INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
Regularization of Periodic Vortex Sheets

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy in the
Graduate School of The Ohio State University

By

Lan D. Pham. M.S.

* * * * *

The Ohio State University

2001

Dissertation Committee:
Professor Gregory R. Baker. Adviser
Professor Edward Overman
Professor Michael R. Foster

Approved by

[Signature]
Adviser
Department of Mathematics
ABSTRACT

A vortex sheet is an asymptotic model of a parallel shear flow where the width of the vorticity layer is much smaller than the layer's characteristic length. Unfortunately, the equation governing the time-evolution of vortex sheets in an inviscid incompressible flow is ill-posed in that smallest scales grow the fastest. Even starting with an analytic initial profile, the sheet forms a curvature singularity in finite time. Theoretically, the sheet has a "weak" solution after the singularity time, but the precise nature of the solution is unknown. Since numerical methods fail to produce meaningful solutions, researchers turned to regularized vortex sheet motion. One common method has been to smooth the kernel in the Birkhoff-Rott integral that determines the vortex sheet velocity. In this thesis, I study two classes of smoothed kernels numerically. Different algorithmic approaches are examined with the goal to reduce calculational costs. Numerical solutions are then obtained to suggest possible form of the limiting solution as the smoothing parameter tends to zero. Similarity studies is conducted to understand the connection between the smoothing parameter and time.
This work is dedicated to my loving parents

iii
ACKNOWLEDGMENTS

I am eternally grateful to my advisor Professor Gregory Baker, without whose support, encouragement, and guidance, this work would not be possible. Prof. Baker has provided generous financial support and the use of the Beowulf cluster.

I would like to acknowledge Professor Stephen Cowley, whom I had a pleasure to meet in Summer of 2000, for contributing some ideas to this project.

Much appreciation goes to all the professors I have had the pleasure to learn from over the years. In particular, I am very grateful to have learned much from George Majda, Edward Overman, Saleh Tanveer, Michael Foster, Richard Bodonyi, Thorwald Herbert, and Richard Parent.

I express my gratitude to Dave Alden, Chad Cunningham, and the computer staff for providing valuable computer support and answering numerous questions over the years.

I thank my parents and family for their continual support of my pursuit of higher education. I like to express my warmest gratitude to my fiance Tien for her patience and love.
VITA

November. 2 1969 ......................... Born - Saigon, Vietnam

1993 ................................................. B.S. University of California, Berkeley

1995 ................................................. M.S. University of California, Davis

1995-1998 ........................................ Graduate Teaching Associate.
                                             Ohio State University.

                                             Ohio State University.

2001-present ..................................... Temporary Lecturer, Ohio State Uni-
                                             versity

FIELDS OF STUDY

Major Field: Mathematics

Studies in:

  Numerical Analysis                      Prof. Gregory R. Baker
  Numerical Partial Differential Equations  Prof. George Majda
  Asymptotics                              Prof. Michael Foster
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iii</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>iv</td>
</tr>
<tr>
<td>Vita</td>
<td>v</td>
</tr>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>Chapters:</td>
<td></td>
</tr>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Thesis Outline</td>
<td>4</td>
</tr>
<tr>
<td>1.3 The Birkhoff-Rott Equation</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Notations</td>
<td>7</td>
</tr>
<tr>
<td>2. FORMULATION OF THE PROBLEM</td>
<td>8</td>
</tr>
<tr>
<td>2.1 The Exact Vortex Sheet Kernels</td>
<td>8</td>
</tr>
<tr>
<td>2.2 The Desingularized Equation</td>
<td>10</td>
</tr>
<tr>
<td>2.2.1 The Blob Method</td>
<td>10</td>
</tr>
<tr>
<td>2.2.2 Krasny’s and Baker’s Kernels</td>
<td>11</td>
</tr>
<tr>
<td>2.2.3 Another Regularization Method</td>
<td>12</td>
</tr>
<tr>
<td>2.2.4 Two classes of smoothed kernels</td>
<td>13</td>
</tr>
<tr>
<td>2.3 Numerical Implementation</td>
<td>15</td>
</tr>
<tr>
<td>2.3.1 Implementing the ODE solvers</td>
<td>17</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>2.3.2 Calculating the Instantaneous Velocity</td>
<td>19</td>
</tr>
<tr>
<td>2.3.3 Debugging the code</td>
<td>21</td>
</tr>
<tr>
<td>2.4 Performance Consideration</td>
<td>27</td>
</tr>
<tr>
<td>2.4.1 The Fast Algorithm in Full Range</td>
<td>30</td>
</tr>
<tr>
<td>2.4.2 The Zig-Zag Method</td>
<td>32</td>
</tr>
<tr>
<td>2.4.3 The Symmetric Method</td>
<td>35</td>
</tr>
<tr>
<td>2.4.4 The Lump Method</td>
<td>38</td>
</tr>
<tr>
<td>2.4.5 Parallelization</td>
<td>42</td>
</tr>
<tr>
<td>2.5 The Bracket Technique</td>
<td>47</td>
</tr>
<tr>
<td>2.6 Other Improvements</td>
<td>52</td>
</tr>
<tr>
<td>2.6.1 The Spectrum of the vortex sheet position</td>
<td>53</td>
</tr>
<tr>
<td>2.6.2 The Subtracted Kernel</td>
<td>65</td>
</tr>
<tr>
<td>2.6.3 Beale's Correction</td>
<td>72</td>
</tr>
<tr>
<td>2.6.4 The Mapping Technique</td>
<td>83</td>
</tr>
<tr>
<td>2.6.5 Conclusion</td>
<td>88</td>
</tr>
<tr>
<td>2.7 Some Results</td>
<td>91</td>
</tr>
<tr>
<td>2.8 Summary of Chapter 2</td>
<td>96</td>
</tr>
<tr>
<td>2.9 Preludes to Chapter 3 and 4</td>
<td>96</td>
</tr>
<tr>
<td>3. CONVERGENCE TO WEAK LIMIT</td>
<td>98</td>
</tr>
<tr>
<td>3.1 Summary</td>
<td>98</td>
</tr>
<tr>
<td>3.2 Convergence and Theoretical Issues</td>
<td>99</td>
</tr>
<tr>
<td>3.2.1 The Outer Arms</td>
<td>100</td>
</tr>
<tr>
<td>3.3 Results</td>
<td>102</td>
</tr>
<tr>
<td>3.4 Form-Fit of The Outer Arms</td>
<td>111</td>
</tr>
<tr>
<td>3.5 Future Work</td>
<td>112</td>
</tr>
<tr>
<td>4. SIMILARITY STUDIES</td>
<td>116</td>
</tr>
<tr>
<td>4.1 Motivation For Similarity Studies</td>
<td>116</td>
</tr>
<tr>
<td>4.2 Similarity Studies with Baker's Kernel</td>
<td>121</td>
</tr>
<tr>
<td>4.3 Confirming The Similarity Relation</td>
<td>126</td>
</tr>
<tr>
<td>Appendices:</td>
<td></td>
</tr>
<tr>
<td>A. The Method Of Images</td>
<td>131</td>
</tr>
<tr>
<td>A.1 Beale's Second Kernel</td>
<td>134</td>
</tr>
<tr>
<td>B. COMPLETE DEBUG TABLES</td>
<td>136</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Time comparison between the four kernels. Normalization is with respect to Krasny’s time.</td>
<td>15</td>
</tr>
<tr>
<td>2.2 Testing accuracy of the velocity calculation for Krasny’s Kernel at $\xi = \pi/2$.</td>
<td>23</td>
</tr>
<tr>
<td>2.3 Testing accuracy of the velocity calculation for Baker’s Kernel at $\xi = \pi/2$.</td>
<td>23</td>
</tr>
<tr>
<td>2.4 Testing accuracy of the velocity calculation for Beale’s First Kernel at $\xi = \pi/2$.</td>
<td>26</td>
</tr>
<tr>
<td>2.5 Testing accuracy of the velocity calculation for Beale’s Second Kernel at $\xi = \pi/2$.</td>
<td>26</td>
</tr>
<tr>
<td>2.6 Timing studies for the fast summation method compared to the previous (naive) sum calculation. The variable $T_1$ is the time needed to perform one integration step in the naive method introduced earlier, and $T_2$ is the time for the the fast method.</td>
<td>31</td>
</tr>
<tr>
<td>2.7 Timing studies of the zig-zag method compared to the fast summation method for Krasny’s kernel over a range of $N$.</td>
<td>34</td>
</tr>
<tr>
<td>2.8 Timing studies of the symmetric method compared to the fast summation method for Krasny’s kernel with $\delta = 0.1$ over a range of $N$.</td>
<td>39</td>
</tr>
<tr>
<td>2.9 Timing comparison between the symmetric and the lump method using Krasny’s kernel.</td>
<td>42</td>
</tr>
<tr>
<td>2.10 Bracket table for Krasny’s kernel when $\epsilon = 10^{-5}$. Although the result for 4096 points doesn’t satisfy the criterion, we include the result to show how difficult it is to generate an accurate profile at small $\delta$.</td>
<td>50</td>
</tr>
</tbody>
</table>
2.11 Bracket table for Baker's kernel with $\epsilon = 10^{-5}$. Although the result for 4096 points doesn't satisfy the criterion, we include the result to show how quickly the resolution requirements increase with decreasing $\delta$.  

2.12 Bracket table for Beale's kernel with $\epsilon = 10^{-5}$. 

2.13 Bracket table for Krasny's kernel with $\epsilon = 10^{-8}$. 

2.14 Bracket table for Baker's kernel with $\epsilon = 10^{-8}$. 

2.15 Bracket table for Beale's kernel with $\epsilon = 10^{-8}$. 

2.16 Filtering grid size as a function of time step for a calculation using the adaptive point-insertion technique. 

B.1 Complete debug data for Krasny's kernel. 

B.2 Complete debug data for Baker's Kernel. 

B.3 Complete debug data for Beale's first Kernel. 

