate of air as

\[ y_a(r') S(r') = -2\pi \int_0^{r'} \left( y_a(r) \frac{d\delta(r)}{dt} + \delta(r) \frac{dy_a(r)}{dt} \right) r dr. \]  
(9.196)

This can be expressed as

\[ y_a(r' + \delta r') S(r' + \delta r') - y_a(r') S(r') = -2\pi \int_{r'}^{r' + \delta r'} \left( y_a(r) \frac{d\delta(r)}{dt} + \delta(r) \frac{dy_a(r)}{dt} \right) r dr. \]  
(9.197)

where \( \delta r' \) is a finite radial increment.

Noting that \( y_{a,o + \frac{1}{2}} \) represents the volume fraction of air within the layer at the right border of subcell \( o \), and that \( y_a(r) \) represents the volume fraction of air within subcell \( o \) at location \( r \), equation (9.197) can be applied over cell \( o \) to give

\[ y_{a,o + \frac{1}{2}} S_{o + \frac{1}{2}} - y_{a,o - \frac{1}{2}} S_{o - \frac{1}{2}} = -2\pi \int_{r_{o + \frac{1}{2}}}^{r_{o + \frac{1}{2}}} \left( y_a(r) \frac{\widehat{\delta_o}}{dt} + \widehat{\delta_o} \frac{dy_a(r)}{dt} \right) r dr. \]  
(9.198)
Here we have assumed that the vapour layer height and vapour layer height velocity are constant over the width of the subcell. Rearranging, noting that the limits of integration are not dependent on time, equation (9.198) becomes,

\[
y_{a,o} + \frac{1}{2} S_{o+\frac{1}{2}} - y_{a,o} - \frac{1}{2} S_{o-\frac{1}{2}} = \ -2\pi \left[ \frac{d\hat{\delta}_o}{dt} \int_{r_{o+\frac{1}{2}}}^{r_{o+\frac{1}{2}+\frac{1}{2}}} y_a(r) r dr + \hat{\delta}_o \frac{d}{dt} \left( \int_{r_{o+\frac{1}{2}}}^{r_{o+\frac{1}{2}+\frac{1}{2}}} y_a(r) r dr \right) \right]. \tag{9.199}
\]

The average air volume fraction within subcell \(o\) is defined as

\[
y_{a,o} = \frac{2}{r_{o+\frac{1}{2}}^{2} - r_{o-\frac{1}{2}}^{2}} \int_{r_{o+\frac{1}{2}}}^{r_{o+\frac{1}{2}+\frac{1}{2}}} y_a(r) r dr. \tag{9.200}
\]

Substitution into equation (9.199) gives the average air volume fraction within a subcell, in terms of the volume flowrate of air at the borders of the cell, as

\[
y_{a,o} + \frac{1}{2} S_{o+\frac{1}{2}} - y_{a,o} - \frac{1}{2} S_{o-\frac{1}{2}} = -\pi \left( r_{o+\frac{1}{2}}^{2} - r_{o-\frac{1}{2}}^{2} \right) \left( y_{a,o} \frac{\partial \hat{\delta}_o}{\partial t} + \hat{\delta}_o \frac{\partial y_{a,o}}{\partial t} \right). \tag{9.201}
\]

In BOUNCE, an upwind scheme is used to approximate the volume of air fractions at the borders of each subcell. Thus

\[
y_{a,n} + \frac{1}{2} = \begin{cases} 
  y_{a,n+1} & \text{if} \quad S_n < 0 \\
  y_{a,n} & \text{if} \quad S_n \geq 0
\end{cases}. \tag{9.202}
\]

Also, the time derivative in equation (9.201) is differenced using

\[
\frac{\partial y_{a,o}}{\partial t} = \frac{y_{a,o} - y_{a,o}^*}{\delta t_{y_a}}, \tag{9.203}
\]

where the asterisk denotes a variable from the last computational timestep, and \(\delta t_{y_a}\) is the length of the timestep over which the air volume fraction transport equation is applied.
Applying equations (9.202) and (9.203) to equation (9.201) gives an implicit relationship between three adjacent air volume subcell variables as

\[-\zeta_o \frac{1}{2} S_o \frac{1}{2} y_{a,o-1} + \left[\zeta_o \frac{1}{2} S_o \frac{1}{2} + \left(1 - \zeta_o \frac{1}{2}\right) S_o \frac{1}{2} + \gamma_o \left(\frac{\partial \hat{\delta}_o}{\partial t} + \frac{\hat{\delta}_o}{\partial y}\right)\right] y_{a,o} + \left(1 - \zeta_o \frac{1}{2}\right) S_o \frac{1}{2} y_{a,o+1} = \gamma_o \frac{\hat{\delta}_o y_{a,o}}{\delta t y_o},\]  

(9.204)

where the area of a subcell has been defined as

\[\gamma_n = \pi \left(r_{n+\frac{1}{2}}^2 - r_{n-\frac{1}{2}}^2\right),\]  

(9.205)

and

\[\zeta_n = \begin{cases} 
1 & \text{if } S_n > 0 \\
0 & \text{if } S_n \leq 0 
\end{cases}\]  

(9.206)

has been defined to identify the correct volume air fraction for the cell boundary upwind scheme.

### 9.5.2 Air Volume Fraction Boundary Conditions

At the axis of the droplet, the vapour flowrate is zero. This is expressed by equation (9.179). Substituting this boundary condition into equation (9.204), applied over cell \(o = 2\) gives

\[\left[\zeta_2 \frac{1}{2} S_2 \frac{1}{2} + \frac{\gamma_2}{2} \left(\frac{\partial \hat{\delta}_2}{\partial t} + \frac{\hat{\delta}_2}{\partial y}\right)\right] y_{a,2} + \left(1 - \zeta_2 \frac{1}{2}\right) S_2 \frac{1}{2} y_{a,3} = \frac{\gamma_2 \hat{\delta}_2 y_{a,2}}{\delta t y_2}.\]  

(9.207)

Equation (9.207) is the central air volume fraction boundary condition.

At the periphery of the vapour layer, we assume that the air volume fraction beyond the vapour layer is equal to the ambient, initial air volume fraction within the atmosphere. Thus, as \(o = o_R\) defines the outermost subcell within the vapour layer,

\[y_{a,o_R+1} = y_{a,i},\]  

(9.208)

250
where \( y_{a,i} \) is the initial or ambient air volume fraction within the atmosphere.

Applying equation (9.208) to equation (9.204), applied over cell \( o = o_R \) gives

\[
-\zeta_{or-1/2} S_{or-1/2} y_{a,or-1} + \left[ \zeta_{or+1/2} S_{or+1/2} + \left( 1 - \zeta_{or-1/2} \right) S_{or-1/2} + \gamma_{or} \left( \frac{\partial \delta_{or}}{\partial t} + \frac{\delta_{or}}{\delta t} \right) \right] y_{a,or} \\
+ \left( 1 - \zeta_{or+1/2} \right) S_{or+1/2} y_{a,i} = \gamma_{or} \frac{\delta_{or} \delta y_{a,or}}{\delta t}. 
\]

Equation (9.209) is the peripheral air volume fraction boundary condition.

The ambient air volume fraction within the atmosphere is chosen so that a mixture of droplet vapour and air at the initial temperature of the droplet and at atmospheric pressure would be in equilibrium. Under such conditions, the partial pressure of the droplet vapour equals the saturation pressure of the droplet liquid at the initial temperature of the droplet. Thus, utilising equations (8.25) and (8.26) of the previous chapter, the initial or ambient air volume fraction is given by

\[
y_{a,i} = 1 - \frac{P_{sat}(T_{i,i})}{P_{atm}}, \quad \text{(9.210)}
\]

noting that equation (9.86) gives the saturation pressure as a function of temperature.

### 9.5.3 Accuracy Analysis

It was found that accuracy of the vapour layer pressure distribution, and stability of the vapour layer iterative solution procedure, were both strongly dependent of the accuracy achieved in solving the air volume fraction transport equation. In this section an analysis is performed on the accuracy of this calculation.
Consider the control volume shown in Figure 9.7, which is contained within the vapour layer subcell, \( o \). The top of the control volume is stationary, and located at a distance \( \hat{\delta}_o \) above the base.

To ensure accurate simulation of the air volume fraction transport equation, a Courant type accuracy criterion is imposed on the control volume shown in Figure 9.7. Thus, it is assumed that the volume of fluid leaving the control volume during a timestep must be less than or equal to the volume contained within the control volume. Thus,

\[
S_{\text{out}} \delta t \leq V_{\text{cv}},
\]

(9.211)

where \( S_{\text{out}} \) is the total volume of fluid leaving the control volume during the timestep, and \( V_{\text{cv}} \) is the volume contained within the control volume. The
volume of fluid leaving the control volume is given by

\[ S_{out} = \max\left(-S_{o-\frac{1}{2}}, 0\right) + \max\left(-\left(\dot{w}_o - \frac{\partial \delta_o}{\partial t}\right) \gamma_o, 0\right) + \max\left(S_{o+\frac{1}{2}}, 0\right), \]  

(9.212)

where the absolute velocity of fluid crossing the top boundary of the control volume has been used, and \( \gamma_o \), as previously defined, is the upper surface area of subcell \( o \). From equation (9.146) however,

\[ \gamma_o \left(\dot{w}_o - \frac{\partial \delta_o}{\partial t}\right) = S_{o+\frac{1}{2}} - S_{o-\frac{1}{2}}, \]  

(9.213)

and thus,

\[ S_{out} = \max\left(-S_{o-\frac{1}{2}}, 0\right) + \max\left(S_{o-\frac{1}{2}} - S_{o+\frac{1}{2}}, 0\right) + \max\left(S_{o+\frac{1}{2}}, 0\right). \]  

(9.214)

The total volume of the control volume is

\[ V_{cv} = \hat{\delta}_o \gamma_o. \]  

(9.215)

Substituting equations (9.214) and (9.215) into equation (9.211) gives the Courant accuracy criterion for the vapour layer as

\[ \delta t \leq \delta t_V, \]  

(9.216)

where the vapour flow maximum timestep has been defined by

\[ \delta t_V = \frac{\delta_o \gamma_o}{\max\left(-S_{o-\frac{1}{2}}, 0\right) + \max\left(S_{o-\frac{1}{2}} - S_{o+\frac{1}{2}}, 0\right) + \max\left(S_{o+\frac{1}{2}}, 0\right)}. \]  

(9.217)

### 9.5.4 Air Volume Fraction Solution Technique

To solve for the air volume fraction distribution, the coefficients of the transport equation (9.204), combined with the coefficients from the boundary condition equations (9.207) and (9.209), are cast into the form of a tridiagonal
matrix. This matrix is solved in subroutine YASOLVER using a tridiagonal matrix inversion algorithm, as detailed in Section 9.1.2.

The timestep used to advect the air volume fractions is given by

$$\delta t_{ya} = \min (\delta t, \delta t_{V}),$$  \hfill (9.218)

where $\delta t$ is the timestep used by the main droplet dynamics algorithm, and $\delta t_{V}$ is the accuracy limited air fraction timestep, as calculated above. Equation (9.218) has been implemented to ensure that values calculated for the air volume fractions during the iterative process are realistic.

Once convergence of the vapour layer solution has been achieved, a check is made that the timestep used to advect the air fraction solution is in fact the timestep taken by the main droplet dynamics algorithm. Thus, the final check ensures

$$\delta t_{ya} = \delta t.$$  \hfill (9.219)

If equation (9.219) is not satisfied, the main droplet dynamics algorithm timestep is halved, and the timestep repeated.

An integrity test of the air volume fraction solution algorithm is presented in Appendix D.
Chapter 10

Droplet Impact Simulations — Saturated Droplets

In this chapter, droplet impact simulations performed using the BOUNCE code are compared against actual impacts detailed in experimental studies. In all the impacts examined in this chapter, the droplet is initially at the saturation temperature corresponding to atmospheric pressure.

10.1 The Wachters and Westerling Impacts

The classic paper of Wachters & Westerling [1966], as discussed in Chapter 2, was one of the first investigations into the heat transfer and dynamics of impinging saturated water droplets. In this section, three individual impacts detailed in the Wachters & Westerling [1966] study are used to validate the BOUNCE code. The three impacts have the initial Weber numbers of 15, 74 and 184, and are consequently representative of the three different hydrodynamical regimes of droplet impact discussed in Section 1.3.
10.1.1 The We = 15 Wachters and Westerling Impact Hydrodynamic Behaviour

Figure 3.5 in Chapter 3 shows an actual droplet impact presented in Wachters & Westerling [1966], and Figure 10.1 shows the BOUNCE simulation of the same impact. The Weber number for this impact was 15, corresponding to an impact velocity of 0.63 m/s.

The BOUNCE simulation shown in Figure 10.1 was performed using 50×140 square mesh cells over a computational domain of 0 to 2.5 mm radially and −0.1 to 6.9 mm vertically. The code took approximately 13,000 iterations to simulate 18 ms of impact, giving an average computational timestep of just 1.4 µs. Such a small timestep is a result of the inherently stiff nature of the physical problem. The liquid, vapour and solid properties used in this computation are detailed in Appendix C. A listing of the computational properties used in simulating the impacts of this chapter and the next is also given in Appendix C.

The droplet images shown in Figure 10.1 were generated by plotting the VOF contour line, \( F = 0.5 \). As the computation was performed in two dimensional cylindrical coordinates, each contour plot was reflected around the axis of the droplet to produce the cross sectional images displayed in the figure. The computational images were generated at times which approximately correspond to the times of the experimental images shown in Figure 3.5.

Comparing Figures 3.5 and 10.1 it is evident that BOUNCE predicts the dynamic behaviour of the droplet well. The first four frames, that is up to 1 ms, display the initial impact of the droplet. As shown in the generated images, upon impact, a wave of water at the bottom surface of the droplet...
Figure 10.1: Simulated droplet impact replicating the Wachters and Westerling $We = 15$ impact.
moves radially outward from the centre. The crest of this wave is very close to the solid, but with increasing distance from the centre, moves slightly upward. Due to the reflection of light off the outer surface of the photographed droplet, it is not possible to directly compare the behaviour of the droplet base within the vapour layer. However, consistent with the simulation, the photographs do show that the vapour layer height is very small at the extremity of the layer.

The next five frames, from 1.5 to 3.5 ms, show the droplet spreading out over the surface. By this stage, the thickness of the vapour layer has increased to a visible level in the computational images, however, the resolution of the photographed impact does not allow validation of this behaviour.

There appears to be a distinctive central feature within the droplet shown in photograph nine (3.85 ms) of the experimental impact. It is not clear from the photograph whether this feature is a hump of fluid or a void region within the droplet. The computational image at approximately the same time shows a hump of fluid which is less pronounced than the feature shown in the experimental impact. The discrepancy between the simulated and experimental impact at this time is most likely to be caused by deficiencies in the surface tension algorithm, however, it would be difficult to confirm this theory.

The last seven frames, from 5 ms onwards, show the droplet recollecting as it is pushed away from the surface. During this period the vapour layer geometry and vapour layer height change quickly and substantially, but it is not possible to validate this behaviour from the photographed images of Figure 3.5. BOUNCE predicts the droplet recoil process well, particularly the dumbbell shape shown in the final photograph and computational frame.

A small void area, or bubble, is shown in frames 13 and 14 of the compu-
tational images. This void region was formed at approximately 6.7 ms, when a small vapour layer region at the centreline of the droplet was enclosed in liquid. Although the resolution of the photographic images is not as high as the resolution of the computational images, bubble formation is not observed in the experimental impact during the droplet recollecting process.

Void regions within fluid volumes can be formed by VOF codes when a channel of void within a fluid region becomes narrower than the computational cell dimensions. In such instances, the discrete nature of the VOF function can act to sever the channel of void, effectively creating bubbles inside the fluid volume. Surface tension forces act to re-enforce this phenomenon by combining small void regions into larger void regions, or bubbles. Thus, when bubbles, which have dimensions of the order of the computational cell size, are formed in computations, their existence must be treated with caution. The bubble shown in Figure 10.1 is of a similar size than the computational cell dimensions, so its presence may not be realistic.

Computations were continued past the last experimental photograph shown in Figure 10.1 for a further 3 ms. These showed that the dumbbell shape shown at 14.7 ms reforms into one oscillating droplet within a few milliseconds.

It should be noted that the photographs shown in Figure 3.5 are a record of a single droplet impact, rather than an average of how many such similar droplets would behave. Consequently, caution should be exercised when comparing the finer details of these computational and experimental results. Indeed, most of the experimental data with which comparison will be made in this study has been taken from single droplet impacts, so caution should be exercised when comparing specific details of any of these results.

Figure 10.2 shows the vapour layer height at the centre of the droplet, $\delta_2$, 

259
and the minimum vapour layer height within the vapour layer, $\delta_{\text{min}}$, during the Wachters & Westerling $\text{We} = 15$ impact.

The minimum vapour layer height shown in Figure 10.2 has been smoothed over a time period of 0.1 ms. In a real droplet impact, when a wave of fluid propagates outward along the lower surface of a droplet, the minimum height within the vapour layer remains fairly constant because the fluid interface can deform continuously throughout both space and time. In an impact simulation however, when a wave of fluid moves outward along the lower surface of a droplet, the minimum vapour layer height increases and decreases
as droplet liquid moves from one column of VOF computational cells to the next. Thus, the minimum vapour layer height in impact simulations oscillates with a frequency which is dependent on the size of the VOF computational cells — the smaller the size of the computational cells, the more continuously the droplet surface deforms, and the smaller the oscillations in the minimum vapour layer height are. A 0.1 ms time period of smoothing in the above figure removed the majority of vapour layer height oscillations, while not affecting the form of the results.

The vapour layer heights shown in Figure 10.2 support the phenomena that were observed in the photographic frames of the experimental impact. When the droplet first approaches the surface, the centre of the droplet comes within approximately 4 µm of the surface before rebounding sharply upward. The minimum height within the vapour layer then remains at the extremity of the vapour layer, gradually increasing to approximately 50 µm over the first 5 ms of the impact. As shown by the photographic images of Figure 3.5, it is during these 5 ms that the droplet expands to its maximum radius of the impact period.

From 5 ms to 11 ms, the height and geometry of the vapour layer display large oscillations. As shown in the photographs of Figure 3.5, it is during this period that the droplet is recollecting and attempting to leave the solid surface. As surface tension acts to reform the droplet, the lower surface of the droplet is forced towards the solid, causing several small vapour layer heights during this period.

Figure 10.2 indicates that the height of the vapour layer at the centreline of the droplet approaches 10 nm during the recollecting period. This figure is unrealistic, the small value being a product of the discrete nature of the vapour layer geometry. A better estimate of the minimum height of the
vapour layer is given by the minimum vapour layer height curve, which as previously discussed, has been smoothed to remove the sharp oscillations resulting from the discrete vapour layer geometry treatment. The minimum height of the vapour layer during the recollecting process is less than the initial impact minimum height, at approximately 3 \( \mu \text{m} \).

Over the last 4 ms of the impact, the droplet is moving away from the solid. During this period the minimum height within the vapour layer occurs at the centreline of the droplet, because as shown in Figure 3.5, the base of the droplet is approximately spherical.

Figure 10.3 shows the maximum height and radius of the droplet during both the simulated and experimental impacts. The correlation between the two results is good, the BOUNCE model accurately predicting the oscillation period and overall behaviour of the experimental droplet.