B.4 Complete debug data for Beale's second Kernel.
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Initial vortex sheet position over one period.</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>Instantaneous velocity at each point needs contribution from all other points in the discrete parameter space.</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>Naive way of computing the velocity. The point that the arrow points to is the point with parameter ( p ) at which the velocity calculation is being done, and the point at the source of the arrow is the point with parameter ( q ). At each point ( p, q ) ranges over all parameter points except ( p ). Also shown is the iteration step.</td>
<td>20</td>
</tr>
<tr>
<td>2.4</td>
<td>Plots of ( c_X^p ) and ( c_Y^q ) for all four smoothed kernels at ( \delta = 0.10 ).</td>
<td>24</td>
</tr>
<tr>
<td>2.5</td>
<td>Plots of ( c_X^p ) and ( c_Y^q ) for all four smoothed kernels at ( \delta = 0.05 ).</td>
<td>25</td>
</tr>
<tr>
<td>2.6</td>
<td>Krasny's and Baker's Kernels.</td>
<td>28</td>
</tr>
<tr>
<td>2.7</td>
<td>Beale's 1st and 2nd Kernels.</td>
<td>29</td>
</tr>
<tr>
<td>2.8</td>
<td>A better algorithm which takes less number of steps.</td>
<td>32</td>
</tr>
<tr>
<td>2.9</td>
<td>This figure shows the iteration steps corresponding to the appropriate points on the grid. Grid points at the end of each iteration is likely to be in the cache. This is the fundamental idea behind the zig-zag method.</td>
<td>33</td>
</tr>
<tr>
<td>2.10</td>
<td>The zig-zag algorithm showing all the steps for the velocity calculation at each grid point.</td>
<td>34</td>
</tr>
<tr>
<td>2.11</td>
<td>Showing the symmetry in the vortex sheet position.</td>
<td>38</td>
</tr>
</tbody>
</table>
2.12 Symmetry testing for all four smoothed kernels. ........................................... 39
2.13 Diving grid points into groups. ................................................................. 40
2.14 Better algorithm. .......................................................................................... 41
2.15 How the parallel code handles communication to reduce the number of communication steps while making sure that the number of computation steps is large. ................................................................. 45
2.16 This figure shows the plot of the quantity \( \gamma \) defined above. ................. 46
2.17 Calculation of the speedup. ........................................................................... 48
2.18 How points are actually clustered about a region that the function changes rapidly. .............................................................................................................. 53
2.19 Spectrum of the position at \( t = 10 \Delta t \). ..................................................... 56
2.20 Spectrum of the position at \( t = 400 \Delta t \). ..................................................... 57
2.21 Spectrum of the position at \( t = 600 \Delta t \). ..................................................... 58
2.22 Spectrum of the position at \( t = 900 \Delta t \). ..................................................... 59
2.23 Spectrum of the position at \( t = 1010 \Delta t \). ................................................... 60
2.24 Filtering Error. .............................................................................................. 61
2.25 Filtering error when an initial grid size of 64 is applied. ............................ 62
2.26 Filtering error when an initial grid size of 128 is applied. ............................ 63
2.27 As the grid is refined, the vortex sheet position generated by the smoothed \( K_s \) appears to converge. This profile was generated at \( \delta = 0.25 \). Slow convergence may be due to irregular motion. ......................... 68
2.28 This figure shows no irregular motion associated with the subtracted kernel derived by method 2. ............................................................... 72
2.29 The second and third position profiles to be used in testing the accuracy of the subtracted kernels. ........................................... 76

2.30 Debugging the subtraction plus third-order correction method. ........ 78

2.31 Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 1. ...................... 79

2.32 Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 2. ...................... 80

2.33 Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 3. ...................... 81

2.34 Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 4. ...................... 82

2.35 This figure the distribution of points used to parameterize the sheet. ... 85

2.36 How we generate \( \phi \) by hand and use Fourier analysis to construct a smooth approximation. Also shown is the derivative of the function \( \phi \) to make sure that it is not zero. .................................................. 87

2.37 Redistribution of markers which parameterize the vortex sheet. The top figure shows result for the non-mapping technique, i.e., \( \lambda = 0 \). The bottom figure shows the result after the mapping. ................. 89

2.38 Top figure: accuracy of the sheet position obtain from the non-mapping kernel. Bottom figure: accuracy of the sheet position obtain from the mapping kernel. .................................................. 90

2.39 How the vortex sheet changes as \( \delta \) is decreased. ...................... 92

2.40 Generation of more spirals as for smaller and smaller \( \delta \). .............. 93

2.41 Comparing vortex sheets generated by the four kernels. .................. 94

2.42 Comparing vortex sheet generated by the four kernels. ................... 95

3.1 Measuring the arms. .......................................................... 103
3.2 Measuring the first outer arm $s_1(\delta)$. ................................................. 104
3.3 Measuring the second outer arm $s_2(\delta)$ .................................................. 105
3.4 Measuring the third outer arm $s_3(\delta)$. ..................................................... 106
3.5 Beale’s first compared to Beale’s second Kernel. .......................................... 107
3.6 Wiggles generated by the first outer arms from Beale’s second kernel. ......... 108
3.7 Irregular motion on the outer portion of the vortex sheet calculated on a 4096 point grid. Oscillations weaken when the vortex sheet is calculated on a grid of 8196 points. .................................................. 110
3.8 Degree 2 polynomial form fit $a + bx + cx^2$. ............................................. 113
3.9 Degree 3 polynomial form fit $a + bx + cx^2 + dx^3$. .................................. 114
3.10 Condition number of the matrix $A$ for degree 3 polynomial form fit $a + bx + cx^2 + dx^3$ as a function of $\delta$. ..................................................... 115
4.1 Evolution of the vortex sheet for fixed final time as the smoothing parameter $\delta$ is decreased. ................................................................. 118
4.2 Evolution of the vortex sheet for fixed $\delta$ and increasing final time. ........... 119
4.3 Measuring the angle at the core. ................................................................. 120
4.4 Time as a function of the sheet angle at the center for $\delta$ ranging from 0.01 to 0.10 in steps of $\Delta\delta = 0.05$. ..................................................... 122
4.5 Studying the time the sheet center angle $\theta$ achieves an angle $\theta$ as a function of smoothing variable $\delta$ for $\theta = \pi, 3\pi/2, \ldots, 6\pi$. ................. 123
4.6 The slope and intersection of each of the lines in the previous plot using a least-square linear fit. ................................................................. 124
4.7 Similarity studies result 1. ........................................................................... 127
4.8 Similarity studies result 2. ........................................................................... 128
4.9 Similarity studies result 3. ...................................................... 129

C.1 Memory hierarchy of today's computers. .......................... 142
CHAPTER 1

INTRODUCTION

1.1 Background

Two-dimensional parallel occurs when a fluid moving essentially in one direction has a gradient perpendicular to this direction. We are particularly interested in the case were this gradient is nonzero in a thin strip, this being then a thin strip of intense vorticity which is called a vorticity layer. The flow can be considered irrotational outside the vorticity layer. At high Reynolds number, the vorticity layer is very small, and in the limit that the Reynolds number approaches infinity, the vorticity layer is modelled as a sheet (of zero thickness) called a vortex sheet. The dynamics of the flow can then be approximated as the evolution of an inviscid vortex sheet in an irrotational flow in which no vorticity diffusion occurs.

Studying vortex sheet evolution is clearly easier than studying the full viscous motion of the vorticity layer. In the latter case, the full Navier-Stokes equations must be solved in two regions, the viscous region and the potential flow region far from the vorticity layer. The solutions from these two computational regions are then combined into one global solution. This involves 2D gridding techniques, and some form of asymptotics matching.
The replacement of a vorticity layer by a vortex sheet is not without disadvantages. The neglect of layer thickness and viscosity give rise to an ill-posed problem. The vortex sheet problem is susceptible to short-wave instability called the Kelvin-Helmholtz instability whereby a perturbation, no matter how small, will grow in time. In fact, as discussed by Saffman and Baker [18], the linearized problem is ill-posed in the sense of Hadamard. Such behavior wreaks havoc on numerical codes implemented on a computer where round-off errors can grow rapidly in time. Secondly, as reported by Moore [14], Meiron, Baker, and Orzag [2], and Krasny [9], the vortex sheet position develops a curvature singularity in finite time where it stops being analytic but is still continuous.

With these two prominent problems plaguing the studies of vortex sheets, why do mathematicians continue to study them? In fluid dynamics, many flows can be approximated by vortex sheets in an irrotational flow where calculation is much easier than the full Navier-Stokes viscous calculation. The instability of vortex sheets and their subsequent roll-up phenomena relate to small scale effects and turbulence. More importantly, many mathematicians believe that understanding the singular nature of the Kelvin-Helmholtz instability may lead to better understandings of other flows with singularities.

In 1931, Rosenhead [16] produced the first calculation of the roll-up of a periodic vortex sheet. He used point vortices to represent the sheet, but only used eight of them due to limitations of computational devices at the time. In 1959, Birkhoff and Fisher [3] tried to reproduce Rosenhead’s result with many more point vortices but found the motion of the point vortices to be irregular. In 1973, Chorin and Bernard [5] introduced the idea of a vortex blob to smooth out point vortices and
hence to smooth the fluid velocity. Krasny [10] in 1986 applied the blob methodology to desingularize the Birkhoff-Rott equation. The desingularized equation that Krasny produced restored well-posedness to the motion, and allows vortex sheet roll-up to be calculated. Krasny also suggested that for times both before and after the singularity time, solutions converge as $\delta \to 0$. Rigorous theory by Delort [6] and Majda [13] now guarantees that a weak solution exists, but may not be unique. Liu and Xin [12] showed that in the zero limit $\delta$ and grid size, the solution converges to a weak limit. The exact nature of the limiting solution is not known.

Another important issue relating to the use of desingularized equations is the relationship between the numerical solutions arising from the desingularized equations and actual physical solutions. In particular, is there any physical meaning to the smoothing parameter $\delta$? Tryggvason, Dahm, and Sbeih [21] in 1991 have compared numerical solutions from Krasny's desingularized equations and numerical solutions from the fully viscous Navier-Stokes equations at high Reynolds number. They discovered that the former reproduces only the large-scale features of the roll-up of the vorticity layer at finite thickness. However, as $\delta \to 0$ and as the Reynolds number approaches infinity, they found that the two numerical solutions appear to be the same. This is important since the underlying concept behind a vortex sheet is that it can be obtained as the zero viscosity limit of a sequence of smooth solutions to the Navier-Stokes equations. Many scientists and mathematicians believe that that studying vortex sheet motion may yield insights into the structure of high Reynolds number flow.
1.2 Thesis Outline

We study two classes of the desingularized Birkhoff-Rott equation. Krasny’s desingularized kernel belongs to one class. The second class of desingularized kernels arise from convolutions of an exact kernel with a smoothing function. We begin by numerically coding these four smoothed kernels on a computer. We provide tests to show that the calculations are done accurately. With the aim of efficient programming, we examine different computer algorithms to generate numerical solutions on serial and parallel computers taking advantage of cache architecture. A bracket table is provided to show approximately how many points are needed to compute a solution for a smoothing parameter $\delta$ with a certain accuracy. Then we venture to analyze the possible convergence of solutions at the outer regions of the vortex sheet as $\delta \to 0$. Finally, we examine a possible similarity relationship connecting the smoothing variable and time.

1.3 The Birkhoff-Rott Equation

In the following discussion, we follow the text of Ying and Zhang [22] on the topic of vortex methods. In two-dimension flow, the vorticity vector is $\zeta = (0,0,\omega)$. The vorticity function for two-dimension flow can therefore be considered as a scalar $\omega$. The 2D Euler equation for incompressible inviscid flow in terms of the vorticity variable is

$$\left[ \frac{\partial}{\partial t} + (\vec{u} \cdot \nabla) \right] \omega = 0$$

(1.1)

where $\vec{u} = (u,v)$. One can interpret the above equation as the vanishing of the material change in time of $\omega$. In other words, the vorticity is conserved along the trajectory of motion. Vortex methods take advantage of this property. Since there are
three unknowns, namely \( u, v, \) and \( \omega \), in Equation 1.1, we need two more equations. Fix some arbitrary point \( \vec{r}_0 \) in space. A stream function \( \psi \), a scalar function, is defined by

\[
\psi(\vec{r}) = \int_{C(\vec{r}; \vec{r}_0)} (-v \, dx + u \, dy),
\]

where \( \vec{r} = (x, y) \), and \( C(\vec{r}; \vec{r}_0) \) is some path from \( \vec{r}_0 \) to \( \vec{r} \). It is well-known that \( \psi \) is related to the velocity by

\[
(u, v) = \left( -\frac{\partial}{\partial y}, \frac{\partial}{\partial x} \right) \psi(\vec{r}) = \nabla \wedge \psi(\vec{r}).
\]

Now apply the operator \( \nabla \wedge \) to the above equation to obtain a relationship between \( \psi \) and \( \omega \) by the Laplacian operator

\[
-\nabla^2 \psi = \omega.
\]

If \( \omega \) is known, then one can solve equation (1.4) to obtain \( \psi \). Then equations (1.3) gives \( u \) and \( v \) which in turn can be used to update \( \omega \) in equation (1.1).

Since equation (1.1) is a first-order hyperbolic partial differential equation for the dependent variable \( \omega \), the vorticity along any characteristic curve is invariant. Like all hyperbolic problems, we need to know what the characteristic curves are, and the initial distribution of the vorticity function.

The point vortex method approximates a continuous distribution of vorticity, say a vortex sheet, by finite number of point vortices at location \( r_j \) with strength \( \alpha_j \). Note that the strength of the point vortex is the circulation. The vorticity distribution has the form

\[
\omega_0(\vec{r}) = \sum_{j=0}^{N} \alpha_j \cdot \delta(\vec{r} - \vec{r}_j).
\]
If $\bar{r}_j(t)$ is the solution to $d\bar{r}/dt = \bar{u}$ satisfying the initial condition $\bar{r}(0) = \bar{r}_j$, then the vorticity will be invariant along the curve $\bar{r}_j(t)$. In other words,

$$\omega(\bar{r}, t) = \sum_{j=0}^{N} \alpha_j \cdot \delta(\bar{r} - \bar{r}_j(t)).$$

(1.6)

To obtain the stream function, we solve

$$-\nabla^2 \psi = \sum_{j=0}^{N} \alpha_j \cdot \delta(\bar{r} - \bar{r}_j(t)).$$

(1.7)

The above equation reduces to solving

$$-\nabla^2 \psi = \delta(\bar{r} - \bar{r}_o).$$

(1.8)

which has solution

$$\psi = -\frac{1}{2\pi} \log |\bar{r} - \bar{r}_o|.$$

(1.9)

By linear superposition, we obtain

$$\psi(\bar{r}, t) = -\frac{1}{2\pi} \sum_{j=0}^{N} \alpha_j \log |\bar{r} - \bar{r}_j(t)|.$$

(1.10)

Now apply the operator $\nabla \wedge$ to $\psi$ to obtain the velocity, or

$$(u, v) = -\frac{1}{2\pi} \sum_{j=0}^{N} \alpha_j \left( \frac{y - y_j(t) \cdot \bar{u} - x + x_j(t)}{(x - x_j(t))^2 + (y - y_j(t))^2} \right).$$

(1.11)

Equation (1.11) is basically the discrete form of the Biot-Savart law. If we restrict the vorticity initially to lie along some curve, then it will remain on the curve as it evolves according to the velocity $\bar{u}$. If we use the circulation variable $\xi$ to parameterize the curve, then the continuous version of equation (1.11) is

$$(u(\xi), v(\xi)) = -\frac{1}{2\pi} \int \left( \frac{y(\xi) - y(\zeta) \cdot x(\xi) - x(\zeta)}{(x(\xi) - x(\zeta))^2 + (y(\xi) - y(\zeta))^2} \right) d\xi.$$

(1.12)

where we have evaluated the velocity at the point $(x(\xi), y(\xi))$ along the curve. For convenience, we have dropped the dependency in time in the variables $x$ and $y$.  

6
Finally, we point out that equation (1.12) may be written in a more compact form by introducing the complex location of the curve \( z(\xi) = x(\xi) + iy(\xi) \). The Birkhoff-Rott equation is then

\[
\frac{\partial z}{\partial t} = \frac{1}{2\pi i} \int \frac{d\zeta}{z(\xi) - z(\zeta)}.
\] (1.13)

1.4 Notations

We use the following notations throughout this thesis

\[ \mathbb{R} : \text{The set of all real numbers.} \] (1.14)

\[ \mathbb{Z} : \text{The set of all integers.} \] (1.15)

\[ i : \text{The complex number } \sqrt{-1}. \] (1.16)

The set \( \mathbb{Z}[N,M] \) contains all integers in the segment \([N,M]\) of the real line. We use the variables \( \xi \) and \( \zeta \) to denote the continuous parameters of the vortex sheet with \( p \) and \( q \) their discrete counterparts.
CHAPTER 2

FORMULATION OF THE PROBLEM

2.1 The Exact Vortex Sheet Kernels

The vortex sheet position is parameterized by the circulation variable \( \xi \) which measures the total circulation between a fixed material point and an arbitrary material point along a curve [10]. As discussed in Chapter 1, the sheet position in complex notation is described by the variable \( z(\xi, t) = x(\xi, t) + iy(\xi, t) \). Its motion is governed by the Birkhoff-Rott equation

\[
\frac{\partial z}{\partial t}(\xi, t) = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \frac{d\zeta}{z(\xi, t) - z(\zeta, t)}. \tag{2.1}
\]

where \( \bar{z} \) denotes the complex conjugation of \( z \). The integral is interpreted as a principal-valued integral due to the singularity of the integrand as \( \zeta \to \xi \). We call the integrand \( K_{x} \), the \text{exact vortex sheet kernel over the real line}. The integral of the vortex sheet kernel \( K_{x} \) (over the domain \( \mathbb{R} \)) gives the instantaneous velocity at each point on the sheet. One can write \( K_{x} \) in an alternative form in which the denominator is a real quantity

\[
K_{x} = \frac{1}{z(\xi, t) - z(\zeta, t)} \tag{2.2}
\]

\[
= \frac{(x(\xi) - x(\zeta)) - i(y(\xi) - y(\zeta))}{(x(\xi) - x(\zeta))^2 + (y(\xi) - y(\zeta))^2}. \tag{2.3}
\]
Performing an integration over an unbounded region is a task most numerical scientists would rather avoid. We adopt the same viewpoint and seek a simplification of the problem by studying periodic vortex sheets. The position variables then conform to the rule

\[
\left( x(\xi + 2\pi k) \cdot y(\xi + 2\pi k) \right) = \left( 2\pi k + x(\xi) \cdot y(\xi) \right) \quad \forall k \in \mathbb{Z}.
\] (2.4)

The periodic vortex sheet profile consequently can be parameterized by \( \xi \) in a finite domain \([0, 2\pi]\). Due to the periodicity of the sheet, \( \xi = 2\pi \) is a sheet parameter but is not considered in the numerical calculation.

The equation of motion must be transformed to an equation defined only over \([0, 2\pi]\). This task involves the transformation of \( K_\mathbb{R} \) to a form for which its integral over \([0, 2\pi]\) evaluates the instantaneous velocity of the sheet at each point. The idea of the transformation is to add contributions from all intervals of length \( 2\pi \) of the type \([2-k\pi, 2-k\pi(k+1)]\) for all \( k \in \mathbb{Z} \). Mathematically, this is equivalent to

\[
\int_{-\infty}^{\infty} K_\mathbb{R}(\xi, \zeta) \, d\zeta = \sum_{k \in \mathbb{Z}} \int_{2\pi(k-1)}^{2\pi(k)} K_\mathbb{R}(\xi, \zeta) \, d\zeta.
\] (2.5)

This process is called the **method of images**. It transforms the kernel \( K_\mathbb{R} \) into a kernel applicable over a finite domain \([0, 2\pi]\), and we call the latter the **periodic version** of \( K_\mathbb{R} \). Appendix A describes the method of images to transform the equation of motion to

\[
\frac{\partial z}{\partial t}(\xi) = -\frac{1}{4\pi} \int_0^{2\pi} \frac{\sinh(y(\xi) - y(\zeta)) + i \sin(x(\xi) - x(\zeta))}{\cosh(y(\xi) - y(\zeta)) - \cos(x(\xi) - x(\zeta))} \, d\zeta.
\] (2.6)

Due to the singularity of the integrand, the integral must be viewed as a principal-valued integral. The equation (2.6) is an exact formulation of vortex sheet motion in
a periodic domain. The integrand is denoted by $K_{|0,2\pi|}$ and is given by

$$K_{|0,2\pi|}(\xi, \zeta) = \frac{\sinh (y(\xi) - y(\zeta)) + i \sin (x(\xi) - x(\zeta))}{\cosh (y(\xi) - y(\zeta)) - \cos (x(\xi) - x(\zeta))}. \quad (2.7)$$

We call $K_{|0,2\pi|}$ the exact periodic kernel, which is simply the periodic version of the exact kernel $K_{\pi}$ over the real line.