The height of the droplet during the recollecting period is underpredicted in the simulations by approximately 1 mm. This underprediction is consistent with the differences between the photographic and computational images of Figures 3.5 and 10.1, respectively, which show that the experimental droplet generally has a smaller mid height radius than the computational droplet during the recoil period. This consequently gives the experimental droplet a larger height than the computational droplet during the last few milliseconds of the impact.

Interestingly, the measurements shown in Figure 10.3 indicate that the experimental droplet, prior to impacting the solid, is not completely spherical. The initial droplet diameter in the radial direction is 2.23 mm, that in the vertical direction is 2.37 mm. This droplet eccentricity may have caused small differences between the experimental and computational results.

Figure 10.4 shows the maximum Knudsen number within the vapour layer
during the We = 15 impact. The maximum Knudsen number shown has been smoothed over a time period of 0.1 ms, consistent with the smoothing performed on the minimum vapour layer height shown in Figure 10.2.

The maximum Knudsen number found during the Wachters and Westerling We = 15 impact is of the order 0.04. Referring to Section A.1 of Appendix A, this Knudsen number places the vapour layer flow in the molecular flow slip regime. Thus, the use of a kinetic treatment at the solid-vapour
and liquid-vapour interfaces was necessary in the simulation of this droplet impact.

Figure 10.5 shows the average vertical velocity of the droplet during the Wachters and Westerling We = 15 simulation. Upon impact, the average vertical velocity of the droplet increases rapidly as the pressures generated within the vapour layer act to slow the impinging liquid mass. Over the next few milliseconds, the vertical velocity of the droplet continues to increase, but the rate of velocity increase slows as the droplet spreads out radially over the solid surface.

At approximately 7 ms, a sharp increase in the vertical velocity of the
Figure 10.5: Average droplet vertical velocity during the Wachters and Westerling We = 15 impact.

droplet occurs. Referring to frames 8 and 9 of the computational images shown in Figure 10.1, it is evident that this sharp increase in droplet velocity is a result of the droplet recollecting, and the top of the droplet moving rapidly upwards and away from the solid.

Figure 10.5 shows that the average vertical velocity of the droplet as it leaves the solid is approximately 0.3 m/s, or approximately one half of the magnitude of the impact velocity. The magnitudes of the vertical impact and rebound velocities are not equal, as a proportion of the kinetic energy possessed by the droplet on impact has been converted into internal kinetic, gravitational potential and surface tension energies. Also, work has been
performed on the fluid contained within the viscous vapour layer region, which has required energy from the droplet. The last 6 ms of the computation shows the average vertical velocity of the droplet decreasing as gravity acts to slow the rebounding droplet.

Figure 10.6 shows the vapour layer pressure at the axis of the droplet during the Wachters and Westerling We = 15 impact simulation. As pressures generated within the vapour layer are dependent on the geometry of the vapour layer, these pressures typically show oscillations which are caused by the discrete treatment of the vapour layer geometry. To remove these oscillations, the pressure shown in Figure 10.6 has been smoothed over a time
period of 0.1 ms.

As shown in Figure 10.6, the average pressure at the axis of the droplet during the impact is quite low, of the order of several hundred pascals. However, when the droplet comes within a small distance of the solid, the local gauge pressure can reach levels of the order of 10,000 Pa. Although these peaks are substantially higher than the average pressure within the layer throughout the impact period, this level is considerably smaller than atmospheric pressure, and as such, the incompressible gas assumption utilised when simplifying the vapour layer momentum equations in Section 8.2.1 is justified.

**Thermodynamic Behaviour**

The predicted vapour layer centreline temperatures at both the liquid-vapour and solid-vapour interfaces during the Wachters and Westerling We = 15 impact are shown in Figure 10.7. The interface temperatures are dependent on the local vapour layer pressures, and are thus subject to oscillations. The temperatures shown in Figure 10.7, like the local vapour layer pressure shown in Figure 10.6, have been smoothed over a time period of 0.1 ms.

The solid centreline temperature shown in Figure 10.7 remains within one degree of its initial value throughout the duration of the impact, a result of the high thermal conductivity of the chosen impact material. Small drops in the solid surface temperature coincide with minimums in the centreline vapour layer height shown in Figure 10.2. Similarly, the temperature of the liquid surface generally remains close to its initial value throughout the impact duration, with the exception of several peaks which also coincide with local vapour height minimums.

Figure 10.7 shows that substantial temperature discontinuities occur over
both the liquid-vapour and solid-vapour interfaces during the impact simulation. However, the level of these discontinuities has little effect on the local rate of heat transfer across the vapour layer, as the magnitudes of both the discontinuities are small compared with the temperature difference between the solid and liquid surfaces.

Figures 10.8 and 10.9 show the total energy amounts transported during the Wachters and Westerling We = 15 simulation, and the local vapour layer centreline energy fluxes calculated during the same simulation, respectively.
Figure 10.8: Total energy transported during the simulated Wachters and Westerling We = 15 droplet impact. The total energy conducted into the body of the droplet during the impact is represented by $e_{\text{drop}}$, the total energy consumed by vaporisation of liquid during the impact is represented by $e_{\text{vap}}$, and the total energy transferred across the vapour layer during the impact is represented by $e_{\text{layer}}$.

As local energy fluxes are heavily dependent on local vapour layer pressures, the energy flux data shown in Figure 10.9 has been smoothed over a time period of 0.1 ms.

Figure 10.8 shows that the total energy transported across the vapour layer during the droplet impact is approximately equal to the total energy consumed in vaporising liquid during the impact. Consequently, the total
energy conducted into the droplet during the impact is negligible. This result is as expected, as the droplet is initially at saturation temperature, and leaves the solid with its surface at saturation temperature. Thus, there is little opportunity for the temperature of the droplet fluid to be increased substantially. Note that the droplet removes approximately 0.012 J from the solid during the impact simulation.

The results of Figure 10.9 support the results shown in Figure 10.8. Peak
local heat transfer rates across the vapour layer shown in Figure 10.9 approach the order of $1.5 \times 10^6$ W/m$^2$ during the impact simulation, but average local heat transfer rates are substantially lower than this value. Consistent with the results found for the total heat transferred, the heat transfer rate into the volume of the droplet is generally very low. However, when the local droplet surface nears the solid, the droplet conduction heat transfer rate displays a large peak, and conversely when the local droplet surface moves away from the solid, the droplet conduction heat transfer rate displays a large trough.

The total amount of liquid vaporised from the droplet during the Wachters and Westerling We = 15 simulation was calculated at $5.0 \times 10^{-12}$ m$^3$. Defining $\eta$ as the ratio of the volume of liquid vaporised during the impact to the initial volume of the droplet,

$$\eta = \frac{\delta V_{\text{drop}}}{V_{\text{drop,i}}}$$  \hspace{1cm} (10.1)

then $\eta = 0.078\%$ for the examined impact. Such a low value for $\eta$ indicates the high efficiency, in terms of the volume of the droplet consumed, of this rebound process.

**Coarse Mesh Wachters and Westerling We = 15 Simulation**

In order to test the integrity of the above droplet impact simulation, the calculation was repeated using a coarser computational mesh. The simulation shown in Figure 10.10 was performed using $24 \times 70$ square mesh cells over a computational domain of 0 to 2.4 mm radially and $-0.2$ to 6.8 mm vertically. All other computational variables remained unchanged from the previous Wachters and Westerling We = 15 fine mesh impact simulation.

Comparing Figures 10.10 and 10.1, it is evident that both calculations produce qualitatively the same droplet impact simulations, however the finer
Figure 10.10: Simulated droplet impact replicating the Wachters and Westerling \( \text{We} = 15 \) impact. The computational cell size used in this calculation was twice the size of the cell size used in the previous calculations.
mesh calculation of Figure 10.1 appears to give a slightly better estimation of the actual droplet behaviour.

The coarse mesh calculation of Figure 10.10 displays a larger bubble within the droplet in the last milliseconds of the impact than the fine mesh calculation does. However, as the bubble is still of a similar size to the computational mesh, its existence may not be realistic.

Other features of the coarse mesh calculation appear to be marginally less accurate than those of the fine mesh calculation. The radius at the top of the droplet at 7.2 ms is better predicted by the fine mesh calculation of Figure 10.1 than the coarse mesh calculation of Figure 10.10. Also, the fine mesh calculation predicts a smaller mid-height radius of the droplet in the last computational frame than the coarse mesh calculation, which is a better approximation of the shape of the actual droplet in the last frame of Figure 3.5.

Despite the differences between the coarse mesh and fine mesh calculations, it is clear that the droplet behaviour predicted in both simulations is similar, and as such, the finer grid size used in the above calculation is of sufficient accuracy to replicate the main features of the droplet impact flow.

**Simplified Vapour Layer Wachters and Westerling We = 15 Simulation**

Figure 10.11 shows the Wachters and Westerling We = 15 fine mesh calculation repeated, but in this simulation the solid temperature was assumed to be uniform and constant at its initial value, and the vapour layer molecular interface treatment was neglected. Comparing Figures 10.1 and 10.11, it is evident that these simplifications have not noticeably affected the accuracy of the calculation.
Figure 10.11: Simulated droplet impact replicating the Wachters and Westerling $We = 15$ impact. In this calculation the temperature of the solid was assumed constant, and the vapour layer molecular interface treatment was neglected.
As shown in Figure 10.7, when heat transfer within the solid phase is solved during the Wachters and Westerling We = 15 impact, the temperature at the surface of the solid deviates little from its initial value. As the heat transfer rate into the bulk of the droplet is dependent on the difference between the liquid and solid surface temperatures, assuming a constant solid surface temperature does little to affect the accuracy of the calculation.

Figure 10.7 also showed that during the Wachters and Westerling We = 15 impact, the temperature discontinuities occurring at both interfaces were substantial, but small compared with the temperature difference between the solid and liquid phases. Thus, neglecting the molecular interface treatment had little effect on the heat transfer rate across the vapour layer, and subsequently little effect on the accuracy of the droplet impact simulation.

10.1.2 The We = 74 Wachters and Westerling Impact

The second droplet dynamics experiment presented in Wachters & Westerling [1966] was performed using identical conditions to the first, except that the impact velocity of the droplet was increased to 1.41 m/s, yielding a Weber number for the impact of 74. Figure 10.12 shows the photographs presented in Wachters & Westerling [1966] of this impact, and Figure 10.13 the BOUNCE simulation of the same impact.

The BOUNCE simulation shown in Figure 10.13 was performed using 100×130 square mesh cells over a computational domain of 0 to 5 mm radially and −0.1 to 6.4 mm vertically. The calculation required approximately 18,000 computational timesteps to produce 18 ms of impact simulation. Thus, the average computational step size used was 1 μs.

A comparison of Figures 10.12 and 10.13 shows that BOUNCE predicts the impact behaviour of the droplet accurately until approximately 3 ms, but
Figure 10.12: The We = 74 impact performed by Wachters and Westerling.
(Taken from Wachters & Westerling [1966].)
Figure 10.13: Simulated droplet impact replicating the Wachters and Westerling We = 74 impact.
beyond this time the correlation between the computational and experimental results deteriorates.

The initial stages of the Wachters and Westerling $\text{We} = 74$ impact are predicted well by BOUNCE. From 0.4 to 1.3 ms, the experimental photographs of Figure 10.12 show a jet of liquid spreading out across the surface of the solid, emanating from the droplet solid impact region. The BOUNCE simulation predicts the movement of this jet accurately, in particular the rounding of the advancing edge of the jet by surface tension forces, and the slight upward movement of the jet evident between 0.9 and 1.4 ms.

At 2.2 ms, the BOUNCE simulation predicts that the leading edge of the liquid jet separates from the main bulk of the droplet, forming a toroid of liquid with a small cross-sectional diameter. The resolution of the experimental images precludes validation of the droplet separating into two discrete masses, however, the experimental images do show that the height of the droplet around the circumference is not uniform. The geometry of the top surface of the main droplet volume is predicted accurately at this time.

From approximately 3 to 6 ms, the BOUNCE simulations of Figure 10.13 show the droplet separating into several more toroids of liquid, while the experimental images of Figure 10.12 show the liquid mass oscillating in a comparatively more random, asymmetrical manner. The difference between simulation and reality at these times can be attributed to the coordinate system in which the simulation is performed.

When a droplet impacts a solid surface, its average velocity passes through zero before the droplet rebounds from the surface. Thus, at some stage during an impact, the total initial kinetic energy possessed by a droplet must be converted into other forms of energy. The two main forms of energy into which the initial energy of the droplet is converted are internal kinetic
energy, responsible for the radial expansion of the droplet during the impact, and surface energy. The higher the initial velocity of the droplet, the more total initial kinetic energy must be converted into these alternative forms.

During the experimental droplet impact shown in Figure 10.12, from approximately 3 ms onwards, the droplet appears to oscillate in an asymmetrical manner. The height and velocities of the droplet appear to be not only a function of the radial and axial location within the droplet, but also the cylindrical location around the axis of the droplet. The asymmetrical oscillations of the droplet shown in these photographs require energy that was provided by the initial total kinetic energy of the droplet — the oscillating velocity of the fluid requires internal kinetic energy, while the undulating surface of the droplet has a high surface area and requires significant surface energy.

The BOUNCE impact simulations are performed using two dimensional cylindrical coordinates. Consequently, free surface geometry and fluid velocity are represented as a function of radial and axial location only, and the asymmetrical oscillations shown in the experimental impact of Figure 10.12 cannot be simulated. Instead, the BOUNCE simulation must conserve the total energy of the droplet using other methods.

As shown in Figure 10.13, from 3 ms onwards, toroids of fluid are formed at the circumference of the simulated droplet. With each new toroid formed, the total surface of the droplet mass increases, and consequently, more surface energy is required. However, the formation of toroids in the simulated impact requires less energy than the asymmetrical oscillations of the experimental impact. As a result, to increase the total surface area of the fluid mass, and to consume internal kinetic energy, the simulated droplet spreads out further and more rapidly across the solid surface than the experimental
impact. Thus, while the experimental droplet remains as an oscillating mass at the centre of the impact region, the simulated droplet breaks into several toroids of liquid whose diameter expands rapidly and substantially across the computational domain.

Beyond 6 ms, the correlation between the experimental and computational impacts shown in Figures 10.12 and 10.13 respectively, continues to deteriorate. The experimental droplet recollects during this period, forming a small diameter elongated droplet and a much smaller ejected droplet by the last frame of the experimental figure. The simulated droplet on the other hand remains as several separate masses of fluid, the outer toroids of fluid not being given sufficient time to return to the centre of the computational domain and reform into one mass of fluid.

Figure 10.14 shows a comparison of the maximum droplet dimensions during the Wachters and Westerling We = 74 impact, found from the experimental and simulated impacts. The experimental results shown in this figure where measured directly from the photographic images of Figure 10.12 and as a result, are accurate to within ±0.12 mm.

The comparison between the experimental and computational results of Figure 10.14 is good until approximately 2 ms, but beyond this time, the radius of the simulated droplet decreases substantially faster than the radius of the actual droplet, and the height of the simulated droplet remains smaller than the height of the actual droplet. These later time discrepancies are caused by two factors.

The maximum radius of the simulated droplet is calculated using

$$ R_{\text{max}} = \max \left( \sum_{i=2,i_{\text{max}}} F_{i,j} \delta x_i \right), \quad (10.2) $$

where the maximum function in this equation is evaluated over all rows con-
Figure 10.14: Maximum radius and height of the experimental and simulated Wachters and Westerling We = 74 impact. The experimental results are indicated by symbols, the computational results by continuous lines. The error-bars attached to the symbols indicate the accuracy of the experimental results.

tained within the computational domain. The maximum radius is calculated in this manner so that the radius changes continuously as the number of VOF cells which contain fluid changes. Thus, when a simulated droplet breaks into a number of discrete masses of fluid, the void regions between the masses are not included in the droplet radius calculation.

The maximum radius of the experimental droplet is calculated as the maximum radial distance from the centre of the impact region at which fluid lies. Thus, the maximum experimental radius includes possible void regions.
contained within the outer radius of the droplet, and is consequently larger than the simulated maximum radius, if the droplet contains void regions. As shown in Figure 10.13, the simulated droplet splits into a number of separate fluid masses beyond 2 ms, and consequently, the simulated maximum radius would be expected to be lower than the actual maximum radius beyond this time.

The second and perhaps more important reason for the later time discrepancies shown in Figure 10.14 is the same reason which caused discrepancies between the simulated images of Figure 10.13 and the experimental images of Figure 10.12. At times beyond 2 ms, the two dimensional BOUNCE algorithm cannot replicate the motion of the oscillating experimental droplet. Consequently, the correlation between the maximum height and radius of the experimental and simulated droplets is poor.

Figure 10.15 shows the maximum Knudsen number calculated during the We = 74 droplet impact. Comparing this figure to Figure 10.4, it is evident that the maximum Knudsen number found during the higher velocity impact is slightly higher than the same number found during the We = 15 impact. This is a result of the slightly smaller initial vapour layer height generated in the higher velocity impact. The maximum level of the Knudsen number shown in Figure 10.15 indicates that the molecular slip regime assumption outlined in Appendix A is still valid in this impact.

10.1.3 The We = 184 Wachters and Westerling Impact

The final Wachters and Westerling impact examined in this study was conducted with a Weber number of 184, corresponding to an impact velocity of 2.23 m/s. All other experimental variables used in this impact remain unchanged from the previously examined Wachters and Westerling impacts.
Figure 10.15: The maximum Knudsen number calculated during the simulated Wachters and Westerling We = 74 impact. The Knudsen number shown in this figure has been smoothed over a time period of 0.1 ms.

Figure 10.16 shows the experimental We = 184 impact as presented in Wachters & Westerling [1966], while Figure 10.17 shows the same impact as simulated by BOUNCE. The simulation required 160×80 square grid cells over a computational domain of 0 to 8 mm radially and −0.1 to 3.9 mm vertically, and took approximately 18,000 iterations to produce 10 ms of droplet behaviour. Thus, the average computational timestep used was approximately 0.56 µs.

In a similar manner to the results of the We = 74 Wachters and Westerling impact, BOUNCE predicts the initial stages of the We = 184 impact
Figure 10.16: The We = 184 impact performed by Wachters and Westerling. (Taken from Wachters & Westerling [1966].)
Figure 10.17: Simulated droplet impact replicating the Wachters and Westerling \( \text{We} = 184 \) impact.
reasonably accurately, but at times beyond 1.5 ms, the correlation between experimental and computational images deteriorates.

The first three frames of the experimental and computational figures show the droplet impacting and expanding over the solid surface. While the motion of the bulk of the fluid mass is predicted accurately by BOUNCE during this period, the experimental images indicate the formation of small ‘satellite’ droplets which are ejected from the main droplet. The simulated images also show a small amount of fluid ejected from the main droplet, but this fluid is in the form of a toroid, and its ejection occurs at a slightly later time than the ejection of the satellite droplets in the experimental images.

The difference in fluid ejection behaviour between the experimental and computational droplets can again be attributed to the coordinate system in which the computations are performed. In the first few frames of the simulated images, BOUNCE predicts that a thin jet of liquid forms at the periphery of the droplet, which subsequently breaks from the main structure to form a small cross-sectional diameter toroid. The existence of either of these features is non-physical, because small perturbations in the fluid flow would quickly break such structures into a number of small satellite droplets. This behaviour is observed in the first frames of the experimental images, but is not reproducible in the simulations because the BOUNCE code uses a two dimensional cylindrical coordinate system in which satellite droplets cannot be represented.