### 2.2 The Desingularized Equation

In a $2\pi$-periodic geometry, vortex sheet motion is governed exactly by

$$\frac{\partial z}{\partial t}(\xi) = -\frac{1}{4\pi} \int_0^{2\pi} K_{|0,2\pi|}(\xi, \zeta) \, d\zeta. \quad (2.8)$$

is not considered as a sheet parameter. A smoothed periodic kernel $K_\delta(\xi, \zeta)$ is an approximation of $K_{|0,2\pi|}(\xi, \zeta)$ with a parameter $\delta > 0$ introduced to remove the singularity at $\zeta = \xi$. The parameter $\delta$ is called the smoothing or regularization parameter. The resulting equation of motion is said to be a desingularized equation. More specifically, the desingularized equation of motion is (2.8) with $K_\delta(\xi, \zeta)$ in place of $K_{|0,2\pi|}$.

In this section, two regularizations (or smoothing) of $K_{\pi}$ are given. Their periodic versions form one type of smoothed vortex sheet kernels. Another type arises from the modification of $K_{|0,2\pi|}$ directly.

#### 2.2.1 The Blob Method

It is a well-documented fact [2] that exact vortex sheet motion leads to a curvature singularity in finite time. The presence of a curvature singularity prevents any meaningful numerical solutions beyond the singularity time $t_c(\approx \frac{3}{8}(2\pi))$ [10]. To avoid the ill-posed nature of vortex sheet motion, Chorin and Bernard [5] introduced
the idea of a smoothed kernel, which has since became known as the **blob method**. They regularized the Birkhoff-Rott equation by adding a $\delta^2$ term to the denominator of $K_\delta$ to obtain

$$K_{\delta,\text{reg}}^{(1)} = \frac{\bar{z}(\xi, t) - \bar{z}(\zeta, t)}{|z(\xi, t) - z(\zeta, t)|^2 + \delta^2}. \quad (2.9)$$

The superscript (1) indicates that equation (2.9) is the first regularization of the exact kernel $K_\delta$ we examine in this thesis. It is a regularization of a singular kernel in the sense that the regularized kernel (2.9) possesses no singularity as $\zeta \to \xi$. Consequently, the motion of the vortex sheet is **regular**.

### 2.2.2 Krasny's and Baker's Kernels

To simplify writing, let us now apply the notation $x = x(\xi)$, $\bar{x} = x(\zeta)$, and similarly for the variable $y$.

Krasny followed the same spirit that inspired the blob method. He modifies the periodic version $K_{0,2\pi}$ of the vortex sheet kernel by adding a $\delta^2$ term to the denominator to obtain

$$K_{\text{Krasny}}(\xi, \zeta) = \frac{\sinh (y - \bar{y}) + i \sin (x - \bar{x})}{\cosh (y - \bar{y}) - \cos (x - \bar{x}) + \delta^2}. \quad (2.10)$$

One observes that Krasny's kernel is not the periodic version of $K_{\delta,\text{reg}}^{(1)}$. By the method of images, Baker (private communication) obtained the periodic version of $K_{\delta,\text{reg}}^{(1)}$ which can be written in real and imaginary parts as

$$\mathcal{D} = \cosh \sqrt{(y - \bar{y})^2 + \delta^2} - \cos (x - \bar{x}). \quad (2.11)$$

$$\text{Re } K_{\text{Baker}}(\xi, \bar{\xi}) = \frac{(y - \bar{y})}{\sqrt{(y - \bar{y})^2 + \delta^2}} \frac{-\sinh \sqrt{(y - \bar{y})^2 + \delta^2}}{\mathcal{D}}. \quad (2.12)$$

$$\text{Im } K_{\text{Baker}}(\xi, \bar{\xi}) = \frac{\sin (x - \bar{x})}{\mathcal{D}}. \quad (2.13)$$
The importance of Baker’s kernel is its origin. The regularization $A_{\lambda, \text{reg}}^{(1)}$ can be viewed as the convolution of $K_{\lambda}$ with an algebraic cut-off function

$$\phi(\vec{r}) = \frac{1}{\pi} \left( \frac{1}{1 + |\vec{r}|^2} \right)^2.$$  \hspace{1cm} (2.14)

The function $\phi$ is often called the blob function. Baker’s kernel is just the periodic version of a regularization which arises from a convolution of the exact kernel with an algebraic blob function. As such, by the work of Liu and Xin [12], solutions of the desingularized equation with Baker’s kernel are guaranteed to possess a weak limit (to be defined later) as the grid size and smoothing parameter $\delta$ tend to zero.

The question now is whether Krasny’s kernel, which does not arise from a convolution with a blob function, yields solutions that have a weak limit as the smoothing parameter $\delta \to 0$. In other words, is the hypothesis of a smoothed kernel being a convolution necessary for existence of weak limit solutions? If the answer is in the negative, are approximate solutions of the Birkhoff-Rott equation from Baker’s and Krasny’s kernels the same?

### 2.2.3 Another Regularization Method

Beale created another smoothed kernel of the form $K_{\lambda} \cdot f(|\vec{r}|/\delta)$ by using a convolution with the blob function $\phi(r) = e^{-r^2}/\pi$. With

$$f(r) = 2\pi \int_0^r s\phi(s) \, ds.$$  \hspace{1cm} (2.15)

he was led to the smoothed kernel (over $\mathbb{R}$)

$$K_{\lambda, \text{reg}}^{(2)} = \frac{1}{z(\xi, t) - z(\zeta, t)} \cdot \left[ 1 - e^{-z(\xi, t) - z(\zeta, t)/\delta^2} \right].$$  \hspace{1cm} (2.16)

The periodic version of $K_{\lambda, \text{reg}}^{(2)}$ based on the method of images, is given by

$$K_{\text{Beale}}^{(2)} = \frac{\sinh(y - \tilde{y}) + i \sin(x - \tilde{x})}{\cosh(y - \tilde{y}) - \cos(x - \tilde{x})} - \eta \tilde{z}.$$  \hspace{1cm} (2.17)
\[ \eta_2 = \sum_{n \in \mathbb{Z}} \frac{(x - \tilde{x} + 2\pi n) + i(y - \tilde{y})}{(x - \tilde{x} + 2\pi n)^2 + (y - \tilde{y})^2} e^{-((x - \tilde{x} + 2\pi n)^2 + (y - \tilde{y})^2)/\delta^2}. \] (2.18)

We call this kernel **Beale's second kernel** with the label *first* reserved for Beale's simpler kernel to be defined momentarily.

In the same spirit as Krasny, Beale provided a simpler smoothing of \( K_{0,2\pi} \) by multiplying it with a cut-off factor. The result is

\[ K_{\text{Beale}}^{(1)}(\xi, \zeta) = \frac{\sinh(y - \tilde{y}) + i \sin(x - \tilde{x})}{\cosh(y - \tilde{y}) - \cos(x - \tilde{x})} \cdot \eta_1 \] (2.19)

where

\[ \eta_1 = \left[ 1 - e^{-2(\cosh(y - \tilde{y}) - \cos(x - \tilde{x})/\delta^2)} \right]. \] (2.20)

We refer to Equation 2.19 as **Beale's first kernel**.

### 2.2.4 Two classes of smoothed kernels

We have presented four different smoothed \( 2\pi \)-periodic kernels to approximate the exact formulation \( K_{0,2\pi} \). We can classify them into two important classes.

**Class A of Regularized Kernels**

The class A kernels are smoothings of the exact kernel \( K_{0,2\pi} \). These include Krasny’s kernel and Beale’s first kernel.

**Class B of Regularized Kernels**

The second class of smoothed kernels consists of convolutions of the exact kernel \( K_{2\pi} \) with a blob function. The first of these is Baker’s kernel where the blob function is an algebraic cut-off. The other kernel in this class is Beale’s first kernel, and the blob function is an exponential cut-off. These kernels are important since there is rigorous theory guaranteeing the existence of a weak limit solution as the smoothing parameter tends to zero.
Some Comparisons Between These Two Classes

We observe immediately that class B smoothed kernels are more expensive to compute than class A kernels. This is especially true of Beale’s second kernel where an infinite number of exponentials must be summed. Since the exponential terms are dominated by $e^{-i\pi^2n^2 \delta^2}$, computationally it makes sense to sum only the first few terms. In my code, I sum only three terms corresponding to the indices $n = -1, 0,$ and $1$.

The important question is which one of the four smoothed kernels reveal the limiting behavior of small $\delta$ the fastest? After all, the goal of using a desingularized equation is to learn about the behavior of the solution as $\delta \to 0$. The faster $\delta$ tends to zero, the better. From the appearance of the smoothed kernels, it would seem that Krasny’s kernel is the slowest since it contains the term $\delta^2$ as a summand in the denominator. Beale’s kernels appear the fastest since they involve exponential terms, although a superficial inspection cannot reveal the any relationship between Beale’s first and second kernel. In Baker’s kernel, it is not a priori clear how the square root term is connected to the $\delta$-influence of the kernel.

Let’s now take a brief look at the differences between these kernels in regards to computational complexity. In Table 2.1, we tabulated the time it takes to evaluate the velocity for all points on a 2048-point grid. At each grid point, two integrals are evaluated to calculate $u$ and $v$, giving a total of 4096 integrals. We observe that Beale’s second kernel is very computationally demanding due to evaluations of exponential terms.
Table 2.1: Time comparison between the four kernels. Normalization is with respect to Krasny's time.