At times beyond 1.5 ms, the correlation between the simulated and experimental images deteriorates, as the experimental droplet breaks up into an increasing number of smaller droplets, and the simulated droplet breaks into a slightly fewer number of small cross-sectional diameter toroids. Interestingly, droplets are ejected from the bounds of the photographic images of
Figure 10.16 seemingly never to return to the main mass of fluid. This is a difference between the computational and experimental impacts, as fluid ejected from the main droplet mass in a simulated impact will always return to the centre of the computational domain, provided the computational domain is large enough, and given sufficient time.

The final frames of the experimental images of Figure 10.16 show that the real droplet has divided into a large number of smaller droplets. It is not clear from the images whether the majority of these will reform into a substantially sized fluid mass. Conversely, the computational images of Figure 10.17 show that a significant proportion of the droplet mass has reformed and is moving away from the solid surface, while a number of slow inwardly moving toroids reside in the vicinity of the hot surface.

Figure 10.18 shows the maximum Knudsen number calculated during the simulated Wachters and Westerling We = 184 impact. The maximum Knudsen number achieved during this impact is slightly higher than the We = 74 impact, but still within the molecular slip flow regime detailed in Appendix A.

The three Wachters and Westerling impacts presented above have demonstrated the ability of BOUNCE to reproduce the fluid dynamics of hot surface droplet impingement.

The first impact studied showed that for a Weber number of 18, BOUNCE was able to predict all aspects of the impingement process well, including the surface tension driven rebound process of the droplet away from the hot surface.

The second impact examined showed that for a Weber number of 74, BOUNCE was able to predict the important initial spreading of the droplet across the hot surface accurately, but when the actual droplet remained oscil-
Figure 10.18: The maximum Knudsen number calculated during the simulated impact of the Wachters and Westerling We = 184 impact. The Knudsen number shown in this figure has been smoothed over a time period of 0.1 ms.

lating on the hot surface, the two dimensional coordinate system employed by BOUNCE was not able to replicate this behaviour. This resulted in a poor correlation between the experimental and simulated droplet behaviour as it attempted to leave the solid surface.

The final impact examined showed that for a high Weber number of 184, BOUNCE was able to predict the impact and spreading behaviour of the main body of the droplet, but was not able to predict the generation and ejection of the small satellite droplets observed in the experimental images, and the subsequent breakup of the experimental droplet into a large number
of smaller droplets. It was shown that this deficiency was again a result of the two dimensional coordinate system employed by BOUNCE. The final rebound of the droplet in this high speed impact was poorly predicted.

10.2 The Groendes and Mesler Impact

The Wachters and Westerling impacts discussed in the previous section were used primarily to validate the droplet dynamics algorithms employed in the BOUNCE code. While the comparison presented in that section was encouraging, it provided only an indirect validation of the ability of BOUNCE to predict the properties and flow within the viscous vapour layer.

For a more direct examination of the viscous vapour layer, in this section we compare solid surface temperatures measured under an actual impacting droplet to the same temperatures calculated by the BOUNCE code. The experiment used to make this comparison is taken from the study by Groendes & Mesler [1982], which has previously been discussed in Section 2.2.1 of Chapter 2.

10.2.1 Hydrodynamic Behaviour

Figures 10.19 and 10.20 show photographs of the actual Groendes and Mesler droplet impact, while Figure 10.21 shows the BOUNCE simulation of the same impact. Figure 10.19 shows the droplet during the first 20 ms of the impact, while Figure 10.20 shows the droplet from 20 ms onwards. The BOUNCE simulation was performed using 94×100 square grid cells over a computational domain of 0 to 9.4 mm radially and −0.2 to 9.8 mm vertically. The average timestep used over the 65 ms of droplet simulation was approximately 1.1 μs.
Figure 10.19: The Groendes and Mesler water droplet impact. This figure displays frames from the first 20 ms of the impact. (Taken from Groendes & Mesler [1982].)
The BOUNCE code predicts the behaviour of the actual droplet well until approximately 15 ms, but beyond this time the correlation between the experimental and simulated results deteriorates.

The first two photographic frames of Figure 10.19 show that the shape of the impacting droplet, before coming into contact with the solid surface, is not precisely spherical. Having a diameter of 4.7 mm, the water droplet used in the Groendes experiment is over twice as large as the droplet used in
Figure 10.21: The Groendes and Mesler droplet impact simulated by the BOUNCE code.
the Wachters and Westerling experiments. As a result, the ratio of inertial to surface tension forces within the droplet is much larger in the Groendes and Mesler droplet than in the Wachters and Westerling droplet, and consequently, a droplet of the Groendes and Mesler size is less geometrically stable. The initial non-sphericity of the experimental droplet may cause some minor differences between the subsequent experimental and simulated impacts.

The first five frames of both the experimental and simulated images show the droplet impacting with, and spreading out over, the solid surface. The correlation between both impacts is good here, with the BOUNCE simulation accurately predicting the behaviour of the jet formed at the intersection between the liquid and solid materials. The BOUNCE simulation shows that at approximately 5.3 ms, the centre of the underside of the droplet moves towards the solid for a second time during the impact. This initiates a secondary wave along underside of the droplet, which moves from the axis of the computational region radially outwards.

By 8.3 ms, the droplet has almost reached the maximum radius of the impact, and the simulation shows that the thickness of the liquid sheet is small, especially near the perimeter of the droplet. The photographic image at the same time supports the simulated image qualitatively, although the resolution of the experimental image is not high enough to determine the geometry of the droplet within the outer perimeter region, and the experimental droplet is starting to show small non-cylindrical variations in thickness.

From 13 ms and onwards, the Groendes and Mesler droplet behaves in a similar fashion to the We = 74 Wachters and Westerling impact described in the previous section, the similarity in droplet dynamics is a result of the similarity of their respective Weber numbers. Frame 7 of the experimental figure shows that the actual droplet has begun to oscillate in an asymmet-
rical manner, while the same frame of the simulated image shows that the simulated droplet has broken into a large number of concentric toroids, which have spread a large distance from the centre of the impact region. The cause of the difference between the two impacts is the coordinate system which is used by BOUNCE, as was discussed in Section 10.1.2—asymmetrical droplet oscillations cannot be replicated in two dimensional cylindrical coordinates.

The final frames of both the experimental and computational figures show the droplets recollecting and moving away from the solid surface. BOUNCE approximately predicts the experimental behaviour, although in the simulated images a significant proportion of the droplet remains behind as a separate toroid of fluid, residing within the vicinity of the solid surface.

Figure 10.22 shows the vapour layer height under the simulated Groendes and Mesler impact, while Figure 10.23 shows the minimum Knudsen number within the vapour layer during the same impact. In general, the vapour layer heights shown in Figure 10.22 are of a similar magnitude to the heights found during the Wachters and Westerling impacts, although the maximum Knudsen number found in this impact is slightly less than the same quantity observed in the Wachters and Westerling impacts.

The droplet axis vapour layer height shown in Figure 10.22 shows a minimum at approximately 5.5 ms. This corresponds to the initiation of the droplet underside secondary wave, as shown in the fifth image of Figure 10.21.

10.2.2 Thermodynamic Behaviour

Figure 10.24 shows the solid surface temperatures measured during the Groendes and Mesler impact, and Figure 10.25 the vapour layer interface temperatures during the simulation of the same impact. The experimental temper-
Figure 10.22: Vapour layer heights below the simulated Groendes and Mesler droplet impact. The vapour layer height at the axis of the computational region is represented by $\delta_2$, while the minimum height within the vapour layer is represented by $\delta_{\min}$. The minimum vapour layer height shown in this figure has been smoothed over a time period of 0.1 ms.

Temperatures represent an average temperature over the area of the thermometer, while the computational temperatures are calculated as the temperatures at the centre of the computational domain.

The correlation between the solid surface temperatures shown in Figures 10.24 and 10.25 over the first 15 ms of the impact is good. The BOUNCE simulation predicts the form and magnitude of the initial temperature drop accurately. This temperature drop corresponds to the underside of the droplet
first impacting the solid surface.

At approximately 5.5 ms, the solid surface temperatures in both the computational and experimental results again decrease. The temperature drop shown in the experimental results confirms the prediction that the underside of the droplet moves towards the solid surface twice during the initial impact and spreading of the droplet over the solid surface. The form of the temperature drop in the experimental results is less severe than the drop shown in the computational results, most probably because the experimental temperature represents an average over a large region beneath the droplet, while

---

**Figure 10.23:** The maximum Knudsen number calculated during the simulated Groendes and Mesler impact. The Knudsen number shown in this figure has been smoothed over a time period of 0.1 ms.
the computational temperature represents the temperature at the centreline of the droplet.

At approximately 15 ms from the initial impact, the experimental and simulated solid surface temperatures both begin a series of fairly rapid oscillations, as the centre of the underside of the droplet oscillates towards and away from the solid surface. Despite the simulated droplet differing in geometrical form to the experimental droplet during this oscillatory period, the solid surface temperatures are reasonably well predicted. The minimum tem-

Figure 10.24: The solid surface temperature measured beneath the Groendes and Mesler subcooled water droplet impact. The initial surface temperature was 462°C, and each vertical axis division represents 2.6°C. The horizontal scale is time, each division representing 5 ms. (Taken from Groendes & Mesler [1982].)
Figure 10.25: Vapour layer interface temperatures at the centre of the computational domain during the simulated Groendes and Mesler droplet impact. All temperatures shown in this figure have been smoothed over a time period of 0.1 ms, and are values in excess of the saturation temperature of water at atmospheric pressure.

Temperature of the solid measured from the experiment is approximately 343°C during this period, while the simulation predicts approximately 344°C.

The experimental results of Figure 10.24 show that the centre of the underside of the droplet leaves the vicinity of the solid surface at approximately 27 ms, while the simulated impact predicts the same behaviour at approximately 33 ms. It is interesting that the formation of a separate toroid of fluid in the simulated results did not substantially effect the behaviour of the
solid surface temperature at the centre of the computational domain.

The correlation between the experimental and computational solid surface temperatures discussed above suggests that BOUNCE is correctly calculating the energy fluxes at the solid-vapour and liquid-vapour interfaces. Further, as the dynamics of the droplet impact is predicted by BOUNCE with reasonable accuracy, this implies that the fluid flow within the viscous vapour layer region is also predicted by BOUNCE with at least reasonable accuracy.

Figure 10.25 also shows vapour layer centreline temperatures at both the liquid-vapour and solid-vapour interfaces, as well as the liquid temperature at the same radial location, calculated during the Groendes and Mesler impact simulation. The temperature interface discontinuity at the solid-vapour interface is small in comparison to the magnitude of the temperature drop of the solid surface, a result of the low thermal diffusivity impact surface used in this experiment.

As in the Wachters and Westerling experiments, the temperature of the liquid and vapour at the liquid-vapour interface remains within a few degrees of the saturation value at all times. The several peaks in these variables coincide with the droplet approaching the solid surface. As with the previously discussed droplet impacts, the temperature discontinuity at the liquid-vapour interface is small in comparison with the temperature difference across the vapour layer, so it has only a small effect on the heat transfer rates across the vapour layer.

Figures 10.26 and 10.27 show the total energy transported across the vapour layer, and the energy fluxes at the centreline of the computational domain during the Groendes and Mesler impact, respectively. As in the Wachters and Westerling We = 15 impact, the total energy conducted into the droplet fluid during the impact is negligible, the energy crossing the
Figure 10.26: Total energy transported during the simulated Groendes and Mesler droplet impact. The total energy conducted into the body of the droplet during the impact is represented by $e_{\text{drop}}$, the total energy consumed by vaporisation of liquid during the impact is represented by $e_{\text{vap}}$, and the total energy transferred across the vapour layer during the impact is represented by $e_{\text{layer}}$.

vapour layer going almost entirely into vaporising droplet liquid.

The centreline vapour layer energy fluxes display similar trends to the total impact energy transfer amounts, with the flux of energy conducting into the volume of the liquid being negligible, except when the droplet nears and moves away from the solid surface. The peak energy flux across the vapour layer during the impact is of the order $2.5 \times 10^6$ W/m$^2$, however the
average flux levels are much lower.

This chapter has been concerned with validating the ability of BOUNCE to predict the impact of droplets on a hot surface, where the droplet is initially at saturation temperature. The Wachters and Westerling experiments [Wachters & Westerling 1966] were used primarily to validate the internal droplet dynamics algorithm, while the Groendes and Mesler experiment was
used primarily to validate the viscous vapour layer algorithm.

In the next chapter we will turn our attention to droplet impacts where the temperature of the droplet is below saturation, thus, the droplet is subcooled.
Chapter 11

Droplet Impact Simulations — Subcooled Droplets

In this chapter BOUNCE droplet impact simulations are compared with experimental droplet impacts. Unlike the impacts considered in the previous chapter however, the initial droplet temperature in these impacts is below the saturation temperature of the liquid, and as such, the droplets are subcooled.

11.1 Subcooled n-Heptane Droplet Impacts

The experiments used in this section to validate the BOUNCE code are taken from the photographic impact study by Chandra & Avedisian [1991] and the low gravity impact heat transfer study by Qiao & Chandra [1996]. The two sets of experiments share similar features, and were detailed previously in Chapter 2 Section 2.2.2.
11.1.1 Hydrodynamic Behaviour

Low Gravity Impact

Figure 11.1(c) shows an experimental impact presented in Qiao & Chandra [1996], and Figure 11.2 the same impact simulated by BOUNCE. In both cases, the temperature of the solid surface was initially 210°C, the diameter of the droplet was 1.5 mm and the initial Weber number of the impact was 32. The computations were performed using 60×210 square mesh cells over a computational domain of 0 to 1.8 mm radially, and −0.06 to 6.24 mm axially. The 16 ms of simulation required approximately 12,200 iterations, giving an average timestep of approximately 1.3 μs.

A comparison of Figures 11.1(c) and 11.2 indicates that BOUNCE simulates the hydrodynamics of the low gravity n-Heptane impact well.

Until approximately 3 ms, the computational and experimental images show the droplet expanding over the surface of the solid. BOUNCE predicts the formation and expansion of a jet at the interface between the solid and liquid phases, and replicates the rounding of this jet by surface tension forces. The slight upward movement of the lower surface of the jet at times beyond 1 ms is also predicted accurately by the code.

While the computational and experimental images generally correlate well during the impact and spreading period, the correlation between the first experimental image and the corresponding computational image is less satisfactory. The label adjacent to the first experimental image indicates that the image was taken 0.2 ms after the droplet impacted the solid surface. During this 0.2 ms, the top of the droplet would have travelled approximately 0.16 mm towards the solid surface, or roughly one tenth of the initial height of the droplet. Examination of the experimental photograph in Figure

304
Figure 11.1: Three low gravity n-Heptane impacts conducted at different initial solid temperatures. Case (a) in the text refers to the 178°C impact, Case (b) in the text refers to the 190°C impact and Case (c) in the text refers to the 210°C impact. (Taken from Qiao & Chandra [1996].)
Figure 11.2: Simulated droplet impact replicating the Qiao and Chandra n-Heptane low-gravity experiment. The initial solid temperature was 210°C.
11.1(c) shows that at the indicated 0.2 ms, the top of the droplet has advanced further than one tenth of the initial height of the droplet towards the solid surface. This suggests that the first experimental image actually represents the droplet position at a time after 0.2 ms from initial impact. Indeed, the first experimental image correlates more closely with the computational image of 0.56 ms than the computational image of 0.26 ms.

At 4 ms, the experimental images of Figure 11.1(c) show that only a thin layer of fluid exists at the axis of the droplet, and the majority of the fluid is contained within a large flat toroid. BOUNCE predicts this behaviour accurately, in particular the downward ‘crimping’ of the central thin sheet at the inner radius of the toroid.

Over the next few milliseconds, the droplet recollects and rebounds from the surface. At 6.6 ms, the experimental images show that the droplet has formed a rough ‘witch’s hat’ shape, and by 13.6 ms, the droplet is in the form of a triple dumb-bell shape and has left the surface. BOUNCE predicts this behaviour well, in particular the rounding of the top of the droplet at 6.6 ms by surface tension forces, and the unusual form shown in the last experimental frame. However, the computational triple dumb-bell droplet appears slightly less ‘top-heavy’ than the corresponding experimental droplet.

Normal Gravity Impact

A second n-Heptane droplet simulation was conducted to validate the ability of BOUNCE to replicate the motion of subcooled n-Heptane droplets under normal gravity conditions. Figure 1.2(c) in Chapter 1 shows an n-Heptane impact presented in Chandra & Avedisian [1991], and Figure 11.3 the BOUNCE simulation of the same impact. The initial solid temperature for this impact was 205°C, and the Weber number of the impact was 43. The
BOUNCE simulation was accomplished using the same grid as was used for the previously discussed n-Heptane low gravity impact, and the simulation took a similar number of timesteps to replicate the 16 ms of droplet motion.

As for the low gravity simulation, the comparison between the experimental images of Figure 1.2(c) and the computational images of Figure 11.3 is generally good.

BOUNCE is able to predict the impact and spreading of the droplet with reasonable accuracy, although the small dimple existing in the centre of the experimental droplet between 2 and 4 ms is not shown in the computations. The formation and rounding of the jet at the liquid-solid interface is predicted accurately.

BOUNCE also predicts the recollection and rebound process reasonably, although the sharp spike on the experimental droplet at 6 ms is less extreme in the computational impact, and the final computational frame does not show the small ejected droplet that is apparent in the experimental image. Note that the oscillatory perimeter height of the droplet at 4 ms in the experimental images, and the slanting of the spike at 6 ms in the experimental images cannot be replicated using the two dimensional coordinate system employed by BOUNCE.

Comparing Figures 11.1(c) and 1.2(c), it is clear that the changes in conditions between the two experiments had only a minor impact on subsequent droplet dynamics. However, additional simulations performed by BOUNCE indicated that the most significant factor causing the minor differences between the two experiments was the difference in Weber number, or impact velocity, rather than the difference in gravitational magnitude. Indeed, computations indicated that the initial impact behaviour of a droplet is largely independent of the magnitude of gravity, a result of the short time period
Figure 11.3: Simulated droplet impact replicating the Chandra and Avedisian n-Heptane normal gravity experiment. The initial solid temperature was 205°C.
over which impacts are accomplished.

11.1.2 Thermodynamic Behaviour

Vapour Layer Interface Temperatures

Figure 11.4 shows the centreline vapour layer interface temperatures calculated during a low gravity Qiao and Chandra n-Heptane impact. The initial solid temperature for this simulation was 300°C. All other computational properties used in this computation were the same as those used during the low gravity simulation of Figure 11.2.

The solid line in Figure 11.4 represents the solid surface temperature beneath the centreline of the droplet. During the first 2 ms of the impact, this temperature decreases rapidly as the droplet impacts and deforms across the surface of the solid. Such a rapid temperature decrease is due to the small central vapour layer height existing during this period, which allows a large heat transfer rate from the solid within the central region.

After 2 ms, the vapour layer height at the centre of the impact region increases as the droplet spreads over the surface of the solid. This reduces the heat transfer rate from the central solid region, in turn causing the central solid surface temperature to stabilise. By the end of the expansion period, at approximately 5 ms from the initial impact, the central solid surface temperature is approximately 5 K below the initial solid temperature.