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (sec.)</th>
<th>Normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Krasny</td>
<td>10.465</td>
<td>1.00</td>
</tr>
<tr>
<td>Baker</td>
<td>11.016</td>
<td>1.05</td>
</tr>
<tr>
<td>Beale's First</td>
<td>13.749</td>
<td>1.31</td>
</tr>
<tr>
<td>Beale's Second</td>
<td>33.398</td>
<td>3.19</td>
</tr>
</tbody>
</table>

2.3 Numerical Implementation

Recall that the desingularized equation of vortex sheet motion has the form

$$\frac{\partial \zeta}{\partial t}(\xi, t) = -\frac{1}{4\pi} \int_{0}^{2\pi} K_\delta(z(\xi, t), z(\zeta, t)) \, d\zeta: \quad \xi \in [0, 2\pi).$$  \hspace{1cm} (2.21)

where $K_\delta$ is one of the four smoothed kernels introduced earlier. We shall restrict our calculation to a final time $T = 2\pi$ which is past the critical time $t_c(\equiv \frac{3}{8}(2\pi))$ where the vortex sheet develops a curvature singularity when $\delta = 0$. The initial condition to be employed is Krasny's initial condition \cite{10}

$$z(\xi, 0) = \xi + \frac{2\pi}{100} (1 - i) \sin \xi. \hspace{1cm} (2.22)$$

It is a small perturbation of a flat vortex sheet by one sinusoidal mode of amplitude $\pi/50$. A graph of the initial vortex sheet location is shown in Figure 2.1. To solve the problem numerically, a uniform $N$-point grid for the parameter space $[0, 2\pi]$ is constructed. The continuous vortex sheet is approximated by point vortices, or vortex markers. With $h = 2\pi/N$, the set of discrete markers of the sheet is written as

$$\{ z_p = (x_p, y_p) = (x(p \cdot h), y(p \cdot h)) : p \in \mathbb{Z}[0, N - 1] \}. \hspace{1cm} (2.23)$$
Figure 2.1: Initial vortex sheet position over one period.
Regarding notations, we use \( \xi, \zeta \in [0, 2\pi] \) to denote the continuous parameter of the sheet, and \( p, q \in \mathbb{Z}[0, N-1] \) to denote the integer grid parameter. Since the problem is periodic, we don’t need to consider \((x_N, y_N)\), which is equal to \((x_0 + 2\pi, y_0)\). The time-evolution of each position marker is given by a set of \( N \) coupled ordinary differential equations

\[
\frac{d}{dt} \tilde{z}_p(t) = -\frac{1}{4\pi} \int_0^{2\pi} K_\delta(z_p(t), z(\zeta, t)) \, dq. \quad \text{for} \ p = 0, 1, \ldots, N-1. \tag{2.24}
\]

Note that the variable \( z_p(t) \) is a discrete variable, but \( z(\zeta, t) \) is a continuous variable.

To discretize equation (2.24), I use the spectrally-accurate trapezoidal approximation of a definite integral

\[
\frac{d}{dt} \tilde{z}_p(t) = -\frac{1}{4\pi} \frac{2\pi}{N} \sum_{q \neq p}^{N-1} K_\delta(z_p, z_q). \quad \text{for} \ p = 0, 1, \ldots, N-1. \tag{2.25}
\]

There are two numerical steps involved in solving the system equations (2.25) numerically. The first step is the evaluation of the sum to obtain the instantaneous velocity. Secondly, the set of points \( \tilde{z}_p \) must be updated by an ODE solver.

### 2.3.1 Implementing the ODE solvers

At each time step, an initial-valued ordinary differential equation must be solved to obtain the position of the \( p \)th vortex marker. To accomplish this task, an ODE solver is implemented to make use of the Runge-Kutta and Adams-Bashford-Moulton methods. We now examine one position marker, and discuss how we numerically update its position as a function of time. For the generic form

\[
\vec{r}(t) = (x(t), y(t)),
\]

\[
\frac{d}{dt} \vec{r}(t) = \vec{f}(\vec{r}(t)). \quad t \in (0, T). \tag{2.27}
\]

\[
\vec{r}(0) = \vec{u} \text{ given.} \tag{2.28}
\]
a uniform time grid for the time parameter space $[0, T]$ with $\Delta t = T/M$ is constructed.
and the variable $\bar{r}$ on this grid is $\bar{r}_k = \bar{r}(k \cdot \Delta t)$. Clearly $\bar{r}_0$ is the given initial value. We
must obtain $\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_M$. We use the two-step fourth-order Runge-Kutta method
to obtain $\bar{r}_1, \bar{r}_2, \text{ and } \bar{r}_3$. Assume $\bar{r}_n$ is already computed. the Runge-Kutta method
computes $\bar{r}_{n-1}$ in the following way:

$$
\bar{k}_1 = \bar{f}(\bar{r}_n), \quad (2.29)
$$

$$
\bar{k}_2 = \bar{f} \left( \bar{r}_n + \frac{\Delta t}{2} \cdot \bar{k}_1 \right), \quad (2.30)
$$

$$
\bar{k}_3 = \bar{f} \left( \bar{r}_n + \frac{\Delta t}{2} \cdot \bar{k}_2 \right), \quad (2.31)
$$

$$
\bar{k}_4 = \bar{f}(\bar{r}_n + \Delta t \cdot \bar{k}_3), \quad (2.32)
$$

$$
\bar{r}_{n-1} = \bar{r}_n + \frac{\Delta t}{6} \cdot (\bar{k}_1 + 2\bar{k}_2 + 2\bar{k}_3 + \bar{k}_4). \quad (2.33)
$$

Four function evaluations are necessary to compute $\bar{r}_{n-1}$ from $\bar{r}_n$. Now we apply a
less-costly method to compute the rest of the grid points $\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_M$.

The fourth-order Adams-Bashford-Moulton (ABM) corrector-predictor method
computes $\bar{r}_{n-1}$ from previous four grid data points $\bar{r}_n$, $\bar{r}_{n-1}$, $\bar{r}_{n-2}$, and $\bar{r}_{n-3}$ by the
following prescription

$$
\bar{r}_{n-1}^{\text{pred}} = \bar{r}_n + \frac{\Delta t}{24} \left[ 55 \bar{f}(\bar{r}_n) - 59 \bar{f}(\bar{r}_{n-1}) + 37 \bar{f}(\bar{r}_{n-2}) - 9 \bar{f}(\bar{r}_{n-3}) \right], \quad (2.34)
$$

$$
\bar{r}_{n-1} = \bar{r}_n + \frac{\Delta t}{24} \left[ 9 \bar{f}(\bar{r}_{n-1}^{\text{pred}}) + 19 \bar{f}(\bar{r}_n) - 5 \bar{f}(\bar{r}_{n-1}) + \bar{f}(\bar{r}_{n-2}) \right] \quad (2.35)
$$

At any step, since $\bar{f}(\bar{r}_{n-1}), \bar{f}(\bar{r}_{n-2}), \text{ and } \bar{f}(\bar{r}_{n-3})$ were already computed at the previous
iteration, only two function evaluations $\bar{f}(\bar{r}_n)$ and $\bar{f}(\bar{r}_{n-1}^{\text{pred}})$ are needed. However,
the ABM method has ten arithmetic operations compared to the Runge-Kutta’s five
operations after function evaluations. Clearly the ABM method is computationally
cheaper than the Runge-Kutta method since function evaluation is computationally
much more expensive than arithmetics operations which are optimized by hardware.
Now that we know how to update the position of each vortex marker, we can apply the same technique to all other markers. Once the instantaneous velocity at each point is calculated, updating one marker is independent of another. The numerical approach to updating all the vortex markers takes the form of a vector arithmetic operation. On a serial computer, there is not much software optimization one can introduce to speed up vector arithmetic operations.

2.3.2 Calculating the Instantaneous Velocity

At each parameter grid point $p$, the instantaneous velocity $(u_p, v_p)$ must be calculated to update the position. The velocity calculation is an integral of a smoothed kernel over the domain $[0, 2\pi)$. It is numerically computed as a sum of $N - 1$ terms. One can view the the summation operation generically as

$$u_p + i v_p = \sum_{\substack{q=0 \atop q \neq p}}^{N-1} f(p; q) + i g(p; q). \quad (2.36)$$

The terms $f(p; q)$ and $g(p; q)$ may be viewed as the real and imaginary contributions to the point at parameter $p$ from the point at parameter $q$, respectively. At each parameter point $p$, the contributions from all other points (except from $p$) must be
Figure 2.3: Naive way of computing the velocity. The point that the arrow points to is the point with parameter $p$ at which the velocity calculation is being done, and the point at the source of the arrow is the point with parameter $q$. At each point $p$, $q$ ranges over all parameter points except $p$. Also shown is the iteration step.
calculated to obtain the velocity at the point \( p \). This process is shown in Figure 2.2. In Figure 2.3, we show how this is done for three partitions of the parameter space \([0, 2\pi]\). We also show which point is used in each step. The parameter point \( 2\pi \) is shown as an open circle since it is not directly involved in the summation.

This algorithm is really a naive way of computing the instantaneous velocity. For \( N \) partitions of the parameter space, this naive method takes \( N(N - 1) \) steps. The calculation of the velocity is innately \( O(N^2) \) which cannot be changed. However, we can try to make the actual number of computational steps smaller than \( N(N - 1) \). Before we examine better algorithms, let us make sure that the most crudest algorithm works. We will use this algorithm both to debug better-performance algorithms, and to see how much better the other algorithms are. The pseudo code for the algorithm is given below.

\[
\text{for } p=0 \text{ to } N-1 \text{ do} \\
u[p] = v[p] = 0; \\
\text{for } q=0 \text{ to } N-1 \text{ do} \\
\quad \text{if } q \neq p \text{ then update} \\
\quad \quad u[p] = u[p] + f(p; q); \\
\quad \quad v[p] = v[p] + g(q; p);
\]

### 2.3.3 Debugging the code

Our serial code is written in C++ using double precision. It is executed on the Ohio State University Math Computation Node: quad-processor AMD Athlon 800 Mhz. computer. The machine epsilon \( \epsilon_M \), defined to be the smallest \( \epsilon > 0 \) such that \( 1 + \epsilon > 1 \), was computed to be approximately

\[
\epsilon_M = 1.0842 \times 10^{-19}. \tag{2.37}
\]
Our program is written to take advantage of processor properties such as pipelined execution of parallel instructions, and partial loop unrolling whenever possible. We shall investigate software and hardware optimization properties further in the next few sections. All graphs in this thesis are displayed using Matlab. Figures are created by Xfig.

For periodic functions, in particular, for a $2\pi$-periodic function $G$, the trapezoidal rule to approximate

$$\int_0^{2\pi} G(x) \, dx$$

is spectrally accurate. For any $N > 0$, there exists some real constant $C$ and $\alpha > 0$ such that

$$\int_0^{2\pi} G(x) \, dx = \sum_{k=0}^{N-1} G\left(\frac{2\pi k}{N}\right) + C \cdot e^{-\alpha N}. \quad (2.38)$$

One large section of our code is dedicated to computing the instantaneous velocity at each time step for each vortex marker. It is therefore essential that this part of the code performs correctly.

To test the numerical evaluation of the velocity, a vortex sheet profile must be specified. One profile that is readily available is the sheet profile at time $t = 0$ given by Krasny’s initial condition (see equation (2.1)). We are interested in the accuracy of the integral evaluation of the smoothed kernel. Therefore it make sense to let $\delta$ stay fixed at some constant.

For the first test case, we fix $\delta = 0.1$ and choose a parameter $\xi \in [0, 2\pi)$ associated to some grid point $p_N(\xi) \in \mathbb{Z}[0, N - 1]$. As an example, for $\xi = \pi/2$ and $N = 32$, the discrete grid point must be $p = p_N(\xi) = 8$.

We next compute the “exact” instantaneous velocity $(u(p_M(\xi)), v(p_M(\xi)))$ by numerically integrating on an $M$-point grid. The value of $M$ should be large enough so
Table 2.2: Testing accuracy of the velocity calculation for Krasny's Kernel at $\xi = \pi/2$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$c_{X,M}^u(\xi;0.1)$</th>
<th>$c_{X,M}^v(\xi;0.1)$</th>
<th>$c_{X,M}^u(\xi;0.05)$</th>
<th>$c_{X,M}^v(\xi;0.05)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>9.9297</td>
<td>8.98325</td>
<td>8.92058</td>
<td>8.92058</td>
</tr>
<tr>
<td>128</td>
<td>23.5073</td>
<td>20.7251</td>
<td>11.3704</td>
<td>11.3704</td>
</tr>
<tr>
<td>256</td>
<td>-</td>
<td>-</td>
<td>19.256</td>
<td>16.8765</td>
</tr>
</tbody>
</table>

Table 2.3: Testing accuracy of the velocity calculation for Baker's Kernel at $\xi = \pi/2$.

that differences in results obtained from an $M$-point grid and a $2M$-point grid is on the order of round-off errors. In double precision, it should be on the order of $10^{-11}$.

For each $N > 0$, we are interested in the accuracy of the velocity evaluation on an $N$-point grid compared to the same calculation on an $M$-point grid (the "exact" case). One way to measure accuracy is to take the negative of the natural log of the error. We define $c_{X,M}^u$ and $c_{X,M}^v$ by

$$c_{X,M}^u(\xi; \delta) = -\ln |u(p_M(\xi)) - u(p_N(\xi))|.$$  \hspace{1cm} (2.39) \\
$$c_{X,M}^v(\xi; \delta) = -\ln |v(p_M(\xi)) - v(p_N(\xi))|.$$  \hspace{1cm} (2.40)

Note that the number of digits of accuracy of $u$ and $v$ (evaluated at $\xi$) are calculated by the base-10 logarithm of the error, or by multiplying $c_{X,M}^u$ and $c_{X,M}^v$ by $\log_{10} e$, respectively. Tables 2.2 to 2.5 list values of $c_{X,M}^u(\xi; \delta)$ and $c_{X,M}^v(\xi; \delta)$ for $\xi = \pi/2$ at two
Figure 2.4: Plots of $c_N^k$ and $c_{N}^k$ for all four smoothed kernels at $\delta = 0.10$. 

24
Figure 2.5: Plots of $c_N^*$ and $c_N^u$ for all four smoothed kernels at $\delta = 0.05$. 

25
Table 2.4: Testing accuracy of the velocity calculation for Beale's First Kernel at \( \xi = \pi/2 \).

<table>
<thead>
<tr>
<th>( N )</th>
<th>( c^u_{X,M}(\xi:0.1) )</th>
<th>( c^v_{X,M}(\xi:0.1) )</th>
<th>( c^u_{X,M}(\xi:0.05) )</th>
<th>( c^v_{X,M}(\xi:0.05) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>8.89515</td>
<td>7.75834</td>
<td>7.52661</td>
<td>9.25461</td>
</tr>
<tr>
<td>64</td>
<td>16.5207</td>
<td>13.6013</td>
<td>9.58813</td>
<td>8.44885</td>
</tr>
<tr>
<td>128</td>
<td>38.5931</td>
<td>35.484</td>
<td>17.2547</td>
<td>14.3372</td>
</tr>
<tr>
<td>256</td>
<td>-</td>
<td>-</td>
<td>37.8046</td>
<td>34.1295</td>
</tr>
</tbody>
</table>

Table 2.5: Testing accuracy of the velocity calculation for Beale's Second Kernel at \( \xi = \pi/2 \).

<table>
<thead>
<tr>
<th>( N )</th>
<th>( c^u_{X,M}(\xi:0.1) )</th>
<th>( c^v_{X,M}(\xi:0.1) )</th>
<th>( c^u_{X,M}(\xi:0.05) )</th>
<th>( c^v_{X,M}(\xi:0.05) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>8.89316</td>
<td>7.75564</td>
<td>7.5264</td>
<td>9.25413</td>
</tr>
<tr>
<td>64</td>
<td>16.5739</td>
<td>13.6534</td>
<td>9.58763</td>
<td>8.44818</td>
</tr>
<tr>
<td>128</td>
<td>37.8046</td>
<td>36.3313</td>
<td>17.2682</td>
<td>14.3505</td>
</tr>
<tr>
<td>256</td>
<td>-</td>
<td>-</td>
<td>38.0053</td>
<td>34.5894</td>
</tr>
</tbody>
</table>

different \( \delta \), 0.10 and 0.05, for all four methods. Only results for \( N \) in powers of two are listed. Results for a wider range of \( N \) are tabulated in Appendix B.

These tables show a trend shared by the four smoothed kernels. For a fixed \( \delta \), Baker's kernel requires a finer grid than Krasny's, and Beale's two kernels require a finer grid than Baker's. Now returning to equation (2.38), since \( c^u_{X,M}(\xi:\delta) \) and \( c^v_{X,M}(\xi:\delta) \) are just the natural logarithm of the error in \( u \) and \( v \) respectively, we have the following relations

\[
\begin{align*}
  c^u_{X,M}(\xi:\delta) & = \ln C_1 - \alpha_1 N. \\
  c^v_{X,M}(\xi:\delta) & = \ln C_2 - \alpha_2 N.
\end{align*}
\]

(2.41)  
(2.42)
for positive real constants $C_1, C_2, \alpha_1,$ and $\alpha_2$. Equations (2.41) and (2.42) imply that the plots of $c_{N,M}^\pm(\xi; \delta)$ and $c_{N,M}^\pm(\xi; \delta)$ versus $N$ are straight lines. In Figures 2.4 to Figures 2.5, the data in Appendix B are plotted. In all cases, there is some region for which a group of data points fall on a straight line. In some cases, the positive concavity of the curves indicates an accuracy slightly better than spectral accuracy.

Recall that the smoothed kernel $K_\delta(p,q)$ is a function of two variables. One can examine the graphs of its real and imaginary parts to understand why the evaluations of $u$ and $v$ are so difficult. Let's call $K_x$ and $K_y$ the $x$ and $y$-component of the smoothed kernel that is being used to obtain $u$ and $v$, respectively. Figures 2.6 to 2.7 show the plots of $K_x$ and $K_y$ as functions of $\zeta$ (fixing $\xi$) for the four methods with $\delta = 0.1$. While the profiles of $K_y$ for Krasny's, Baker's, and Beale's kernels are similar, there is a noticeable difference in the graphs of $K_x$ for the four methods. There is a region in which $K_x$ changes rapidly for all four kernels. This region appears to be smaller for Beale's kernels than it is for both Krasny's and Baker's kernels. This result suggests the $\delta$-influence in Beale's kernels are more local than either Krasny's or Baker's kernels.

### 2.4 Performance Consideration

In this section, different algorithmic improvements are considered in attempts to speed up code execution. As mentioned earlier, there are two numerical tasks to perform. Not much can be done to enhance the efficiency of the ODE solver. The velocity calculation, however, leaves much room for improvement.

The velocity calculation involves summing $2(N - 1)$ terms for each of the $N$ parameter points. We provided one way of computing these sums in the last section.
Figure 2.6: Krasny's and Baker's Kernels.
Figure 2.7: Beale's 1st and 2nd Kernels.
It was noted that the method used can be viewed as a naive method because the code does not take advantage of cache performance. For a brief introduction to computer cache architecture, see Appendix C. We now examine several ways to reorder the sum operations to improve efficiency. Some techniques are successful while others are not.

### 2.4.1 The Fast Algorithm in Full Range

We consider the first "smart" discrete velocity-evaluation algorithm that takes advantage of the symmetry in the smoothed kernel to rearrange the order of the sum calculation. Recall that the computation of the instantaneous velocity at grid point $p$ is given in generic form by

$$u_p + iv_p = \sum_{q=0}^{N-1} f(p; q) + ig(p; q).$$

There is a symmetry in all four smoothed kernels considered in this thesis that allows for a faster summation method. This feature is revealed, for example, by Krasny's kernel, but the same method works for the other three smoothed kernels. In discrete form, $x(p; h)$ (where $h = 2\pi/N$) is replaced by the $x$ profile at the $p$th grid point $x_p$, and similarly for the variable $y$. With Krasny's kernel given in discrete form by

$$f(p; q) = \frac{\sinh (y_p - y_q) + i \sin (x_p - x_q)}{\cosh (y_p - y_q) - \cos (x_p - x_q) + \delta^2},$$

it is almost a trivial fact that $f(p; q) = -f(q; p)$ and $g(p; q) = -g(q; p)$. To discover how these two observations assist in speeding up the velocity calculation, consider the simple case when the grid size is given by $N = 4$. The computation of the $x$-component of the velocity $u_1$ and $u_2$ at grid points 1 and 2, respectively, are calculated by

$$u_1 = f(1; 0) + f(1; 2) + f(1; 3).$$
<table>
<thead>
<tr>
<th>Method</th>
<th>$T_1$ (in sec.)</th>
<th>$T_2$ (in sec.)</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Krasny</td>
<td>10.274</td>
<td>5.398</td>
<td>1.90</td>
</tr>
<tr>
<td>Baker</td>
<td>11.016</td>
<td>5.518</td>
<td>2.00</td>
</tr>
<tr>
<td>Beale’s First</td>
<td>16.283</td>
<td>8.202</td>
<td>1.99</td>
</tr>
<tr>
<td>Beale’s Second</td>
<td>28.02</td>
<td>14.13</td>
<td>1.98</td>
</tr>
</tbody>
</table>

Table 2.6: Timing studies for the fast summation method compared to the previous (naive) sum calculation. The variable $T_1$ is the time needed to perform one integration step in the naive method introduced earlier, and $T_2$ is the time for the fast method.

\[
u_2 = f(2; 0) + f(2; 1) + f(2; 3).
\]

(2.46)

Since $f(1; 2) = -f(2; 1)$, a new algorithm computes $f(1; 2)$ to update $u_1$, and uses the negative of $f(1; 2)$ to update $u_2$ in one step. The steps of this algorithm are summarized in Figure 2.8. We call this algorithm the fast summation algorithm.