After 5 ms, the droplet begins to recollect and rebound from the surface. As this process occurs, the centre of the droplet is forced closer towards the solid, and as a result, the solid temperature at the axis of the droplet again decreases sharply. In the period between 5 and 9 ms, the central solid surface temperature drops a further 11 K. After 9 ms, the droplet leaves the vicinity
Figure 11.4: Temperatures at the liquid-vapour and solid-vapour interfaces predicted during the low gravity Qian and Chandra impact. The initial solid temperature in this simulation was 300°C. Temperatures shown in this figure are measured relative to saturation temperature of the droplet liquid at atmospheric pressure, and are smoothed over a time period of 0.1 ms.

of the solid surface, and the surface temperature at the axis of the droplet gradually increases as heat diffuses from within the body of the solid towards the surface.

The centreline vapour temperature at the solid-vapour interface is also shown in Figure 11.4. As shown, the temperature discontinuity occurring at the solid-vapour interface is most significant when the heat transfer rate from the solid is high. However, as for the saturated impacts discussed in the
previous chapter, the magnitude of the temperature discontinuity across the solid-vapour interface is at all times an order of magnitude smaller than the temperature difference across the vapour layer, and consequently, the effect of this discontinuity on the heat transfer calculation is not significant.

Figure 11.4 also shows the centreline temperatures at the liquid-vapour interface of the droplet. After the droplet impacts the solid, the temperature of the liquid quickly increases from the initial ambient value, as heat is conducted into the body of the droplet. At approximately 9 ms, the impact process is completed and the droplet begins to move away from the solid. From this point onwards the temperature of the liquid decreases, asymptotically approaching the ambient temperature of the surroundings. The temperature discontinuity occurring across the liquid-vapour interface is at all times small in comparison with the temperature difference across the vapour layer.

The maximum central solid surface temperature reduction calculated by BOUNCE using an initial solid temperature of 300°C was approximately 15.7 K. This value does not compare well with the experimental results presented in Qiao & Chandra [1996], where the maximum temperature drop found during actual impacts on a 300°C surface was approximately 6.9 K. The poor correlation probably results from the difference in temperature measurement methods used in the experimental and simulated impacts.

As discussed in Section 2.2.2, the temperature measurements were made in the experimental impacts of Qiao & Chandra [1996] using a flush mounted commercially available thermocouple. However, it is not entirely clear over what size of area beneath the droplet the thermocouple measured the solid temperature, or how precisely the centre of the droplet impacted the centre of the region in which the solid temperature was measured.

To allow comparison between the simulated and experimental results, four
Figure 11.5: The solid surface temperature drop under a low gravity n-Heptane impact. The initial surface temperature for this simulation was 300°C. The four curves shown represent temperatures averaged over a circle of radius $r_{av}$ on the surface of the solid, concentric with the axis of the droplet.

temperature drop curves are shown in Figure 11.5. Each curve represents the average solid temperature within a circle on the surface of the solid, of radius $r_{av}$, and concentric with the axis of the droplet. Thus, each curve shown in Figure 11.5 is calculated using an average temperature over a finite area of the solid surface.

The first curve shown in Figure 11.5 was calculated using a circle of radius $r_{av} = 0.03$ mm. This curve approximately represents the solid surface temperature drop at the axis of the droplet, and uses the same data as used
for the solid surface temperature curve of Figure 11.4. The remaining three curves were calculated using circles having radii of 0.3 mm, 0.6 mm and 0.9 mm. Note that the initial radius of the droplet in this simulation was 0.75 mm.

The results of Figure 11.5 show that as the area over which the solid surface temperature is measured is increased, the magnitude of the temperature drop during the recollection and rebound phase of the impact decreases. As discussed above, during the recollection phase of the impact, the centre of the droplet is forced towards the solid, which increases heat transfer rates within this region. Consequently, the solid surface temperature at the axis of the droplet decreases significantly during this phase, while the solid surface temperatures in regions away from the axis tend to increase slightly. As a result, the larger the area over which the solid surface temperature is measured, the less impact the central temperature drop has on the average value, and the smaller the average temperature drop is during the recollection phase of the droplet impact. Interestingly, the solid surface temperature drop during the initial impact and expansion phase of the simulation is only weakly affected by the area over which it is averaged.

Comparing the experimental maximum solid temperature drop of 6.9K and the simulated results shown in Figure 11.5, the best correlation between the two occurs when the simulated solid surface temperature is averaged over a circle of radius $r_{av} = 0.6$ mm. This appears to be a physically credible dimension for the thermocouple used in the Qiao & Chandra [1996] experiments. Consequently, the difference in solid surface temperature drops between the experimental and computational impacts can be satisfactorily explained by the different measurement techniques used in the two impacts.
Air Volume Fraction

The centreline vapour layer air volume fraction during the same 300°C n-Heptane impact is shown in Figure [11.6]. During the initial expansion stage of the impact, the air volume fraction decreases rapidly as liquid vaporises at the underside of the droplet and the height of the vapour layer decreases. At around 5 ms from initial impact, the air volume fraction displays some large oscillations, a result of the oscillatory and small height of the vapour layer during this short period.
When the impact process has been completed and the droplet begins to move away from the surface, the air volume fraction rapidly increases again as vapour mixture from the surrounding atmosphere is drawn into the region beneath the rebounding droplet.

**Vapour Layer Energy Transfer**

Figures [11.7] and [11.8] show the centreline energy fluxes occurring within the vapour layer and the total energy transported across the vapour layer, respectively, during the same 300°C low gravity n-Heptane impact.

As shown in Figure [11.7] during the initial impact period, the rate of heat transfer across the vapour layer is high, approaching values of the order $1.5 \times 10^6$ W/m$^2$. This energy flux is almost entirely being conducted into the body of the droplet, and as a consequence, the flux of energy used to vaporise liquid in this period is comparatively negligible. Note that the heat flux across the vapour layer is less oscillatory during the subcooled impacts than during the saturated impacts discussed in the previous chapter.

After a short oscillatory period at around 5 ms from initial impact, the heat flux under the centreline of the droplet again increases substantially, as the re-collection and rebound process begins. While the heat flux during this period is higher than that experienced during the initial impact, as previously discussed, this high rate is largely limited to the small region surrounding the axis of the droplet. Once the droplet leaves the vicinity of the solid surface, the heat transfer rates diminish.

It is interesting that as the droplet begins to move away from the solid surface, the conduction of heat into the body of the droplet is momentarily supported by condensation of vapour from the vapour layer onto the surface of the droplet.
Figure 11.7: Local vapour layer centreline energy fluxes during the simulated low gravity Qiao and Chandra impact. The initial surface temperature for this impact was 300°C. The local energy flux conducted into the body of the droplet during the impact is represented by $q_{\text{drop}}$, the local energy flux consumed by vaporisation of liquid during the impact is represented by $q_{\text{vap}}$, and the local energy flux transported across the vapour layer during the impact is represented by $q_{\text{layer}}$. All fluxes have been smoothed over a time period of 0.1 ms.

When the height of the vapour layer initially begins to increase, the heat flux across the vapour layer starts to decrease. The steep temperature gradients existing in the droplet near the liquid-vapour interface then promote diffusion of heat away from the liquid-vapour interface, tending to decrease
Figure 11.8: Total energy transported during the simulated low gravity Qiao and Chandra impact. The initial surface temperature for this impact was 300°C. The total energy conducted into the body of the droplet during the impact is represented by $e_{\text{drop}}$, the total energy consumed by vaporisation of liquid during the impact is represented by $e_{\text{vap}}$, and the total energy transported across the vapour layer during the impact is represented by $e_{\text{layer}}$.

The temperature at the surface of the liquid. However, the atmosphere adjacent to the liquid is at saturation conditions, and at this time is composed almost entirely of droplet vapour. Consequently, as the liquid temperature starts to decrease, vapour from the layer condenses on the surface of the liquid.
Note that the condensation of vapour during the droplet rebound only lasts for a short duration. This is because, as shown in Figure 11.6, the volume of droplet vapour within the vapour layer available to condense quickly decreases as the droplet rises away from the solid surface.

The total amount of energy lost by the solid during this impact is shown in Figure 11.8 as approximately 0.019 J. The majority of this energy has been used to heat the liquid contained within the droplet. Also, the total droplet volume vaporised during this impact was approximately $8.6 \times 10^{-13}$ m$^3$. Thus, the proportion of the initial droplet volume vaporised, that is the ratio $\eta$ defined by Equation (10.1), was just 0.05%.

### 11.1.3 A Comparison Between the Hydrodynamic Behaviour of Saturated and Subcooled Impacts

It is interesting to compare the initial behaviour of the saturated droplet impacts presented in the previous chapter with the subcooled impacts presented here. After the initial impact, the underside of a saturated droplet moves sharply upward, creating a ‘dome’ of vapour under the axis of the droplet. In the subcooled impacts however, the underside of the droplet remains close to the solid surface after the initial impact, so that the height of the vapour layer appears to be smaller and more uniform than for the saturated impacts. This difference in behaviour can be explained in terms of an energy balance at the lower liquid-vapour interface of the droplet.

When a saturated droplet moves towards a solid surface, pressures in excess of atmospheric generated within the vapour layer allow the temperature of the lower surface of the droplet to increase above the ambient saturation temperature. After impact however, the vapour layer pressure decreases, as the pressure required to sustain the lower droplet surface above the solid
is less than was required to decelerate the droplet during the impact. As the pressure decreases, the temperature of the lower surface of the droplet must also decrease. To facilitate this energy loss, liquid is vaporised, and the lower surface of the droplet is forced away from the solid, creating the visible ‘dome’ of vapour beneath the droplet.

A subcooled impact behaves differently however, because more energy can be conducted into the body of the droplet. When a subcooled droplet impacts a solid surface, the pressures generated within the vapour layer allow the temperature at the lower surface of the droplet to approach or exceed the ambient saturation temperature. As for the saturated impact, once the underside of the droplet has been decelerated, pressures within the vapour layer decrease, and consequently, the temperature at the underside of the droplet must also decrease. However, unlike in the saturated impacts, in the subcooled impacts heat can be conducted into the body of the liquid to decrease the lower droplet surface temperature, rather than being used to vaporise liquid. Subsequently, the lower surface of the subcooled droplet is able to stay closer to the solid, and the vapour layer height appears smaller and more uniform.

11.1.4 The BOUNCE Code and the Different Impact Boiling Regimes

The continuous curve of Figure 11.9 shows the relationship between the maximum solid surface temperature drop during an n-Heptane impact, and the initial temperature of the solid. The curve has been reproduced from the experimental study by Qiao & Chandra [1996], and is applicable to both low gravity and normal gravity impacts conducted with an initial Weber number of 32.
Figure 11.9: The maximum solid surface temperature drop occurring during n-Heptane impacts as a function of the initial temperature of the solid. The curve is taken directly from the study by Qiao & Chandra [1996], and is valid for both low gravity and normal gravity impacts with an initial Weber number of 32. The points indicate results predicted by BOUNCE low gravity simulations.

Figure 11.9 shows that the maximum solid temperature drop during the experimental impacts peaks within an initial solid temperature range of 160°C to 200°C. In the initial temperature range of 200°C to 230°C, the temperature drop decreases sharply with increasing initial temperature, but this decrease gradually relaxes as the initial solid temperature approaches 300°C.
Figure 11.9 also shows solid surface temperature drops calculated by BOUNCE for low gravity n-Heptane simulations performed with initial solid surface temperatures of 300°C, 250°C, 200°C, and 150°C. Each temperature drop was calculated using surface temperatures averaged over a circle of radius $r_{av} = 0.6$ mm, and the remaining aspects of each computation were the same as used for the low gravity n-Heptane impact shown in Figure 11.2.

A comparison between the experimental and simulated results of Figure 11.9 shows that at the initial surface temperature of either 250°C or 300°C, the correlation between the simulated and experimental results is reasonable, but at initial solid temperatures of 200°C and below, the correlation is poor. Interpolating the simulated data between 200°C and 250°C suggests that the initial solid temperature of 230°C marks the temperature boundary between impacts that BOUNCE can simulate thermodynamically, and those it cannot.

The correlation between computational and experimental results shown in Figure 11.9 can be explained in terms of the boiling regimes experienced by the impacting droplets.

That BOUNCE simulates the temperature drop occurring during the 300°C impact is not remarkable. The radius over which the simulated temperature drop was averaged was determined in the previous section to correlate the experimental and computational data at this temperature.

However, that BOUNCE reasonably predicts the solid surface temperature drop down to an initial solid temperature of approximately 230°C demonstrates that BOUNCE is able to predict the heat transfer occurring during subcooled n-Heptane droplet impacts. It also suggests that impacts occurring at initial solid temperatures above 230°C are film boiling impacts, as BOUNCE is a purely film boiling impact code.
At temperatures below 230°C BOUNCE is not able to predict the thermodynamic behaviour of the droplet, because impacts occurring below this initial solid temperature are either transition or nucleate boiling impacts.

As well as showing the impacts that were examined in Section 11.1.1, Figures 11.1 and 1.2 also show impacts that were conducted with initial surface temperatures below 200°C. It is clear from these images that at these lower solid temperatures, vapour bubbles are forming at the liquid-solid interface, and nucleate boiling impacts are occurring. As was discussed in Chapter 1, during nucleate boiling impacts the heat transfer rates experienced at the surface of the solid are far higher than the heat transfer rates experienced during film boiling impacts, because prolonged contact between the liquid and solid phases occurs. Consequently, the experimental temperature drops measured below 200°C are much higher than those predicted by the film boiling BOUNCE code.

As impacts occurring over the initial solid temperature of 230°C are film boiling impacts, and impacts occurring below the initial solid temperature of 200°C are nucleate boiling impacts, we can assume that impacts occurring in the temperature range of 200°C to 230°C are transition boiling impacts. As was discussed in Chapter 1, during transition boiling impacts, direct solid-liquid contact occurs, but only for a short durations.

As shown in Figure 11.9, BOUNCE is not able to predict the thermodynamics of these transition boiling impacts. However, as shown in Section 11.1.1 of this chapter, BOUNCE can simulate the hydrodynamics of these transition boiling impacts.

BOUNCE is predicting the hydrodynamic behaviour of transition boiling impacts because the duration of contact between the solid and liquid phases is much shorter than the timescale of gross droplet deformation. Thus, the
short contacts do not affect the hydrodynamics of the droplet.

However, BOUNCE is not predicting the thermodynamics of the transition boiling impacts because the amount of energy transferred between the solid and liquid phases when contact occurs is large, even if as in this case, the duration of each contact is small. Thus, these small contacts are significantly affecting the heat transfer occurring during the impacts, and consequently BOUNCE is not simulating the thermodynamic behaviour of these droplets.

11.1.5 Vapour Layer Destabilisation

BOUNCE is a film boiling code, and as such, no direct contact between the solid and liquid phases is allowed during a simulation. Consequently, it is not surprising that BOUNCE is not able to model the thermodynamics of transition boiling impacts, or the thermodynamics and hydrodynamics of nucleate boiling impacts.

Determining possible reasons why BOUNCE does not predict solid-liquid contact during these lower temperature impacts is interesting however, for it gives us insight into possible reasons for actual impact vapour layer collapse, and also suggests possible development directions for BOUNCE and other similar two phase heat transfer codes.

With these objectives in mind, three assumptions that were used in the development of the vapour layer code are now examined to determine the relevance each has on vapour layer destabilisation.

Surface Roughness or Contamination

In the development of the vapour layer algorithm, it was assumed that the surface of the solid was perfectly smooth. If however, the impact surface has undulations which have sizes comparable with the thickness of the vapour
layer, then solid spikes may protrude into the liquid phase, promoting locally high heat transfer rates, and thus invalidating calculated vaporisation velocities.

As was discussed in Chapter 9, Section 9.2, a surface which has been finely honed and polished typically has a surface roughness value of $R_a \approx 25 \text{ nm}$. The surface roughness measurement, $R_a$, is defined as [Boundy 1986, p38]

$$...\text{the average value of the departure (both above and below)}$$
$$\text{of the surface from the centre line over a selected sampling or cut-off length.}$$

Thus, the amplitude of the maximum undulations on a finely polished surface would be more than 25 nm, possibly as high as 0.1 $\mu$m.

The stainless steel surface used in the n-Heptane experiments was subject to fine polishing. Thus, it is expected that the maximum amplitude of undulations on the impact surface could be as high as 0.1 $\mu$m.

Figure [11.10] shows the minimum height of the vapour layer during a low gravity n-Heptane impact. The initial surface temperature used for this simulation was 200°C, this being an impact where vapour layer destabilisation is expected to occur. All computational properties used in this simulation were as used in the simulation shown in Figure [11.2].

From Figure [11.10] it appears that the minimum vapour layer height reaches approximately 0.7 $\mu$m during the initial stages of droplet impact. Although this height is an order of magnitude less than the heights experienced during the saturated droplet impacts of the previous chapter, the undulations in the impact surface discussed above are not of this magnitude, so it is unlikely that ‘spikes’ of solid are protruding through the height of the vapour layer and contacting the liquid phase.
Figure 11.10: The minimum height of the vapour layer during a low gravity $n$-Heptane impact. The initial solid temperature for this simulation was $200^\circ$C. The height shown has been smoothed over a time period of 0.1 ms.

However, it is possible that dust or other foreign particles residing on the impact surface could traverse the 0.7 $\mu$m minimum vapour layer height. The experimentalists took many precautions to preclude the presence foreign particles from the impact surface, and the surfaces do appear optically smooth in the images presented in Figures 11.1 and 11.2. However, it is suspected that eliminating sub-micron particles from the impact surface may be difficult. Consequently, no definite conclusions as to the effect of possible surface contaminations on vapour layer destabilisation can be drawn, except that it is recognised that the presence of such contamination may aid vapour layer
Figure 11.11: The maximum Knudsen numbers beneath four low gravity n-Heptane impacts as a function of time. The data shown has been smoothed over a time period of 0.1 ms.

Vapour Layer Molecular Slip Flow

Figure 11.11 shows the maximum Knudsen numbers predicted by BOUNCE beneath four low gravity n-Heptane impacts. Each curve corresponds to an impact calculated with a different initial solid surface temperature, varying from 300°C down to 150°C. The initial Weber number used in these impacts was 32, and all other aspects of these computations are as used in the simulation shown in Figure 11.2.
Referring to Figure 11.11, it is evident that the maximum calculated Knudsen number within the vapour layer in the initial stages of the impact increases as the temperature of the solid surface decreases. With an initial solid temperature of 300°C, the maximum initial Knudsen number calculated is approximately 0.038, while with an initial solid temperature of 150°C, the maximum initial Knudsen number calculated is greater at approximately 0.064. At the lowest possible initial solid temperature before vapour layer collapse occurs, that is 230°C, the maximum initial Knudsen number calculated is approximately 0.05.

The Knudsen numbers shown in Figure 11.11 are determined using a two dimensional coordinate system, and as such, represent the maximum of the average Knudsen numbers evaluated at each radial location. In reality however, the underside of a droplet may show small variations in vapour layer height as one rotates around the axis of the droplet. Thus, the actual maximum Knudsen numbers within the layer may be larger than those calculated by BOUNCE.

Also, the surface roughness discussion of the previous section indicated that at the limit of applicability of the BOUNCE code, surface undulations may be only an order of magnitude smaller than the calculated minimum vapour layer heights. Thus, the presence of surface undulations may also cause the calculated maximum Knudsen numbers to be underpredicted by BOUNCE.