For $N = 4$, only three steps are needed (instead of six before) to compute the velocity at all grid points. In general, this method requires $N(N - 1)/2$ computation steps. A speed-up of about two can be expected compared to the earlier algorithm to compute the velocity. This speed-up factor is observed in practice. Timing was performed on a velocity calculation for all points on a 4096-grid using all four smoothed kernels. Results are recorded in Table 2.6.

The efficiency of the fast summation algorithm comes from two sources. The elimination of redundant calculations is a big factor in the speed-up since each calculation requires the sinh, cosh, sine, and cosine evaluations. Secondly, updating $u_1$ and $u_2$ at the same time capitalizes on spatial cache locality, because $u_1$ and $u_2$ are likely to be in the cache at the same time.
2.4.2 The Zig-Zag Method

Recall that the computation of the velocity at parameter point $p$ is a sum of contributions from all points $q \neq p$. The next algorithm to be studied is motivated by Figure 2.9. It shows the steps in the calculation that must be done to obtain the velocity at the point $p = 0$.

At the end of the velocity calculation for the sheet marker at $p = 0$, the data near the end of the array (i.e., the 6th, 7th, and 8th elements) are already in the cache. These elements should be in the first level cache, and if they are not, then they should be in the second level cache (refer to Appendix C for a discussion of cache architecture). As a consequence, it makes sense to reorder the sum. The next calculation of the velocity of the marker at $p = 1$ or the calculation of $(u_1, v_1)$ should sum contributions starting with the sixth element and proceed backwards to the first element of the grid. The calculation of $(u_2, v_2)$ starts from the third element.

Figure 2.8: A better algorithm which takes less number of steps.
Calculating $u_0$

Figure 2.9: This figure shows the iteration steps corresponding to the appropriate points on the grid. Grid points at the end of each iteration is likely to be in the cache. This is the fundamental idea behind the zig-zag method.

and heads to the end of the array. By the nature of the loop process, we call this method the **zig-zag** method.

The complete algorithm of the zig-zag method is shown on Figure 2.10. The zig-zag method incorporates the idea of the fast summation method. Surprisingly, this method does not provide any considerable speedup. In Table 2.7, a timing study is conducted for Krasny’s kernel for grid sizes $N = 1024, 2048$ and $4096$. We measure the time in seconds needed to perform the velocity calculation for every grid point using the fast summation and the zig-zag method. There is almost no improvement which is evident by the fact that the speed-up factor is about one. We believe that processor optimization already rearranges the order of the calculations by the method of **out-of-order execution** [15](page 242) of instructions. Forcing a rearrangement of instruction executions therefore does not gain any efficiency.
Figure 2.10: The zig-zag algorithm showing all the steps for the velocity calculation at each grid point.

<table>
<thead>
<tr>
<th>$N$</th>
<th>Fast Krasny (sec.)</th>
<th>Zig-Zag Method (sec.)</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>1.342</td>
<td>1.321</td>
<td>1.02</td>
</tr>
<tr>
<td>2048</td>
<td>5.438</td>
<td>5.327</td>
<td>1.02</td>
</tr>
<tr>
<td>4096</td>
<td>21.501</td>
<td>21.290</td>
<td>1.01</td>
</tr>
</tbody>
</table>

Table 2.7: Timing studies of the zig-zag method compared to the fast summation method for Krasny's kernel over a range of $N$. 

34
2.4.3 The Symmetric Method

In the previous section, we used the symmetry in the equation in an attempt to improve efficiency. We now examine the symmetry in the vortex sheet profile to reduce computational cost.

The symmetry of the vortex sheet location at any time $t$ is hinted at by the initial sheet profile. In Figure 2.11, Krasny's initial condition is replotted along with the points $z(\pi - \xi)$ and $z(\pi + \xi)$ (for $\xi \in [0, \pi]$) shown in circles. The figure indicates a symmetry connection between these two points. Specifically, the symmetry relations are

$$x(\pi + \xi) - \pi = \pi - x(\pi - \xi) \quad \text{and} \quad y(\pi + \xi) = -y(\pi - \xi). \quad (2.47)$$

These relations can be stated in a form to test symmetry. For all $\xi \in [0, \pi]$, symmetry of the sheet holds if

$$x(\pi + \xi) - x(\pi - \xi) - 2\pi = 0 \quad \text{and} \quad y(\pi + \xi) + y(\pi - \xi) = 0. \quad (2.48)$$

Numerically, this is equivalent to testing

$$\left( x_{N/2-q} + x_{N/2+q} - 2\pi, y_{N/2-q} + y_{N/2+q} \right) = (0, 0) \quad \text{for} \quad q = 1, \ldots, N/2 - 1. \quad (2.49)$$

The sheet's symmetry can be analyzed by examining the following quantities at time $t$

$$
e_x(t) = \max_{1 \le q < N} -\log_{10} |x_{N/2-q}(t) + x_{N/2+q}(t) - 2\pi|.
$$

$$
e_y(t) = \max_{1 \le q < N} -\log_{10} |y_{N/2-q}(t) + y_{N/2+q}(t)|.
$$

$$
e(t) = \max(e_x(t), e_y(t)).
$$

In Figure 2.12, we plot $e(t)$ versus time $t$ which we normalize to range from 0 to 1. The calculation is done for all four smoothed kernels with $\delta = 0.1$ up to a final time $T = 2\pi$. Both the spatial and temporal grid size are set at $N = 2048$ points.
The figure indicates a gradual decay of symmetry in the vortex sheet profile as a function of time. It is more prevalent in Beale’s second kernel. The slow decay of symmetry is a consequence of round-off error accumulation. Since Beale’s second kernel involves more computations than the other three kernels, accumulation of round-off error is faster for Beale’s second kernel.

The use of the sheet’s symmetry to evaluate the velocity leads to the **symmetric method**. We demonstrate the use of the symmetric method with Krasny’s kernel. This technique can be applied to the other three kernels. The velocity calculation at parameter point \( p \) is given by

\[
2N \cdot \tilde{Q}_p = \sum_{q=0}^{N-1} \frac{\sinh(y_p - y_q) + i \sin(x_p - x_q)}{\cosh(y_p - y_q) + \cos(x_p - x_q) + \delta^2}.
\]  

(2.53)

Now think of the sum in equation (2.53) symbolically as

\[
2N \cdot \tilde{Q}_p = \sum_{q=0 \atop q \neq p}^{N-1} K(p, q).
\]  

(2.54)

One can break the sum into two smaller sums to obtain

\[
2N \cdot \tilde{Q}_p = K(p, 0) + K(p, N/2) + \sum_{q=1}^{(N/2)-1} K(p, q) + \sum_{q=N/2}^{N-1} K(p, q)
\]  

(2.55)

Shifting the counter of the second sum gives

\[
2N \cdot \tilde{Q}_p = K(p, 0) + K(p, N/2) + \sum_{q=1}^{(N/2)-1} [K(p, q) + K(p, q + N/2)].
\]  

(2.56)

Equation (2.56) is true for \( p \in \mathbb{Z}[0, N - 1] \). By the symmetry of the sheet, only half of the sheet needs to be considered. The other half can be obtained using the symmetry relation. Thus the discrete sheet parameters \( p \) and \( q \) should lie in the space \( \mathbb{Z}[0, (N/2) - 1] \). Consequently, the term \( K(p, p + N/2) \) must be pulled out of the sum.
in equation (2.56) to maintain the index condition \( q \neq p \). Hence the sum is written as

\[
2N \cdot \vec{Q}_p = K(p, 0) + K(p, N/2) + K(p, p + N/2) + \sum_{i=1}^{(N/2) - 1} [K(p, q) + K(p, q + N/2)].
\]

for \( p \in \mathbb{Z}[0, (N/2) - 1] \). Using the fact that \((x_{N/2}, y_{N/2})\) is the center of the sheet and has coordinate \((\pi, 0)\), the term \(K(p, N/2)\) is

\[
K(p, N/2) = \frac{\sinh y_p + i \sin(x_p - \pi)}{\cosh y_p - \cos(x_p - \pi) + \delta^2} = \frac{\sinh y_p - i \sin x_p}{\cosh y_p + \cos x_p + \delta^2}.
\]

Using the symmetry relation (2.49), the term \(K(p, q + N/2)\) is

\[
K(p, q + N/2) = \frac{\sinh(y_p + y_q) + i \sin(x_p + x_q - 2\pi)}{\cosh(y_p + y_q) - \cos(x_p + x_q - 2\pi) + \delta^2} = \frac{\sinh(y_p + y_q) + i \sin(x_p + x_q)}{\cosh(y_p + y_q) - \cos(x_p + x_q) + \delta^2}.
\]

The velocity calculation at the point \( p \) is now a sum of \((N/2) - 1\) terms

\[
\vec{Q}_p = \frac{\sinh y_p + i \sin x_p}{\cosh y_p - \cos x_p + \delta^2} + \frac{\sinh y_p - i \sin x_p}{\cosh y_p + \cos x_p + \delta^2} + \frac{\sinh 2y_p + i \sin 2x_p}{\cosh 2y_p - \cos 2x_p + \delta^2} + \sum_{i=1}^{(N/2) - 1} \left[ \frac{\sinh(y_p + y_q) + i \sin(x_p + x_q)}{\cosh(y_p + y_q) - \cos(x_p + x_q) + \delta^2} + \frac{\sinh(y_p - y_q) + i \sin(x_p - x_q)}{\cosh(y_p - y_q) - \cos(x_p - x_q) + \delta^2} \right].
\]

Even though the symmetric form of Krasny’s kernel now has more terms, the sum is only over \((N/2) - 1\) summands. This is important since the execution of multiple instructions within one loop iteration can be programmed to execute on a multi-stage
Figure 2.11: Showing the symmetry in the vortex sheet position

pipeline. A speed-up factor of two can be expected. Table 2.8 shows execution time of the symmetric method compared to the fast summation method to evaluate the velocity at every grid point as a function of grid size $N$. A speed-up factor of two is observed.