The assumption utilised in the formulation of the vapour layer model was that the fluid flow beneath the droplet was in the molecular slip regime. As discussed in Appendix A, Section A.1, this means that the continuum fluid assumption is used within the body of the vapour layer, but that a molecular slip treatment is included at the interfaces of the vapour layer. It has been
estimated that this regime is valid below a Knudsen number of 0.1, however, this number represents an order of magnitude limit rather than a precise figure [Rohsenow & Choi 1961].

Therefore, as the maximum Knudsen number within the vapour layer during the 230°C n-Heptane impacts is at least 0.05, and as the slip molecular flow regime is valid only below a Knudsen number of the order of 0.1, a reason that the BOUNCE results are not applicable below 230°C may be that the continuum fluid assumption used within the vapour layer is no longer valid in these impacts. It is interesting to note that the Knudsen number of 0.05 was never exceeded in the saturated water impact simulations considered in the previous chapter.

Rayleigh Taylor Instabilities

There could possibly be another reason why the droplet behaviour predicted by BOUNCE is not applicable below the initial surface temperature of 230°C.

An assumption that was implicitly used in the formulation of the vapour layer model was that the lower surface of the droplet was smooth, in comparison with the height of the vapour layer. This assumption may not be valid however, when the height of the vapour layer is very small, and the lower surface of the droplet is being accelerated upward. Under such conditions, Rayleigh-Taylor instabilities may develop at the lower surface of the droplet [Pilch, Erdman & Reynolds 1981], causing ripples to form at the liquid-vapour interface. These ripples may result in direct contact between the solid and liquid.

Classical Rayleigh-Taylor theory predicts that in the initial linear or exponential growth phase of surface waves, only waves with a wavenumber greater than \( k_c \) may grow, as waves having a wavenumber less than this critical value
are stabilised by surface tension [Pilch et al. 1981, Chandrasekhar 1961]. Thus, for stability of the surface we require

\[ k \leq k_c. \] (11.1)

The wavenumber is defined by

\[ k = \frac{2\pi}{\lambda}, \] (11.2)

where \( \lambda \) is the wavelength of the disturbance. The critical wavenumber is given by [Pilch et al. 1981, Chandrasekhar 1961]

\[ k_c = \left(1 - \varepsilon \right) \frac{\rho_f a}{\sigma} \right)^{\frac{1}{2}}, \] (11.3)

where in this analysis \( a \) is the acceleration of the lower surface of the droplet, \( \rho_f \) is the density of the liquid, \( \sigma \) is the surface tension of the liquid, and

\[ \varepsilon = \frac{\rho_m}{\rho_f}, \] (11.4)

where \( \rho_m \) is the density of the vapour mixture beneath the droplet. In this application, \( \varepsilon \approx 0 \), as the density of the liquid is far greater than the density of the vapour. Note that the critical wavenumber given by equation (11.3) is independent of the viscosity of the liquid [Chandrasekhar 1961].

Substituting equations (11.2), (11.3) and (11.4) into equation (11.1) and rearranging gives,

\[ a \leq \frac{\sigma}{\rho_f} \left( \frac{2\pi}{\lambda} \right)^2. \] (11.5)

Equation (11.5) gives the range of accelerations that the lower surface of the droplet may experience before a surface wave of wavelength \( \lambda \) will grow.

The VOF fluid dynamics model used by BOUNCE can resolve surface geometries down to the size of the VOF cell dimensions. Thus, if instabilities are developing on the surface of the droplet which are not observed in the
computations, then their dimensions must be of the order or smaller than the computational cell size.

In the low gravity n-Heptane simulation shown in Figure 11.2, each computational cell had the radial dimension of 30 \( \mu \text{m} \). Substituting this value and the properties of n-Heptane into equation (11.5) gives the maximum acceleration that the lower surface of the droplet can experience before surface instabilities occur as,

\[
a_c = 1.3 \times 10^6 \text{ m/s}^2, \tag{11.6}
\]

where the subscript \( c \) denotes a critical value.

Calculations performed on the low gravity n-Heptane impact of Figure 11.2 conservatively showed that the maximum acceleration experienced by the underside of the droplet was less than \( 10^5 \) m/s\(^2\) at all times, and during the highest acceleration periods, the duration of the acceleration was always extremely small. Thus, the level of acceleration experienced by the surface of the droplet was always less than the critical level given by equation (11.6), and we can subsequently conclude that Rayleigh-Taylor instabilities are not affecting the geometry of the lower surface of the droplet.

**Vapour Layer Destabilisation**

The above discussion has highlighted the main assumptions used by BOUNCE that may become invalid when collapse of the n-Heptane vapour layer is imminent. It was found that it is unlikely that surface roughness, or Rayleigh-Taylor instabilities are important factors when predicting such vapour layer collapse.

What did appear to be important in predicting such behaviour was the correct modelling of the gas flow regime within the vapour layer. The maximum Knudsen number experienced during a n-Heptane impact that was on
the boundary between the film and transition boiling impact regimes was 0.05. As discussed previously, this Knudsen number may indicate that the molecular slip flow regime is no longer valid within the vapour layer. Thus, in order to correctly predict vapour layer collapse and thus transition boiling impacts, a transition or rarefied gas molecular flow analysis within the vapour layer may be required.

It has also been recognised that contamination of the impact surfaces in the examined n-Heptane impacts may have promoted vapour layer collapse, another factor not considered in the BOUNCE simulations.

11.2 Subcooled Water Droplet Impacts

11.2.1 The Qiao and Chandra Water Impacts

As well as performing droplet impact experiments with n-Heptane, Qiao & Chandra [1996] also performed subcooled droplet impact experiments with water. These experiments were previously detailed in Chapter 2, Section 2.2.2.

Figure 11.12(c) shows a subcooled water impact presented in Qiao & Chandra [1996] which was performed with an initial solid temperature of 320°C. Figure 11.13 shows the BOUNCE simulation of the same impact. The BOUNCE simulation was performed using a computational domain of 0 to 2.5 mm radially, and −0.1 to 7.9 mm vertically, composed of 50×160 equally sized square cells. The code required approximately 14,000 iterations to produce the 17 ms of simulation.

Comparing the computational and experimental images of Figures 11.13 and 11.12(c), respectively, it is clear that BOUNCE does not replicate the hydrodynamic behaviour of the water droplet. Within the first millisecond of
Figure 11.12: Three low gravity water impacts conducted at different initial solid temperatures. Case (a) in the text refers to the 150°C impact, Case (b) in the text refers to the 250°C impact and Case (c) in the text refers to the 320°C impact. (Taken from Qiao & Chandra [1996].)
Figure 11.13: Simulated droplet impact replicating the Qiao and Chandra water low-gravity experiment. The initial solid temperature was $320^\circ$C.
the experimental droplet impacting the solid, small bubbles of vapour have
developed at the lower surface of the droplet. These bubbles indicate that
nucleate boiling is occurring within the droplet, and suggest, as in the n-
Heptane impacts conducted at initial solid temperatures below 200°C, that
the vapour layer beneath the experimental droplet has collapsed. This be-
haviour is beyond the capabilities of the BOUNCE code, and consequently,
the comparison between the dynamics of the simulated and experimental
droplets is poor.

Figure 11.14 shows the maximum Knudsen number within the vapour
layer during the simulated water impact described above, and Figure 11.15
the minimum vapour layer height during the same impact. Both results have
been smoothed over a time period of 0.1 ms.

The maximum Knudsen number within the vapour layer reaches a level
of approximately 0.21 during the initial stages of the subcooled water im-
pact. This maximum is significantly beyond the range of the molecular slip
flow regime. As a result, the assumptions utilised in the development of
the BOUNCE vapour layer code are not valid in this simulation, and the
poor correlation between the results of Figures 11.13 and 11.12(c) is to be
expected.

It is interesting that the minimum vapour layer height during the initial
stages of the same subcooled water impact, as shown in Figure 11.15
reaches approximately 0.5 µm. This height is slightly less than the minimum vapour
layer heights experienced in the subcooled n-Heptane impacts described in
the previous section, but it is still considerably larger than the level of surface
roughness of the solid, which was calculated in Section 11.1.5 to have a
maximum of approximately 0.1 µm. Thus, surface irregularities of the impact
material are not the reason that BOUNCE is unable to predict the behaviour
Figure 11.14: The maximum Knudsen number within the vapour layer during a low gravity water impact. The initial solid temperature for this simulation was 320°C. The data shown has been smoothed over a time period of 0.1 ms.

of the impacting droplet, but again we recognise that contaminations on the impact surface could possibly promote vapour layer collapse.

It is also interesting that although the minimum vapour layer heights experienced during the n-Heptane impacts of the previous section and the water impacts of this section are of a similar magnitude, the Knudsen numbers experienced during the two impacts are not.

The n-Heptane molecule, which has the chemical form CH$_3$(CH$_2$)$_5$CH$_3$, has a larger molecular collision volume than the water molecule, for as well as containing a larger number of atoms, it has the structure of a long chain.
Thus, given a certain density of gaseous molecules within a given volume, if the gas were n-Heptane, there would be far more molecular collisions experienced within a period of time than if the gas were water. Consequently, at the same temperature and pressure, the molecular free path of n-Heptane is much smaller than the free path of water.

The Knudsen number, as defined by equation (A.1), is the ratio of the molecular mean free path of the vapour mixture to the height of the vapour layer. Thus, the Knudsen number increases with an increase in the mean free path of the vapour. As the molecular mean free path of water is greater than
the mean free path of n-Heptane under the same conditions, the Knudsen numbers experienced within the vapour layer under a water droplet are higher than those experienced under a similar n-Heptane droplet. This is one reason why BOUNCE is more likely to be able to predict the impact of a subcooled n-Heptane droplet than the impact of a subcooled water droplet.

Figure 11.16 shows the maximum solid surface temperature drop experienced under subcooled water impacts performed by Qiao & Chandra [1996]. Also shown in Figure 11.16 is the maximum temperature drop calculated by a BOUNCE simulation using an initial solid temperature of 300°C. The parameters used in this simulation were the same as those used for the impact shown in Figure 11.13 except for the small change in initial solid temperature. The surface temperature used in calculating the BOUNCE result of Figure 11.16 was averaged over a circle on the surface of the solid, of radius $r_{av} = 0.6$ mm.

The correlation between the experimental and computational results of Figure 11.16 is poor, with the experimental temperature drop being an order of magnitude larger than the simulated temperature drop. This difference is to be expected, since the nucleate boiling impact experienced by the experimental droplet involves far higher heat transfer rates than the film boiling impact predicted by BOUNCE.

In the interests of exploring the range of impact conditions with which the BOUNCE code can be used, we now consider an experimental subcooled water droplet impact which was conducted with a significantly higher initial solid temperature. As was shown in Section 11.1.5 the higher the initial solid temperature, the lower the vapour layer Knudsen numbers produced during the impact. Thus, it is more likely that the impact considered in the next section will occur in the film boiling regime, and the results of BOUNCE be
Figure 11.16: The maximum solid surface temperature drop occurring during subcooled water impacts as a function of the initial temperature of the solid. The curve is taken directly from the study by Qiao & Chandra [1996], and is valid for both low gravity and normal gravity impacts. The point indicates a simulated result.

11.2.2 The Chen and Hsu Impact

The experimental subcooled water impact considered in this section was performed by Chen & Hsu [1995]. The experimental conditions for this impact were previously outlined in Chapter 2, Section 2.2.2.

A record of the impact dynamics of the water droplet was not taken.
during the Chen & Hsu [1995] experiments, however, Figure 11.17 shows the solid surface temperatures recorded during the 500°C impact, and also the local heat transfer rates calculated by the authors from the same data.

Figure 11.18 shows the droplet centreline solid surface temperatures calculated by BOUNCE during a simulation of the Chen and Hsu 500°C impact, and Figure 11.19 the calculated solid centreline energy fluxes for the same impact. The BOUNCE simulations were performed over a computational domain of 0 to 7.0 mm radially and −0.2 to 6.8 mm axially, which was composed of 70×70 square, computational mesh cells.

Comparing the heat transfer results of Figures 11.17, 11.18 and 11.19, it is evident that BOUNCE is significantly underestimating the heat transfer across the vapour layer in this subcooled impact. The temperature drop at
the surface of the experimental impact material is over an order of magnitude larger than the same temperature drop predicted by BOUNCE, and the heat transfer rates from the solid at the axis of the droplet are on average also an order of magnitude higher in the experimental impact than in the simulated impact. These differences suggest than the vapour layer has collapsed in the experimental impact, and that the vapour layer assumptions on which the BOUNCE algorithm is based are not applicable during this impact.

Further evidence of the inapplicability of the BOUNCE simulations dur-
The centreline energy flux from the solid as predicted by BOUNCE during the Chen and Hsu subcooled water impact. The data has been smoothed over a time period of 0.1 ms.

The maximum Knudsen number within the layer, as calculated during the BOUNCE simulation. As shown, this non-dimensional number approaches a maximum of 0.125 during the initial stages of the Chen and Hsu impact. Although lower than the maximum Knudsen number found during the subcooled water impacts of Qiao & Chandra [1996], this number still indicates that the flow within the vapour layer is not always in the molecular slip regime, and reinforces the inapplicability of the BOUNCE Chen and Hsu simulation.
It has been found that BOUNCE is unable to predict any of the available significantly subcooled water droplet experiments, because for all the cases available, vapour layer collapse appears to occur during the impact process. The primary assumption that appears to invalidate the BOUNCE results when vapour layer collapse is imminent is the assumption of a molecular slip flow regime existing within the vapour layer, although it is recognised that surface contaminations, which are not modelled by BOUNCE, may also contribute to the destabilisation of the vapour layer.

In the interests of determining the relevance of BOUNCE to subcooled
water impacts, the Chen & Hsu [1995] impact simulations were repeated with a variety of experimental conditions changed. It was found that either of the following changes to the impact conditions would produce a simulation which did not violate the molecular slip flow assumption in the vapour layer, and may consequently be valid:

1. Reducing the subcooling of the droplet from 80K to 25K.

2. Reducing the impact velocity of the droplet from 1 m/s to 0.2 m/s.

Thus, it is expected that BOUNCE may be able to simulate a significant number of subcooled water impacts, although there does not appear to be data presently available to confirm this.
Chapter 12

Conclusions and Future Work

The process of a droplet impacting with a hot solid surface occurs in a wide variety of industrial, environmental and domestic applications. When the temperature of the solid is high enough, film boiling droplet impacts occur, which are defined as those in which there is no direct contact between the impacting droplet and hot solid surface. Film boiling impacts are characterised by low heat transfer and vapour production rates.

In this thesis a computational fluid dynamics code, termed BOUNCE, has been written to simulate the hydrodynamic and thermodynamic behaviour of film boiling droplet impacts. BOUNCE is based around two fluid dynamics algorithms: A Volume of Fluid (VOF) algorithm used to model the hydrodynamics of the fluid contained within the droplet, and a separate one dimensional algorithm used to simulate the flow within the viscous vapour layer existing between the droplet and the solid surface.

The VOF algorithm employed in BOUNCE is loosely based on the SOLA-VOF code of Nichols et al. [1980], but also employs features from the RIPPLE code of Kothe et al. [1994] and other features developed in this study. Notable amongst the features employed from the RIPPLE code are a Contin-
uum Surface Force (CSF) surface tension treatment [Brackbill et al. 1992], and a solution to the Navier-Stokes equations using the Pressure Poisson Equation (PPE) method of Kothe et al. [1994]. Features developed in this study include two new VOF advection schemes, the Defined Donating Region (DDR) and Stream schemes, an implicit treatment of momentum advection, viscous stresses and surface location, and coupling of the viscous vapour layer solution to the main VOF code.

Both the DDR and Stream advection schemes developed in this study use a linear piecewise method for reconstructing the fluid free surfaces, coupled to a fully multidimensional method for integration of cell boundary fluxes. The DDR scheme has been shown to be an accurate and efficient VOF advection algorithm, particularly suited to complex flow problems, such as the studied case of impacting droplets, where maintaining the stability and discrete nature of the free surface interface is paramount. The DDR scheme has been applied to two dimensional Cartesian and cylindrical coordinates.

The Stream scheme has been shown to be marginally more accurate than the DDR scheme, but also more computationally expensive. However, one of the chief advantages of the Stream scheme is its simplicity. While only applied to two dimensional Cartesian coordinates in this study, it is expected that the Stream scheme would be the easiest of the currently available fully multidimensional algorithms to apply to alternative coordinate systems.

The viscous vapour layer code models the movement of fluid within the vapour layer existing beneath the droplet, as well as the transport of thermal energy within the liquid, solid and vapour phases. Simplification of the Navier-Stokes equations within the layer allows one-dimensional parabolic flow to be assumed, and a molecular slip flow treatment allows for an effective velocity discontinuity to occur at both the liquid-vapour and solid-vapour
interfaces. A transport equation for the air volume fraction within the vapour layer is solved.

Within the solid phase, thermal energy conduction in the axial direction is solved, while in the liquid phase, both conduction in the axial direction and convection in both the axial and radial directions are solved. The heat transfer across the vapour layer is assumed to be dominated by one dimensional conduction, and a molecular interface treatment at the liquid-vapour and solid-vapour interfaces allows for effective temperature discontinuities to occur at both.

The BOUNCE code has been validated against several well-documented experimental droplet impact studies. In simulating saturated droplet impacts, BOUNCE has performed well. A comparison with the hydrodynamical experiments of Wachters & Westerling [1966] has shown that BOUNCE is able to predict the hydrodynamics of impacting saturated droplets accurately, if the Weber number, a non-dimensional velocity, is less than 30. Within the Weber number range of 30 to 80, BOUNCE is able to predict the impact process accurately, but not the breakup of the droplet during rebound. For impacts with a Weber number greater than 80, BOUNCE can predict the initial impact of the main body of the droplet, but not the early formation of satellite droplets or subsequent droplet disintegration that tends to occur in such high velocity impacts. It has been shown that the two dimensional coordinate system employed by BOUNCE limits the ability of the code to predict high velocity impacts.

A comparison of the solid surface temperatures predicted by BOUNCE with those measured during the saturated water droplet impact of Groendes & Mesler [1982] has shown that BOUNCE can accurately predict the heat transfer, and thus thermodynamic behaviour, of impacting saturated
In validating the accuracy of BOUNCE in simulating subcooled droplet impacts, experimental impacts with the liquids n-Heptane and water have been used. BOUNCE is able to simulate the hydrodynamical behaviour of n-Heptane impacts performed by Chandra & Avedisian [1991] and Qiao & Chandra [1996], in both low gravity and normal gravity environments. Also, BOUNCE is able to simulate the thermodynamic behaviour of impacting n-Heptane droplets when the impacts occur in the film boiling regime. BOUNCE was not able to model any available subcooled water impacts, as it has been shown that during all documented impacts significant contact between the liquid and solid phases is occurring.

While not intended to model impacts outside the film boiling regime, the primary assumption which limits the ability of BOUNCE to predict lower temperature impacts is the assumption that the vapour layer flow is within the molecular slip regime at all times. Simulations suggest that this assumption may not be correct when vapour layer collapse is imminent, suggesting that a transition molecular flow treatment within the vapour layer may be needed to model transition boiling droplet impacts. Also, while not quantified, it has been recognised that contamination of the impact surface may also lead to vapour layer collapse, also invalidating the results of the BOUNCE code.

This thesis has suggested many areas for further research. The Stream VOF advection algorithm could benefit from further code optimisation, and requires application to other coordinate systems. Also, as outlined previously, modifications to the Stream scheme comprising of a continuous velocity field and exact donating region integration technique could possibly produce the ultimate VOF advection scheme, although it is recognised that such modifi-
cations may be analytically and computationally complex.

The ability of BOUNCE to simulate impacting droplets requires further validation. This is particularly true of subcooled water droplets, where presently no film boiling impact data are available for comparison. With this objective in mind, the author suggests that further water droplet impact experimental studies be performed, particularly involving droplets with low levels of subcooling, low impact velocities or high impact surface temperatures, and particularly thermodynamical studies where heat transfer rates from the impact surface are measured.

Finally, the BOUNCE code could be extended to model a greater variety of impacts. A porous treatment of the impact material could be implemented for modelling droplet impacts on surfaces such as wood and char. Using a three dimensional coordinate system, rather than the two dimensional system presently used, could allow higher velocity impacts to be simulated. Also, BOUNCE could be extended to model impacts within the transition boiling impact regime, although this may be prohibitively difficult given the presently somewhat limited knowledge of film boiling destabilisation behaviour.
Bibliography


Leidenfrost, J. G. [1756], De Aquae Communis Nonnullis Qualitatibus Tractus (A Tract About some Qualities of Common Water), Duisburg on Rhine. An original copy of this treatise is in the Yale University Library.


Patterson, G. N. [1956], *Molecular Flow of Gases*, John Wiley and Sons.


Appendix A

Kinetic Theory Analysis of Vapour Region Boundaries

In this appendix, a molecular treatment of conditions existing at vapour region boundaries is detailed. The solid-vapour interface treatment is mainly a review of work accomplished by others, as is the mass balance analysis at a liquid-vapour interface. The temperature discontinuity and slip velocity analysis at a liquid-vapour interface, where evaporation and condensation are assumed to occur, has been developed in this study.

A.1 Molecular Flow Regimes

As briefly discussed in Section 8.2.1, the Knudsen number is defined as the ratio of the mean free path of a gas to a dimension of the system in which the gas is contained. The magnitude of the Knudsen number determines whether using the continuum assumption to describe the behaviour of a gas is appropriate. From Rohsenow & Choi [1961], when gas velocities are small compared with the speed of sound, as they are in this study, gas dynamics
can be loosely classified into four regimes:

1. Continuum regime (Kn < 0.01). At low values of the Knudsen number, the mean free path of the gas is small compared with the dimensions of the container, and the gas may be approximated as a continuous medium.

2. Slip flow regime (0.01 < Kn < 0.1). In this range, the gas may be regarded as a continuum at several mean free paths distance from any adjoining medium, but near any interface, a kinetic theory treatment should be used to describe the relationship between the gas and adjoining medium properties.

3. Transition regime (0.1 < Kn < 3.0). This region poses the most difficulty in simulating. Here, the mean free path of the gas has comparable length to the dimensions of the system, and collisions between molecules and other molecules and collisions between molecules and the system boundaries assume equal importance in determining gas behaviour.

4. Free-molecule regime (Kn ≥ 3.0). In this range the mean free path of the gas is large compared with the dimensions of the system. Thus, collisions between molecules and other molecules occur infrequently, the majority of molecules having experienced their last collision with a boundary of the gas region. Flow solutions in this regime are performed using rarefied gas kinetic theory [Knudsen 1950, Loeb 1934, Kennard 1938].

In this work, the Knudsen number for the vapour layer is defined as

$$\text{Kn} (r) = \frac{\lambda_{m,a} (r)}{\delta (r)},$$  \hspace{1cm} (A.1)
where \( \lambda_{m,a}(r) \) is the average mean free path of the vapour mixture, evaluated at the average temperature within the vapour layer at radial position \( r \).

In this study we limit the applicability of the droplet impact model to levels of the Knudsen number below 0.1 because of the difficulties involved in modelling gas dynamics within the transition regime. As shown in Chapters 10 and 11, this allows simulation of a significant number of real hot surface droplet impacts. In this chosen range, vapour flow within the viscous layer is in the slip regime, and a kinetic treatment must be used to model gas behaviour near the solid-vapour and liquid-vapour interfaces. It is this kinetic treatment of gas dynamics at vapour layer interfaces which is now examined.

\section{A.2 A Molecular Treatment of the Solid-Vapour Interface}

The kinetic theory analysis of conditions existing at a solid-vapour interface follows closely the analysis by Chapman & Cowling [1970].

\subsection{A.2.1 Interface Slip Velocity}

Figure A.1 shows an interface between a solid and gas region. In this diagram, the \( x \) coordinate is directed normal to the interface, and the \( y \) coordinate tangential to the interface. Velocity in the \( y \) direction, tangential to the interface, is represented by \( v \). The velocity of the wall is \( v_w \), the average tangential velocity of molecules impacting the wall is \( v_i \), and the average tangential velocity of molecules rebounding from the wall is \( v_r \). The average tangential velocity of all gas molecules at the interface is \( v_o \), so that the magnitude of the slip velocity at the interface is \( v_o - v_w \).
In general, the tangential velocity of molecules impacting the wall, $v_i$, is not equal to the average tangential molecular velocity at the wall, $v_o$, because molecules impacting the wall experienced their last collisions at some distance to the right of the wall, and it is from these collisions that they have gained their tangential velocity. Thus, the tangential velocity of molecules impacting the wall can be represented as,

$$v_i = v_o + U\lambda \frac{dv}{dx}, \quad (A.2)$$

where the distance $U\lambda$ is the characteristic distance to the right of the interface from which the impacting molecules gained their tangential velocity, and $\lambda$ is the mean free path of the gas.

In a simple gas, molecules experience their last collision before impacting the wall at an average distance of $\frac{2}{3}\lambda$ from the wall. The value of $U$ in equation (A.2) is greater than $\frac{2}{3}$ however, because more than a single collision is generally required to bring a molecule into equilibrium with its surroundings. A
detailed Chapman-Enskog analysis for the magnitude of $U$ is given in Chapman & Cowling [1970], where it is found that for spherical rigid molecules, $U = 0.998$, or approximately unity. For polyatomic molecules, this value represents a reasonable approximation for momentum calculations, as the average momentum of molecules tangential to an interface is not dependent on the internal energy of individual molecules [Chapman & Cowling 1970].

The average tangential velocity of molecules rebounding from the interface, $v_r$, is not in general equal to the velocity of the solid, $v_w$. This is because molecules impacting the surface may reflect in a manner varying from diffusely to specularly, depending on the molecular geometry of the interface. To quantify this behaviour, we define the Specular Reflection Coefficient [Rohsenow & Choi 1961] as,

$$\sigma_v = \frac{v_i - v_r}{v_i - v_w}.$$  \hspace{1cm} (A.3)

The Specular Reflection coefficient is dependent on the combination of vapour and solid interface type. It may be interpreted as the proportion of molecules impacting the solid surface which reflect diffusely from the surface, the remainder reflecting specularly. For the majority of solid vapour interfaces, this coefficient has a value near unity [Rohsenow & Choi 1961].

From Newton’s Second Law, the shear stress imposed on the solid surface is equal to the rate of change of tangential momentum at the surface. Noting that only molecules who reflect diffusely at the interface transfer momentum with the solid, we find,

$$\tau_{xy} = \sigma_v J (v_i - v_w),$$  \hspace{1cm} (A.4)

where $J$ is the total mass rate of molecules [kg/m$^2$s] impacting the solid surface. Combining equations (A.2) and (A.4) gives for the shear stress
imposed at the solid interface,

\[ \tau_{xy} = \sigma_v J \left( v_o + U \lambda \frac{dv}{dx} - v_w \right). \]  

(A.5)

To determine the value of the slip velocity at the interface, we equate the value for the actual shear stress at the interface, given by equation (A.5), to the shear stress existing at a plane in a Newtonian fluid,

\[ \tau_{xy} = \mu \frac{dv}{dx}, \]  

(A.6)

giving,

\[ \mu \frac{dv}{dx} = \sigma_v J \left( v_o + U \lambda \frac{dv}{dx} - v_w \right). \]  

(A.7)

The mass flow rate of molecules through a plane is given by Kinetic Theory [Chapman & Cowling 1970] as

\[ J = \frac{1}{4} \rho \bar{c}, \]  

(A.8)

where \( \bar{c} \) is the mean molecular speed of the gas. Further, the coefficient of viscosity is given by kinetic theory [Chapman & Cowling 1970] as

\[ \mu = \frac{1}{2} U \rho \bar{c} \lambda. \]  

(A.9)

Substituting equations (A.8) and (A.9) into equation (A.7) and rearranging gives the slip velocity at an interface,

\[ v_o - v_w = \lambda \left( \frac{2 - \sigma_v}{\sigma_v} \right) \frac{dv}{dx} \]  

(A.10)

where we have used the previously discussed Chapman-Enskog result, \( U \approx 1.0. \)

### A.2.2 Interface Temperature Discontinuity

The interface temperature discontinuity analysis at a solid-vapour interface is similar to the interface slip velocity analysis presented above.
Figure A.2 shows temperatures at a solid-vapour interface. The temperature of the gas is represented by $T(x)$, where $x$ is a coordinate directed normal to the interface. In an analogous manner to the velocity slip calculation, we define the temperature of the wall as $T_w$, the average temperature of molecules impacting the solid as $T_i$, the average temperature of molecules rebounding from the solid as $T_r$, and the average temperature of all gas molecules at the interface as $T_o$.

The average temperature of molecules impacting the solid is determined using an analogous equation to the impacting tangential velocity equation \((A.2)\). Thus,

$$T_i = T_o + U'\lambda \frac{dT}{dx},$$

and $U'\lambda$ is the characteristic distance from the interface from which the molecules gained their temperature. The characteristic tangential velocity distance, $U\lambda$, is not equal to the characteristic temperature distance, $U'\lambda$, as molecules with a higher temperature have greater translational energy, and
can thus travel further between collisions [Chapman & Cowling 1970].

Defining $f$ as the ratio of characteristic distances, we have

$$f = \frac{U'}{U} \approx U', \quad (A.12)$$

where the Chapman-Enskog value for $U$ has again been used. Results from the Chapman-Enskog analysis [Chapman & Cowling 1970] for the transport of molecular temperature give $f = U' = \frac{5}{2}$ for spherical rigid molecules. Calculating the characteristic temperature distance for polyatomic molecules is more difficult, however, because molecular energy is composed of not only translation, but also internal molecular energy. Eucken’s formula [Chapman & Cowling 1970] gives an approximation for $f$ for polyatomic molecules as

$$f = U' = \frac{9\gamma - 5}{4}, \quad (A.13)$$

where $\gamma$ is the ratio of specific heats of the gas,

$$\gamma = \frac{c_p}{c_v}. \quad (A.14)$$

The temperature of molecules rebounding from the solid, $T_r$, may not be equal to the temperature of the solid, $T_w$. In an analogous manner to the Specular Reflection coefficient, we define the Thermal Accommodation coefficient [Knudsen 1950, Chapman & Cowling 1970] for a solid-vapour combination as

$$\sigma_t = \frac{T_i - T_r}{T_i - T_w}. \quad (A.15)$$

This may be interpreted as the proportion of impacting molecules who come into thermal equilibrium with the solid during the impact, the remainder having their temperatures unchanged by the impact. Thus, the rate of energy transfer from the gas to the solid [W/m$^2$] is given by,

$$q_s = \sigma_t J \left( H_g (T_i) - H_g (T_w) \right), \quad (A.16)$$
where $J$ is again the mass rate of molecules impacting the surface, and the gas phase enthalpies, $H_g$, are evaluated at the pressure at the interface.

In the analysis of Chapman & Cowling [1970], a difference in internal energy, rather than enthalpy, was used in equation (A.16) to calculate heat transfer into the solid. In general, the molecular density of molecules approaching the interface is different to the molecular density of molecules rebounding from the interface, as the temperatures of the two streams are not equal. As the density of the gas changes during impact, flow work energy is liberated from, or required by, the change in volume of the gas. This work is accounted for by using enthalpy in equation (A.16), rather than internal energy.

Combining equations (A.11) and (A.16), the energy flux into the solid is given by

$$q_s = \sigma_t J c_p \left( T_o + U' \lambda \frac{dT}{dx} - T_w \right),$$

(A.17)

where the enthalpy change for an ideal gas, given by

$$H_g (T_2) - H_g (T_1) = c_p (T_2 - T_1),$$

(A.18)

was used.

Using Fourier’s law for the rate of heat transfer through a medium by conduction, the energy flux into the solid is

$$q_s = k \frac{dT}{dx},$$

(A.19)

where $k$ is the thermal conductivity of the gas. Equating equations (A.17) and (A.18), we find

$$k \frac{dT}{dx} = \sigma_t J c_p \left( T_o + U' \lambda \frac{dT}{dx} - T_w \right).$$

(A.20)

The mass rate of molecules impacting the solid, $J$, is given by equation (A.8). Also, from kinetic theory [Chapman & Cowling 1970], the thermal
conductivity for a simple gas is given by

$$k = \frac{1}{2} \rho c_v U' \lambda c.$$  \hspace{1cm} (A.21)

Combining equations (A.8), (A.20) and (A.21) and rearranging gives,

$$T_o - T_w = U' \lambda \left( \frac{2}{\gamma \sigma_t} - 1 \right) \frac{dT}{dx}.$$  \hspace{1cm} (A.22)

Making use of Eucken’s formula for polyatomic gases, equation (A.13), the temperature discontinuity existing at a solid-vapour interface is determined as

$$T_o - T_w = \lambda \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_t} - 1 \right) \frac{dT}{dx}.$$  \hspace{1cm} (A.23)

### A.3 A Molecular Treatment of the Liquid-Vapour Interface

The primary difference between a solid-vapour and liquid-vapour interface is that at a liquid-vapour interface, gas may be produced from or lost to the liquid phase. Thus, the rate at which vapour molecules impact a liquid interface may not equal the rate at which vapour molecules leave the interface.

In this section, we develop a kinetic treatment for gas behaviour at a liquid-vapour interface. Firstly, an analysis is conducted to determine the rate at which molecules impact and leave a liquid interface. A slip velocity and temperature discontinuity analysis is then performed at a liquid-vapour interface, following closely the methods employed in the previous solid-vapour interface analysis, described in Section A.2.

#### A.3.1 Interface Mass Balance

Figure A.3 shows schematically the evaporation and condensation processes occurring at a liquid-vapour interface. The mass rate of molecules impact-
Figure A.3: Molecular evaporation and condensation rates at a liquid-vapour interface.

ing the liquid is $J_i$, the mass rate of molecules rebounding from the liquid without penetrating the surface is $J_r$, and the mass rate of molecules actually condensing into the liquid is $J_c$. The mass rate of molecules evaporating from the surface is $J_e$.

From simple kinetic theory [Rohsenow & Choi 1961, Collier 1981], the rate at which molecules pass through a plane is given as

$$J = \frac{1}{4} \rho c = \sqrt{\frac{M}{2\pi R}} \frac{P}{\sqrt{T}},$$  \hspace{1cm} (A.24)

where $M$ is the molecular weight of the gas [kg/kmol] and $R$ is the universal gas constant [J/kmol.K]. Thus, the rate at which molecules impact a liquid surface is given by

$$J_i = \sqrt{\frac{M}{2\pi R}} \frac{P}{\sqrt{T_i}}.$$  \hspace{1cm} (A.25)

Note that although the impacting molecules are on average at a temperature $T_i$, the frequency with which they impact the liquid surface is determined by
the density of molecules in the vicinity of the interface. Thus, $T_o$, the average vapour molecular temperature at the liquid interface, is used in equation (A.25).

We again define a Thermal Accommodation Coefficient for the surface, $\sigma_t$, but the definition for this coefficient is subtly different to the definition given in Section A.2.2 for the solid-vapour interface. The Accommodation Coefficient is here defined as the proportion of impacting molecules who condense into the liquid phase upon impact, the remainder are assumed to rebound from the surface without altering their kinetic energies. Thus, the mass rate of molecules condensing into the liquid is given by

$$J_c = \sigma_t J_i = \sigma_t \sqrt{\frac{M}{2\pi R}} \frac{P}{\sqrt{T_o}}, \quad (A.26)$$

and conversely, the mass rate of molecules rebounding from the surface is given by

$$J_r = (1 - \sigma_t) J_i = (1 - \sigma_t) \sqrt{\frac{M}{2\pi R}} \frac{P}{\sqrt{T_o}}. \quad (A.27)$$

The total mass rate of molecules evaporating from a liquid-vapour interface, $J_t$, is then given as

$$J_t = J_e - J_c = J_e - \sigma_t \sqrt{\frac{M}{2\pi R}} \frac{P}{\sqrt{T_o}}. \quad (A.28)$$

If a liquid is in dynamic equilibrium with a vapour, then the net rate of molecules evaporating is zero, the temperature of the liquid is equal to the temperature of the vapour and the pressure within the liquid is equal to the saturation pressure at the vapour temperature. Thus, in dynamic equilibrium, equation (A.28) becomes,

$$J_t = 0 = J_e - \sigma_t \sqrt{\frac{M}{2\pi R}} \frac{P_{sat}(T_w)}{\sqrt{T_w}}, \quad (A.29)$$
where \( T_w \) is the temperature of the liquid surface. Thus, in dynamic equilibrium, the evaporating mass flux is given by

\[
J_e = \sigma_t \sqrt{\frac{M}{2\pi R} \frac{P_{\text{sat}}(T_w)}{\sqrt{T_w}}}.
\] (A.30)

A commonly used assumption in molecular condensation theories, which we will employ here, is that the evaporation mass flux is determined solely by conditions existing within the liquid at the liquid-vapour interface [Collier 1981]. Thus, the liquid evaporation mass flux is given by equation (A.30), irrespective of whether the interface is in equilibrium or not, and is a function solely of the temperature of the liquid at the liquid vapour interface. Note that the pressure within the liquid is not necessarily equal to the saturation pressure corresponding to the liquid temperature — the saturation pressure in equation (A.30) has been empirically determined to equate evaporation and condensation mass fluxes when the interface is in dynamic equilibrium.

Finally, combining equations (A.28) and (A.30) gives the total evaporating mass flux at a liquid-vapour interface as

\[
J_t = \dot{\omega} \rho_v = \sigma_t \sqrt{\frac{M}{2\pi R} \left( \frac{P_{\text{sat}}(T_w)}{\sqrt{T_w}} - \frac{P}{\sqrt{T_o}} \right)},
\] (A.31)

where use has been made of the droplet vaporisation velocity, \( \dot{\omega} \). The pressure \( P \) in equation (A.31) is the pressure at the liquid-vapour interface. Note that as mean fluid velocities normal to the interface are small compared with the speed of sound, there is no pressure discontinuity resulting from evaporation processes across the liquid-vapour interface.

### A.3.2 Interface Slip Velocity

In Section A.2.1, an analysis was conducted to determine the momentum exchange rate between a gas and a solid interface. This interface exchange
rate was equated to the Newtonian stress existing at the interface of a fluid, and an expression for the vapour slip velocity at the interface was found.

The liquid-vapour interface slip velocity is identical to the solid-liquid interface slip velocity. From equation (A.4), the rate of momentum exchange at a surface is a function of the incident tangential velocity of the impacting molecules, \( v_i \), the tangential wall velocity, \( v_w \), the mass flux of molecules impacting the interface, \( J_i \), and the Specular Reflection Coefficient for the vapour and adjoining medium combination, \( \sigma_v \). None of these variables are affected by either molecular condensation into the liquid, or molecular evaporation from the liquid. Thus, the slip velocity at a liquid-vapour interface, like that at a solid-vapour interface, is given by equation (A.10).

A.3.3 Interface Temperature Discontinuity

While similar, the temperature discontinuity analysis at a liquid-vapour interface is different to the analysis at a solid-vapour interface, because the flux of molecules impacting the adjoining medium is not necessarily equal to the flux of molecules leaving the medium.

Figure A.4 shows a control volume surrounding a liquid-vapour interface. The right side of the control volume lies on the interface, while the left side is anchored at a location within the liquid. Performing an energy balance on this control volume, we have,

\[
q_{in} - q_{out} = \frac{dH}{dt},
\]

where \( q \) represents energy flow rates \([\text{W/m}^2]\), and \( H \) is the total enthalpy of the system. Note that enthalpy is again the appropriate quantity for equation (A.32), as work is required to expand or contract the boundaries of the control volume.
The energy entering the control volume is determined by the enthalpy of the molecules which condense at the liquid-vapour interface. As discussed in Section A.2.2, the temperature of these impacting molecules is \( T_i \), and thus

\[
q_{in} = J_c H_g(T_i).
\]  

(A.33)

Again, all enthalpies are evaluated at the interface pressure. The energy flux leaving the control volume is composed of energy carried away by evaporating molecules, and energy loss into the body of the liquid, \( q_l \). Thus,

\[
q_{out} = J_c H_g(T_w) + q_l.
\]  

(A.34)

The total enthalpy of the liquid in the control volume [J/m\(^2\)] is

\[
H = l \rho_l H_l(T_w),
\]  

(A.35)

where \( l \) is the width of the control volume, assumed to be small compared with the total width of the liquid, \( \rho_l \) is the density of the liquid, and \( H_l \) is
the enthalpy of the liquid, evaluated at the liquid vapour interface pressure. Differentiating equation (A.35) with respect to time gives the total rate of change of enthalpy of the system as

\[ \frac{dH}{dt} = (J_c - J_e) H_l(T_w). \] (A.36)

Combining equations (A.33), (A.34) and (A.36) and rearranging gives the rate of energy loss into the bulk of the liquid as

\[ q_l = J_c H_g(T_i) - J_e H_g(T_w) - (J_c - J_e) H_l(T_w). \] (A.37)

Using the ideal gas enthalpy relation of equation (A.18), the average temperature of molecules impacting the interface can be calculated from equation (A.11) and defining the latent heat of vaporisation at the interface as

\[ H_{fg}(T_w) = H_g(T_w) - H_l(T_w), \] (A.38)

equation (A.37) becomes

\[ q_l = (J_c - J_e) H_{fg}(T_w) + J_c c_p \left( T_o + U' \lambda \frac{dT}{dx} - T_w \right). \] (A.39)

A conventional energy balance on the control volume of Figure A.4 is now performed, assuming the gas is a continuum up to the liquid interface. The energy entering the control volume from conduction within the vapour is given by Fourier’s Law as

\[ q_{in} = k \frac{dT}{dx}, \] (A.40)

where the x coordinate is defined normal to the liquid-vapour interface and directed from the interface position into the vapour. Again, k is defined as the thermal conductivity of the gas. The magnitude of energy leaving the control volume is composed of that lost due to a net evaporation rate of
molecules from the liquid, and the rate of heat loss to the bulk of the liquid. Thus,

\[ q_{out} = (J_e - J_c) H_g (T_w) + q_l. \]  (A.41)

The rate of change of total enthalpy of the system is given by equation (A.36).

Substituting equations (A.36), (A.40) and (A.41) into equation (A.32) and rearranging gives the heat loss into the bulk of the liquid as

\[ q_l = k dT dx - (J_e - J_c) H_g (T_o) - (J_e - J_c) H_l (T_w), \]  (A.42)

where it is assumed that molecules evaporating from the liquid enter the continuous gas phase at the temperature \( T_o \). Utilising equation (A.38) for the latent heat of vaporisation and equation (A.18) for the enthalpy of an ideal gas gives

\[ q_l = k dT dx + (J_e - J_c) (H_f (T_w) + c_p (T_o - T_w)). \]  (A.43)

To determine the magnitude of the temperature discontinuity existing at the liquid-vapour interface, the actual heat loss into the liquid, given by equation (A.39), is equated to the heat loss into the liquid that would be achieved if the gas were a continuum right up to the liquid interface, as given by equation (A.43). Thus,

\[ (J_e - J_c) H_f (T_w) + J_e c_p \left( T_o + U' \lambda dT dx - T_w \right) = k dT dx + (J_e - J_c) (H_f (T_w) + c_p (T_o - T_w)). \]  (A.44)

Cancelling like terms and rearranging gives the temperature discontinuity at the interface as

\[ T_o - T_w = \left( \frac{k - J_e c_p U' \lambda}{J_e c_p} \right) \frac{dT}{dx}. \]  (A.45)

Utilising equation (A.21) for the vapour thermal conductivity, equation (A.13) for Eucken’s approximation for the characteristic temperature length for
impacting molecules, equation (A.14) for the ratio of specific heats, equations (A.24) and (A.26) for the rate of molecules condensing into the liquid medium, and finally defining the ratio of evaporation and condensation rates as

$$\phi = \frac{J_e}{J_c}, \quad (A.46)$$

the temperature discontinuity at the interface is given by

$$T_o - T_w = \frac{\lambda}{\phi} \left( \frac{9\gamma - 5}{4} \right) \left( \frac{2}{\gamma \sigma_t} - 1 \right) \frac{dT}{dx}. \quad (A.47)$$

Note that when the interface is in dynamic equilibrium, the evaporation and condensation rates are equal, $\phi = 1$, and equation (A.47) reduces to the solid-vapour temperature discontinuity equation (A.23).

### A.4 Vapour Mixture Mean Free Path

The kinetic theory treatment of vapour region boundaries presented above is based on the rigid elastic sphere molecular model. This assumption was implied in the values quoted for the Chapman-Enskog characteristic lengths, $U\lambda$ and $U'\lambda$. Under such a model, the mean free path for the gas is defined as the average distance a molecule travels between collisions with other molecules. For a gas at temperature $T$ and pressure $P$, this mean free path is given as [Patterson 1956, Rohsenow & Choi 1961],

$$\lambda = \frac{\kappa T}{\sqrt{2\pi P\sigma^2}}, \quad (A.48)$$

where $\kappa$ is the Boltzmann constant [J/K]. The variable $\sigma$ in this equation is defined as the collision diameter of the molecule [m].

In this study, an average mean free path for the vapour mixture has been used to develop the vapour interface molecular treatments. The average
vapour mixture mean free path is defined as

$$\lambda_m = \frac{\kappa T}{\sqrt{2\pi} P \sigma_m^2}, \quad (A.49)$$

where the average collision diameter for the mixture is defined on a volume molecular average basis as

$$\sigma_m = \left[ y_a \left( 0.21 \sigma_{O_2}^3 + 0.79 \sigma_{N_2}^3 \right) + (1 - y_a) \sigma_v^3 \right]^{\frac{1}{3}}. \quad (A.50)$$

In this equation, $\sigma_{O_2}$ is the collision diameter of an oxygen molecule, $\sigma_{N_2}$ is the collision diameter of a nitrogen molecule, and $\sigma_v$ is the collision diameter of a droplet vapour molecule.

As the collision diameters of molecules relevant to this work are rarely published, for the droplet impact simulations, collision diameters were calculated from well-defined vapour viscosity values. The rigid sphere molecular model calculated using the Chapman-Enskog theory predicts the absolute viscosity of a gas to be [Chapman & Cowling 1970],

$$\mu = 0.1792 \sqrt{\frac{\kappa m T}{\sigma^2}}, \quad (A.51)$$

where $m$ is the mass of a single molecule. Noting that the molecular mass of a gas is given by [kg/kmol]

$$M = 1000 m N_A, \quad (A.52)$$

equation (A.51) gives the collision diameter of a gas as

$$\sigma = 0.0753 \left( \frac{\kappa M T}{\mu^2 N_A} \right)^{\frac{1}{4}}. \quad (A.53)$$

In calculating collision diameters for droplet impact simulations, absolute viscosity values at atmospheric pressure and at the average temperature of the vapour layer were used. This gave the most accurate representation for the magnitude of each collision diameter.
Appendix B

The Thermal Energy Transport Equation Applied to a VOF Simulation

The thermal energy transport equation for a $n$-component system, in terms of the internal energy, is given by Bird et al. [1960] as

$$\rho \frac{D\hat{U}}{Dt} = - \left( \nabla \cdot q \right) - \left( \pi : \nabla v \right) + \sum_{i=1}^{n} \left( j_i \cdot g_i \right). \quad (B.1)$$

In this equation $\rho$ is the density of the multi-component fluid, $\hat{U}$ is the internal energy, $q$ is the multi-component energy flux relative to the mass average velocity, $v$ is the velocity of the fluid, $j_i$ is the mass flux of component $i$ relative to the mass average velocity, $\pi$ is the stress tensor and $g_i$ is the total body force per unit mass of component $i$.

Assuming incompressible and inviscid flow, the stress tensor term of equation (B.1) becomes zero. Also, for the examined droplet impact problem, the body force term in equation (B.1) is negligible. Thus, the thermal energy
The density of the two component droplet fluid is given by equation (4.5) as

$$\rho = F \rho_f,$$  \hspace{1cm} (B.3)

where $\rho_f$ is the density of the droplet liquid, and the density of the surrounding medium is assumed to be negligible. Also, assuming a constant specific heat for the liquid, $c$, we have

$$\hat{U} = cT,$$  \hspace{1cm} (B.4)

where $T$ is the temperature of the fluid, and again the effect of the surrounding medium is neglected. In the droplet calculation, energy fluxes are assumed to occur only within the droplet liquid. Thus, the multi-component energy flux is given by

$$q = -kF \nabla T,$$  \hspace{1cm} (B.5)

where $k$ is the constant thermal conductivity of the liquid and $F$ is the volume of fluid fraction.

Substituting equations (B.3), (B.4) and (B.5) into equation (B.2), and evaluating the vector operators in two dimensional cylindrical coordinates gives the heat diffusion equation for a VOF calculation,

$$F \frac{\partial T}{\partial t} + uF \frac{\partial T}{\partial r} + vF \frac{\partial T}{\partial z} = \alpha \left[ \frac{1}{r} \frac{\partial}{\partial r} \left( rF \frac{\partial T}{\partial r} \right) + \frac{\partial}{\partial z} \left( F \frac{\partial T}{\partial z} \right) \right].$$  \hspace{1cm} (B.6)
Appendix C

Material and Computational Properties used in the Droplet Impact Simulations

In this appendix the material and computational properties used in the droplet impact simulations are specified.

C.1 Liquid Properties

C.1.1 Water

Tables C.1 and C.2 display the liquid properties used for water at the initial droplet temperatures of 373 K and 298 K, respectively.

Table C.3 displays the saturation constants used to define the saturation curve for water. The constants $D_{\text{sat}}$ and $E_{\text{sat}}$ define upper and lower pressure limits between which the saturation curve is applicable. Note that the constants shown define a saturation curve with [K] as the temperature units and
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_f$</td>
<td>957.9 kg/m$^3$</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_l$</td>
<td>$680 \times 10^{-3}$ W/mK</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal diffusivity, $\alpha_l$</td>
<td>$1.68 \times 10^{-7}$ m$^2$/s</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>surface tension, $\sigma$</td>
<td>0.0589 N/m</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>latent heat, $H_{fg}$</td>
<td>$2.257 \times 10^6$ J/kg</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_f$</td>
<td>$279 \times 10^{-6}$ Ns/m$^2$</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

**Table C.1:** Liquid water properties at 373 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_f$</td>
<td>997 kg/m$^3$</td>
<td>at 298 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_l$</td>
<td>$613 \times 10^{-3}$ W/mK</td>
<td>at 298 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal diffusivity, $\alpha_l$</td>
<td>$1.471 \times 10^{-7}$ m$^2$/s</td>
<td>at 298 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>surface tension, $\sigma$</td>
<td>0.0717 N/m</td>
<td>at 298 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>latent heat, $H_{fg}$</td>
<td>$2.257 \times 10^6$ J/kg</td>
<td>at 373 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_f$</td>
<td>$855 \times 10^{-6}$ Ns/m$^2$</td>
<td>at 298 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

**Table C.2:** Liquid water properties at 298 K.
<table>
<thead>
<tr>
<th>Saturation Constant</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{\text{sat}}$</td>
<td>$0.426776 \times 10^2$</td>
<td></td>
<td>Irvine Jr &amp; Liley [1984]</td>
</tr>
<tr>
<td>$B_{\text{sat}}$</td>
<td>$-0.389270 \times 10^4$</td>
<td></td>
<td>Irvine Jr &amp; Liley [1984]</td>
</tr>
<tr>
<td>$C_{\text{sat}}$</td>
<td>$-0.948654 \times 10^1$</td>
<td></td>
<td>Irvine Jr &amp; Liley [1984]</td>
</tr>
<tr>
<td>$D_{\text{sat}}$</td>
<td>$0.000611 \text{ MPa}$</td>
<td></td>
<td>Irvine Jr &amp; Liley [1984]</td>
</tr>
<tr>
<td>$E_{\text{sat}}$</td>
<td>$12.33 \text{ MPa}$</td>
<td></td>
<td>Irvine Jr &amp; Liley [1984]</td>
</tr>
<tr>
<td>Boiling point, $T_{\text{boil}}$</td>
<td>$373.15 \text{ K}$</td>
<td></td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

Table C.3: Saturation constants for water.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_f$</td>
<td>$679.4 \text{ kg/m}^3$</td>
<td>at 298 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>thermal conductivity, $k_l$</td>
<td>$127.5 \times 10^{-3} \text{ W/mK}$</td>
<td>at 298 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>thermal diffusivity, $\alpha_l$</td>
<td>$8.364 \times 10^{-8} \text{ m}^2/\text{s}$</td>
<td>at 298 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>surface tension, $\sigma$</td>
<td>$0.0202 \text{ N/m}$</td>
<td>at 298 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>latent heat, $H_{fg}$</td>
<td>$0.3167 \times 10^6 \text{ J/kg}$</td>
<td>at 298 K</td>
<td>Vargaftik [1975]</td>
</tr>
</tbody>
</table>

Table C.4: Liquid n-Heptane properties at 298 K.

[MPa] as the pressure units. The constant $T_{\text{boil}}$ is used to define the ambient pressure, $P_{\text{atm}}$.

C.1.2 n-Heptane

Table C.4 displays the liquid properties used for n-Heptane at the initial droplet temperature of 25°C.

Table C.5 displays the saturation constants used to define the saturation curve for n-Heptane. The constants $D_{\text{sat}}$ and $E_{\text{sat}}$ the define upper and lower pressure limits between which the saturation curve is applicable. Within this range, the maximum error in the predicted saturation temperature is always
Table C.5: Saturation constants for n-Heptane.

<table>
<thead>
<tr>
<th>Variable (Saturation Constant)</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{sat}$</td>
<td>$0.56617 \times 10^2$</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
<tr>
<td>$B_{sat}$</td>
<td>$-0.29133 \times 10^4$</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
<tr>
<td>$C_{sat}$</td>
<td>$-0.696042 \times 10^1$</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
<tr>
<td>$D_{sat}$</td>
<td>$0.00001$ MPa</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
<tr>
<td>$E_{sat}$</td>
<td>$3.0$ MPa</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
<tr>
<td>Boiling point, $T_{boil}$</td>
<td>$371.58$ K</td>
<td></td>
<td>Ohe [1976]</td>
</tr>
</tbody>
</table>

Table C.6: Steam properties at 520 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_v$</td>
<td>$0.4156$ kg/m$^3$</td>
<td>at 520 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>thermal conductivity, $k_v$</td>
<td>$38.28 \times 10^{-3}$ W/mK</td>
<td>at 520 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_v$</td>
<td>$18.22 \times 10^{-6}$ Ns/m$^2$</td>
<td>at 520 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>corrected latent heat, $H_{fg,c}$</td>
<td>$2.556 \times 10^6$ J/kg</td>
<td></td>
<td>Cullen [1981]</td>
</tr>
</tbody>
</table>

less than 0.12 K. As previously, the constants shown define a saturation curve with [K] as the temperature units and [MPa] as the pressure units.

C.2 Vapour Properties

C.2.1 Wachters and Westerling Impacts — Water

Table C.6 shows the properties required for steam evaluated at 520 K, which is the average vapour layer temperature for the Wachters & Westerling [1966] impacts detailed in Section 10.1.
Table C.7: Steam properties at 550 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_v$</td>
<td>0.393 kg/m$^3$</td>
<td>at 550 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>thermal conductivity, $k_v$</td>
<td>$41.37 \times 10^{-3}$ W/mK</td>
<td>at 550 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_v$</td>
<td>$19.46 \times 10^{-6}$ Ns/m$^2$</td>
<td>at 550 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>corrected latent heat, $H_{fg,c}$</td>
<td>$2.616 \times 10^6$ J/kg</td>
<td></td>
<td>Cullen [1981]</td>
</tr>
</tbody>
</table>

Table C.8: Gaseous n-Heptane properties at 447 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_v$</td>
<td>2.73 kg/m$^3$</td>
<td>at 447 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>thermal conductivity, $k_v$</td>
<td>$27.2 \times 10^{-3}$ W/mK</td>
<td>at 447 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_v$</td>
<td>$88.5 \times 10^{-7}$ Ns/m$^2$</td>
<td>at 447 K</td>
<td>Vargaftik [1975]</td>
</tr>
<tr>
<td>corrected latent heat, $H_{fg,c}$</td>
<td>$0.441 \times 10^6$ J/kg</td>
<td></td>
<td>Vargaftik [1975]</td>
</tr>
</tbody>
</table>

C.2.2 Groendes and Mesler Impacts — Water

Table C.7 shows the properties required for steam evaluated at 550 K, which is the average vapour layer temperature for the Groendes & Mesler [1982] impacts detailed in Section 10.2.

C.2.3 Qiao and Chandra and Chandra and Avedisian Impacts — n-Heptane and Air

Tables C.8 and C.9 show the vapour properties required for the Qiao & Chandra [1996] and Chandra & Avedisian [1991] impacts detailed in Section 11.1. For brevity, only data corresponding to an average vapour layer temperature of 447 K are shown.
Table C.9: Properties for air at 447 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_a$</td>
<td>0.774 kg/m$^3$</td>
<td>at 447 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_a$</td>
<td>37.3×10$^{-3}$ W/mK</td>
<td>at 447 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_a$</td>
<td>250.7×10$^{-7}$ Ns/m$^2$</td>
<td>at 447 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

Table C.10: Steam properties at 473 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_v$</td>
<td>0.4604 kg/m$^3$</td>
<td>at 473 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>thermal conductivity, $k_v$</td>
<td>33.37×10$^{-3}$ W/mK</td>
<td>at 473 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_v$</td>
<td>16.18×10$^{-6}$ Ns/m$^2$</td>
<td>at 473 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>corrected latent heat, $H_{fg,c}$</td>
<td>2.457×10$^6$ J/kg</td>
<td></td>
<td>Cullen [1981]</td>
</tr>
</tbody>
</table>

C.2.4 Qiao and Chandra Impacts — Water and Air


C.2.5 Chen and Hsu Impact — Water and Air

Tables C.12 and C.13 show the vapour properties required for the Chen & Hsu [1995] water impact detailed in Section 11.2.2.

Table C.11: Properties for air at 473 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_a$</td>
<td>0.7383 kg/m$^3$</td>
<td>at 473 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_a$</td>
<td>38.86×10$^{-3}$ W/mK</td>
<td>at 473 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_a$</td>
<td>259.6×10$^{-7}$ Ns/m$^2$</td>
<td>at 473 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>Variable</td>
<td>Value</td>
<td>Notes</td>
<td>Reference</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
<td>-------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>density, $\rho_v$</td>
<td>0.3790 kg/m$^3$</td>
<td>at 573 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>thermal conductivity, $k_v$</td>
<td>$43.49 \times 10^{-3}$ W/mK</td>
<td>at 573 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_v$</td>
<td>$20.29 \times 10^{-6}$ Ns/m$^2$</td>
<td>at 573 K</td>
<td>Cullen [1981]</td>
</tr>
<tr>
<td>corrected latent heat, $H_{fg,c}$</td>
<td>$2.656 \times 10^6$ J/kg</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table C.12: Steam properties at 573 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_a$</td>
<td>0.6088 kg/m$^3$</td>
<td>at 573 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_a$</td>
<td>$45.3 \times 10^{-3}$ W/mK</td>
<td>at 573 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>absolute viscosity, $\mu_a$</td>
<td>$296.4 \times 10^{-7}$ Ns/m$^2$</td>
<td>at 573 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

Table C.13: Properties for air at 573 K.

C.3 Solid Properties

C.3.1 Wachters and Westerling Impacts — Gold

Table C.14 shows the properties used for gold at 673 K, as required for the Wachters & Westerling [1966] impacts detailed in Section 10.1.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_s$</td>
<td>19300 kg/m$^3$</td>
<td></td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_s$</td>
<td>303.2 W/mK</td>
<td>at 673 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>specific heat, $c_s$</td>
<td>133.4 J/kg.K</td>
<td>at 673 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal diffusivity, $a_s$</td>
<td>$1.178 \times 10^{-4}$ m$^2$/s</td>
<td>at 673 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

Table C.14: Gold properties at 673 K.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_s$</td>
<td>2650 kg/m$^3$</td>
<td></td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_s$</td>
<td>3.20 W/mK</td>
<td>at 735 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>specific heat, $c_s$</td>
<td>1193 J/kg.K</td>
<td>at 735 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal diffusivity, $a_s$</td>
<td>$1.012 \times 10^{-6}$ m$^2$/s</td>
<td>at 735 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

Table C.15: Quartz properties at 735 K.

C.3.2 Groendes and Mesler Impacts — Quartz

Table C.15 shows the properties used for quartz (Silicon dioxide, crystalline) at 735 K, as required for the Groendes & Mesler [1982] impacts detailed in Section 10.2. The value for the thermal conductivity was taken as normal to the grain structure of the crystal, this being the most likely orientation of the experimental material.

C.3.3 Qiao and Chandra and Chandra and Avedisian Impacts — Stainless Steel

Table C.16 shows the properties used for stainless steel, as required for the Qiao & Chandra [1996] and Chandra & Avedisian [1991] impacts detailed in Section 11.1. It was assumed that the steel was of type AISI 304. For brevity, the properties are shown at only one of the temperatures used in the simulations, 523 K.

C.3.4 Chen and Hsu Impacts — Inconel 600

Table C.17 shows the properties used for the nickel alloy Inconel 600, as required for the Chen & Hsu [1995] impacts detailed in Section 11.2.2.
### Table C.16: Stainless steel properties at 523 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_s$</td>
<td>7900 kg/m$^3$</td>
<td></td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal conductivity, $k_s$</td>
<td>18.6 W/mK</td>
<td>at 523 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>specific heat, $c_s$</td>
<td>541 J/kg.K</td>
<td>at 523 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
<tr>
<td>thermal diffusivity, $a_s$</td>
<td>$4.35 \times 10^{-6}$ m$^2$/s</td>
<td>at 523 K</td>
<td>Incropera &amp; DeWitt [1996]</td>
</tr>
</tbody>
</table>

### Table C.17: Inconel 600 properties at 773 K.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Notes</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>density, $\rho_s$</td>
<td>8420 kg/m$^3$</td>
<td></td>
<td>Smithells &amp; Brandes [1976]</td>
</tr>
<tr>
<td>thermal conductivity, $k_s$</td>
<td>21.99 W/mK</td>
<td>at 773 K</td>
<td>O’Sullivan, Jr [1955]</td>
</tr>
<tr>
<td>specific heat, $c_s$</td>
<td>540.1 J/kg.K</td>
<td>at 773 K</td>
<td>O’Sullivan, Jr [1955]</td>
</tr>
<tr>
<td>thermal diffusivity, $a_s$</td>
<td>$4.835 \times 10^{-6}$ m$^2$/s</td>
<td>at 773 K</td>
<td>O’Sullivan, Jr [1955]</td>
</tr>
</tbody>
</table>

### C.4 Gas Molecular Properties

Table C.18 shows the molecular properties required for the four gases used in this study. Molecular collision diameters were calculated from gas viscosity measurements, as detailed in Section A.4 of Appendix A. The viscosity measurements used in the collision diameter calculations were evaluated at 500 K. The viscosity measurements for nitrogen, oxygen and steam were taken from Cullen [1981], while the values for n-Heptane were taken from Vargaftik [1975].

There is considerable debate in the literature regarding the magnitude of Thermal Accommodation coefficients for various solid, liquid and vapour combinations. In a study on vapour film destabilisation by Knowles [1985], it was estimated that the most accurate estimation for the Thermal Accommodation coefficient for water is that it lies in the range $0.35 < \sigma_t \leq 1.0$.  

392
Mills & Seban [1967] estimated that the Thermal Accommodation coefficient for water lies in the range $0.45 < \sigma_t \leq 1.0$, but stressed that the range of possible values quoted must not be taken to favour an intermediate value. For these reasons, all Thermal Accommodation coefficients used in this study were given the value of one. Following the observations made in Chapters 10 and 11 however, it is not expected that the Thermal Accommodation coefficients will have a significant influence on the heat transfer rates predicted by BOUNCE anyway.

In general Specular Reflection coefficients have been established with a greater precision than Thermal Accommodation coefficients. Rohsenow & Choi [1961] give the Specular Reflection coefficients for a large variety of vapour and solid combinations, and the values quoted are always close to unity. For this reason, all Specular Reflection coefficients used in this study were given the value of one.

### C.5 Computational Properties

Table C.18 shows the computational specifications used to integrate the PPE equation during the droplet simulations presented in Chapters 10 and 11.

In all droplet impact simulations presented, the Defined Donating Region VOF advection scheme was used, coupled with the error minimisation

<table>
<thead>
<tr>
<th>Gas</th>
<th>nitrogen</th>
<th>oxygen</th>
<th>steam</th>
<th>n-Heptane</th>
</tr>
</thead>
<tbody>
<tr>
<td>molecular mass, $M$ [kg/kmol]</td>
<td>28.01</td>
<td>32.00</td>
<td>18.02</td>
<td>100.20</td>
</tr>
<tr>
<td>viscosity for $\sigma$ calc, $\mu \times 10^7$ [Ns/m²]</td>
<td>257.7</td>
<td>303.3</td>
<td>165.9</td>
<td>98.6</td>
</tr>
<tr>
<td>collision diameter, $\sigma \times 10^{10}$ [m]</td>
<td>3.51</td>
<td>3.34</td>
<td>3.91</td>
<td>7.80</td>
</tr>
</tbody>
</table>

Table C.18: Molecular gas properties.
<table>
<thead>
<tr>
<th>Computational Property</th>
<th>Value</th>
<th>Refer to Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial timestep, $\delta t_i$</td>
<td>$1 \mu s$</td>
<td></td>
</tr>
<tr>
<td>PPE convergence comparator, $\varepsilon$</td>
<td>$1 \times 10^{-5}$</td>
<td>4.4</td>
</tr>
<tr>
<td>Iterative pressure solution comparator, $\varepsilon_P$</td>
<td>$1 \times 10^{-2}$</td>
<td>4.3.3</td>
</tr>
<tr>
<td>Momentum advection type coefficient, $\alpha$</td>
<td>0.70</td>
<td>4.4.3</td>
</tr>
</tbody>
</table>

Table C.19: Computational properties used in integrating the PPE equation.

<table>
<thead>
<tr>
<th>Computational Property</th>
<th>Value</th>
<th>Refer to Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes in solid mesh, $k_{\text{max}}$</td>
<td>32</td>
<td>9.3.1</td>
</tr>
<tr>
<td>Stretch rate for solid mesh, $\chi_s$</td>
<td>1.2</td>
<td>9.3.1</td>
</tr>
<tr>
<td>Minimum solid mesh size, $\delta z_{s,\text{min}}$</td>
<td>1 $\mu m$</td>
<td>9.3.1</td>
</tr>
<tr>
<td>Number of nodes in liquid mesh, $l_{\text{max}}$</td>
<td>37</td>
<td>9.3.2</td>
</tr>
<tr>
<td>Stretch rate for liquid mesh, $\chi_l$</td>
<td>1.2</td>
<td>9.3.2</td>
</tr>
<tr>
<td>Minimum liquid mesh size, $\delta z_{l,\text{min}}$</td>
<td>0.5 $\mu m$</td>
<td>9.3.2</td>
</tr>
</tbody>
</table>

Table C.20: The solid and liquid phase temperature grid specifications used in the droplet impact simulations.

method for determining free surface orientations.

In calculating the surface tension volume forces, the VOF functions were smoothed prior to free surface curvature evaluation, and the size of the kernel over which this smoothing was performed varied within the range $1 \leq s_k \leq 1.5$ (see Section 4.5.1). As detailed in Section 4.5.1, free surface curvatures were altered to reflect the actual position of the fluid void interface, and the curvature correction coefficient used in this algorithm was $C_{\text{correct}} = 0.5$.

Table C.20 shows the liquid and solid phase temperature grid specifications used in the droplet simulations presented in Chapters 10 and 11.
Appendix D

Viscous Vapour Layer Code

Integrity Calculations

In this Appendix we present several simple test calculations performed to demonstrate the integrity of the BOUNCE vapour layer code.

D.1 Pressure Application Test

This simple test is used to demonstrate the integrity of the mechanism used to apply the generated vapour layer pressures to the main VOF droplet dynamics code.

A constant pressure is applied to the lower surface of a cylinder of fluid, and as a result, the cylinder accelerates upward. The cylinder of fluid is bounded at the periphery by a frictionless computational boundary. The analytical solution to this problem is given by

\[
z - z_0 = \frac{P}{h\rho_f} \frac{t^2}{2},
\]

where \( P \) is the applied gauge pressure at the lower surface of the cylinder, \( h \) is the height of the cylinder, \( z \) is the height of the vapour layer, \( z_0 \) is the
Figure D.1: Pressure application test case showing the height of the vapour layer when a cylinder of fluid is accelerated by a constant vapour layer pressure. Both analytical and BOUNCE solutions are shown.

height of the vapour layer when the pressure is first applied, and $\rho_f$ is the fluid density.

Figure D.1 shows example BOUNCE and analytical solutions to this problem. The correlation between the results is good, indicating that the vapour layer pressure is correctly applied to the VOF dynamics algorithm. The average timestep used by BOUNCE in this test was 0.015 s.

D.2 Pressure Generation Test

The purpose of this test is to demonstrate that BOUNCE is correctly calculating the viscous vapour layer pressures and volume flowrates.
Assuming that the height of the vapour layer is constant, and the vapour
generation velocity at the underside of the droplet is uniform and constant,
then the vapour layer mass conservation equation \( (8.23) \) can be integrated
to give

\[
S(r) = \pi wr^2. \tag{D.2}
\]

In this equation, \( S \) is the vapour flow rate within the layer and \( w \) is the
uniform vaporisation velocity. Dividing equation \( (D.2) \) by the maximum
radius of the vapour layer, \( R \), and rearranging yields the non-dimensional
equation

\[
\frac{S}{\pi wR^2} = \left( \frac{r}{R} \right)^2. \tag{D.3}
\]

Both the BOUNCE and analytical solutions to equation \( (D.3) \) are shown
in Figure \( D.2 \). The correlation between the two solutions is good, indicating
that the volume flowrates within the vapour layer are correctly calculated by
BOUNCE. The BOUNCE solution shown in Figure \( D.2 \) was generated using
50 vapour layer computational subcells.

The pressures generated within the vapour layer are given by equation
\((8.15)\). Assuming that there is no velocity slip at the vapour layer interfaces,
and assuming that the velocity of the droplet liquid at the underside of
the droplet is zero, this equation can be combined with equation \( (D.2) \) and
integrated to give,

\[
P = \frac{3\mu_m w \delta}{3} \left( R^2 - r^2 \right). \tag{D.4}
\]

Here \( \delta \) is the uniform and constant height of the vapour layer, and \( \mu_m \) is the
absolute viscosity of the vapour within the vapour layer. Equation \( (D.4) \) can
be written in a non-dimensional form as,

\[
\frac{P \delta}{3\mu_m w R^2} = 1 - \left( \frac{r}{R} \right)^2. \tag{D.5}
\]
Analytical and BOUNCE solutions to equation (D.5) are shown in Figure D.2. The correlation between the two solutions is good, indicating that BOUNCE is correctly calculating vapour layer pressures. The BOUNCE pressure solution shown in Figure D.2 was generated using 50 vapour layer computational subcells.

### D.3 Air Volume Fraction Test

The purpose of this test is to ensure that the vapour layer air volume fraction is correctly calculated in the BOUNCE code.

Equation (8.36) is a transport equation, describing the air volume fraction distribution within the vapour layer. Assuming that the height of the vapour layer is constant and uniform, that the volume flowrate within the vapour
layer is given by equation (D.2), then equation (8.36) can be manipulated to give
\[ 2\delta \frac{\partial y_a}{\partial t} + r \frac{\partial y_a}{\partial r} + 2y_a = 0. \] (D.6)
Here \( y_a \) is the air volume fraction within the vapour layer, and the variables \( \delta \) and \( w \) are defined as previously.

Using the initial condition
\[ y_a(r) = y_{a,i}, \] (D.7)
equation (D.6) can be integrated to give the air volume fraction within the vapour layer as
\[ y_a = y_{a,i} \exp \left( -\frac{wt}{\delta} \right). \] (D.8)
Interestingly, this fraction is independent of radial location.

The analytical and BOUNCE solutions of equation (D.8) are shown in Figure D.3 for an initial air volume fraction of \( y_{a,i} = 0 \). This initial fraction was chosen for reasons of computational convenience. The average timestep used over the 3 s of simulation shown was 0.02 s. The correlation between the two solutions is good, indicating that BOUNCE is correctly calculating the convection of the air volume fraction within the vapour layer.

### D.4 Solid Temperature Conduction Test

This test is used to demonstrate the integrity of the solid phase conduction algorithm used in the BOUNCE vapour layer code.

Equation (8.64) describes the heat loss from the solid phase. If the vapour layer interface molecular treatment is discarded, and the temperature of the liquid surface and height of the vapour layer are held at constant values, then the heat loss from the solid phase can be modelled as the heat loss
from a semi-infinite solid with an effective heat transfer coefficient of \( \frac{k_m}{\delta} \). The surface temperature of such a solid can be calculated analytically using [Incropera & DeWitt 1996]

\[
\frac{T_{ss} - T_{s,i}}{T_{li} - T_{s,i}} = 1 - \exp \left( \left( \frac{k_m}{k_s \delta} \right)^2 \alpha_s t \right) \left[ 1 - \text{erf} \left( \frac{k_m \alpha_s t}{k_s \delta} \right) \right],
\]

(D.9)

where \( T_{s,i} \) is the initial temperature of the solid, \( k_m \) and \( k_s \) are the thermal conductivities of the vapour and solid phases, respectively, and \( \alpha_s \) is the thermal diffusivity of the solid phase.

The non-dimensional solid surface temperature calculated by BOUNCE under the described conditions is shown in Figure D.4. For comparison the same temperature, calculated using equation (D.9), is also shown at several times. The temperatures calculated analytically compare well with the
surface temperatures calculated by BOUNCE. Consequently, the BOUNCE algorithm is correctly calculating heat transfer from the solid phase.

**D.5 Liquid Temperature Conduction Test**

A conduction test within the liquid phase, similar in form to the conduction test within the solid phase, is presented to demonstrate the integrity of the BOUNCE liquid phase conduction algorithm.

Equation (8.69) describes heat transfer into the liquid phase. Assuming that

1. the height of the vapour layer is constant,

2. the thermal conductivity within the vapour layer is constant,
3. the interface molecular treatment is neglected,

4. no vapour is generated at the lower surface of the droplet,

5. the temperature of the surface of the solid is held constant, and

6. convection within the liquid is neglected,

then heat transfer within the liquid phase can be modelled as that within a semi-infinite solid, using an effective heat transfer coefficient of \( \frac{k_m}{\delta} \). The surface temperature of such a liquid can be calculated analytically using [Incropera & DeWitt 1996]

\[
\frac{T_{ll} - T_{li}}{T_{ss} - T_{li}} = 1 - \exp \left[ \left( \frac{k_m}{k_l \delta} \right)^2 \alpha_l t \right] \left[ 1 - \text{erf} \left( \frac{k_m \sqrt{\alpha_l t}}{k_l \delta} \right) \right],
\]

(D.10)

where variables with the \( l \) subscript now describe properties of the liquid phase.

Figure D.5 shows non-dimensional liquid surface temperatures calculated by BOUNCE and equation (D.10) for the conditions described above. The correlation between the two results is good, indicating that liquid phase conduction is correctly calculated by the BOUNCE algorithm.

### D.6 Liquid Temperature Convection Test

The purpose of this test is to demonstrate the integrity of the liquid phase convection algorithm employed in the BOUNCE vapour layer code.

Thermal energy transport within the liquid phase is described by equation (8.57). If conduction within the material is neglected, and if the height of the vapour layer is assumed to be uniform and constant, then equation (8.57) reduces to

\[
\frac{\partial T_l}{\partial t} = -u \frac{\partial T_l}{\partial r} - v \frac{\partial T_l}{\partial z},
\]

(D.11)
where the variables are as defined in Section 8.4.3.

Assuming a velocity field within the liquid phase described by

\[ v = -z \quad \text{and} \quad u = \frac{r}{2}, \quad (D.12) \]

a solution to equation (D.11) is given by

\[ T_l = ze^t, \quad (D.13) \]

where the initial condition

\[ T_l = z \quad (D.14) \]

has been assumed. Note that equation (D.13) specifies the temperature at the front surface of the liquid phase as zero at all times.

Using the velocity field described by equation (D.12), and the initial temperature condition given by equation (D.14), the temperature within the
depth of the liquid phase was calculated by BOUNCE. The results are shown in Figure D.6 together with the analytical solution specified by equation (D.13).

The correlation between the two results is good, indicating that BOUNCE is correctly calculating convective heat fluxes within the body of the liquid.
Appendix E

Listing of the BOUNCE Source Code

Upon request, the author can provide a personal computer disk containing a listing of the BOUNCE FORTRAN source code, sample input files and sample output files. Enquires should be directed to:

Dalton Harvie,
c/- Department of Mechanical and Mechatronic Engineering,
University of Sydney, NSW, 2006,
Australia.