2.4.4 The Lump Method

There is still a large degree of cache incoherence in the sum calculation even with the use of the symmetric method. Here we are concerned with the case when the grid size $N$ is larger than the size of the cache (which happens most of the time). To provide faster code execution, modifications are made to the symmetric method to sum terms in groups. It uses the idea of the zig-zag method.
Figure 2.12: Symmetry testing for all four smoothed kernels.

<table>
<thead>
<tr>
<th>$N$</th>
<th>Fast Krasny (sec.)</th>
<th>Symmetric Version (sec.)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.331</td>
<td>0.17</td>
<td>1.95</td>
</tr>
<tr>
<td>1024</td>
<td>1.342</td>
<td>0.661</td>
<td>2.03</td>
</tr>
<tr>
<td>2048</td>
<td>5.408</td>
<td>2.684</td>
<td>2.01</td>
</tr>
</tbody>
</table>

Table 2.8: Timing studies of the symmetric method compared to the fast summation method for Krasny’s kernel with $\delta = 0.1$ over a range of $N$. 
Figure 2.13: Dividing grid points into groups.

Recall that the velocity calculation at any grid point $p$ needs contribution from all grid points $q$, for $q$ not equal to $p$. At the end of each iteration in $q$, the level 1 cache (and most likely level 2 as well) will contain the data near the end of the grid. The next iteration in $p$ begins with $q = 0$ and thus needs data at the beginning of the grid which is not likely to be in either of the two caches. Cache thrashing occurs and data must be fetched from local memory, a timewise expensive process.

To make better use of the cache, we apply the same philosophy that the FFTW software library uses. FFTW first analyzes cache sizes of the computer before it performs any calculation. Then it arranges the arithmetic order of array elements accordingly. In our code, the user must enter cache parameters. There is free software available, and it is very easy to write a code to analyze cache sizes using array striding. Once cache parameters are entered into the code, the code will arrange the order of array arithmetic in the following way.

For simplicity of discussion, suppose the grid size $N$ is 16, and cache parameter $M$ is equal to 4. The code will divide the parameter space into $M$ groups, each of size $N/M$ as illustrated in the Figure 2.13. Group $k$ contains the array element $k(N/M)$ to the array element $k(N/M) - 1$. 
Figure 2.14: Better algorithm.
<table>
<thead>
<tr>
<th>$N$</th>
<th>Symmetric Krasny (sec.)</th>
<th>Lump (sec.)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>512</td>
<td>0.17</td>
<td>0.141</td>
<td>1.201</td>
</tr>
<tr>
<td>1024</td>
<td>0.661</td>
<td>0.541</td>
<td>1.221</td>
</tr>
<tr>
<td>2048</td>
<td>2.684</td>
<td>2.230</td>
<td>1.204</td>
</tr>
</tbody>
</table>

Table 2.9: Timing comparison between the symmetric and the lump method using Krasny's kernel.

The calculation begins with group 0 data points calculating their velocities using contributions from other points within group 0. We say that group 0 is working with itself. Next, group 0 works with group 2, and then group 3. Now group 3 works with itself. Then it works with group 2, and then group 1. Note that group 3 does not need to work with group 0. Group 1 takes over and works with itself. Group 1 then works with group 2, but does not need to work with group 3. Finally, group 2 works with group 2. We illustrate the procedure in the Figure 2.4.

Analysis of memory access still reveals cache incoherence, but it is smaller than the naive method stated at the start of the section. In fact, we obtain a speed-up factor of approximately 1.2 as observed in the timing study in Table 2.9. A better speed-up can be achieved by minimizing the overhead involved in administering the zig-zag loop.

2.4.5 Parallelization

Introduction

There is only so much that be done to speed-up the serial execution time. For a large grid, anything above 4096 points, execution time on a serial computer using our most sophisticated sum rearrangement algorithm still takes a long time. There is a
need to implement the code on a parallel machine to divide the work among multiple processors.

Our parallel code is written in C++ using MPI (Message Passing Interface) binding. The code is executed on a 64-processor AMD Duron 650 megahertz Beowulf cluster. Processors are arranged in a hypercube topology. Interprocessor communication is implemented in hardware using modern-day cut-through routing on a fast ethernet network with maximum transfer speed of 100 MBits per second.

The use of the Ethernet network leads to a communication system that is equivalent to systems of workstations connected by ethernet cables. There is no central switching mechanism. Communication is controlled by the ethernet technique (CSMA/CD) Carrier Sense Multiple Access with Collision Detection [20].

Many cluster systems are connected by the Myrinet interface. Myrinet can support bisection and crossbar switches (up to 16). A typical Myrinet interface can support full duplex communication lines with transfer speed up to 2 gigabits per second. But the cost of the Myrinet technology is high relative to the ethernet technology.

**Parallel Code Description**

For ease of discussion, suppose the grid size is given by \( N = 16 \), and the number of processors \( N_p \) is 4. Due to the slow interprocessor communication network of the ethernet system, the goal of our parallel code is to minimize communication. With faster communication network such as the Myrinet, a better scheme with more communication steps can be implemented to reduce the number of calculations at each processor.

The numerical implementation of the ODE solver can be programmed to execute in complete parallelism. It is the velocity calculation that needs data exchange between
processors. Recall that the velocity at grid point \( p \) is calculated using data from points \( q \) not equal to \( p \). Thus each processor needs the data from all other processors.

At time step \( t_k \), every processor broadcasts its local data (4 array elements) to the root processor 0. Processor 0 then combines the data received from each processor into a complete package of 16 data points, and re-broadcasts it back to all the processors. Each processor then uses these 16 data points to compute the velocity for the 4 local data points. The process described is shown in Figure 2.15.

Note that there are redundant calculations being performed at each processor. To eliminate redundant calculations, each processor must broadcast some of its results to all other processors. This technique only works well on a fast communication network. Without this extra communication load, each processor must execute \( N(N/N_p) \) calculations. As communication cost is much larger than processor execution of instructions, this is the trade off that we are willing to take. The velocity evaluation uses the symmetric methodology discussed earlier.

**Timing Results**

Since we are interested in trading interprocessor communication time for instruction execution time, it makes sense to study the quantity \( \gamma \) defined by

\[
\gamma = \frac{\text{Computation Time}}{\text{Communication Time}}. \tag{2.63}
\]

Ideally, \( \gamma \) should be large because a large portion of the total program execution time should not be dedicated to passing data from one processor to another. Since the subroutine that calculates the velocity at each grid point uses the communication network, the study of the variable \( \gamma \) is restricted to this subroutine. The computation time in the variable \( \gamma \) only measures the time to evaluate the velocity. It does not
Figure 2.15: How the parallel code handles communication to reduce the number of communication steps while making sure that the number of computation steps is large.
Figure 2.16: This figure shows the plot of the quantity $\gamma$ defined above.

take into account the execution time of the ODE solver. For timing, we use the MPI commands `MPI.Wtime()` and `MPI.Barrier()` (see [8]). Due to fluctuation in timing, timing results are averages of results from 10 runs.

In Figure 2.16, we plot the variable $\gamma$ for different number of processors $N_p$. The plot suggests if a large number of processors is used, then to maintain a large $\gamma$, the grid size $N$ should be large also.

Next, we measure the speed-up of our parallel code. The serial code is a genuine serial code written specifically to run on one processor. It is not the parallel code
running on one processor. The velocity calculation of the serial code uses the symmetric method. We use the \texttt{time.h} library to measure the running time of the serial code. We study the speed-up [11] of the computation. We call $T_p(N)$ be the total execution time on a parallel machine to solve the vortex sheet problem on an $N$-point grid, and $T_{\text{Serial}}(N)$ be the total execution time on a serial machine. The speed-up factor $S_p(N)$ is given by

$$S_p(N) = \frac{T_{\text{Serial}}(N)}{T_p(N)}.$$  

(2.64)

Figure 2.17 plots the speed-up factor $S_p(N)$ as functions of $N$ and $p$.

2.5 The Bracket Technique

As the smoothing parameter $\delta$ tends to zero, a larger and larger grid size $N$ is needed to both resolve the sheet profile, and to evaluate the velocity at each grid point accurately. The number $N$ can be used to characterize both the spatial grid and the temporal grid. To differentiate between these two grid sizes, we use the notation $N_s$ and $N_t$ to denote the spatial and temporal grid sizes, respectively.

The accuracy of the vortex sheet profile boils down to the following question. For a given $\delta$, what are the sizes of $N_s$ and $N_t$ that guarantee an accuracy of a sheet profile to a certain number of digits? The answer to this question is the bracket technique. Before discussing this technique, let us define the discrepancy.

Given two sequences of complex numbers $u = \{u_0, \ldots, u_{N-1}\}$ and $v = \{v_0, \ldots, v_{N-1}\}$, we define the discrepancy between $u$ and $v$ by

$$\text{discrepancy}(u, v) = \max \{ |u_i - v_i| : 0 \leq i < N \}.$$  

(2.65)
Figure 2.17: Calculation of the speedup.
Furthermore, if the sequence $w$ has length $2N$, then we define the discrepancy between the sequence $u$ (which has length $N$) and sequence $w$ by

$$\text{discrepancy}(u, w) = \max \{ |u_i - w_i| : 0 \leq i < N \}. \quad (2.66)$$

Given a fixed tolerance for the accuracy $\epsilon$, the idea of the bracket technique is the following. Pick some smoothing parameter $\delta_1$, choose some $N$ (as a power of 2) reasonably sized, and set $N_x = N_t = N$. Now run the code for three cases: (1) $(N_x, N_t) = (N, N)$ to obtained a vortex sheet profile (as a sequence of complex number) $x_n$. (2) $(N_x, N_t) = (2N, N)$ to obtain $x_t$ of length $2N$, and (3) $(N_x, N_t) = (N, 2N)$ to obtain $x_t$ of length $N$. We define the spatial and temporal errors by

$$\text{Spatial Error } E_{\text{space}}(\delta_1) = \text{discrepancy}(x_n, x_t). \quad (2.67)$$

$$\text{Temporal Error } E_{\text{time}}(\delta_1) = \text{discrepancy}(x_n, x_t). \quad (2.68)$$

If $E_{\text{space}}(\delta_1)$ and $E_{\text{time}}(\delta_1)$ are both less than $\epsilon$, then we know that for all $\delta \geq \delta_1$, a numerical code can be executed with $N$ spatial and temporal points to obtain an accuracy of at least $\epsilon$. If either $E_{\text{space}}(\delta)$ or $E_{\text{time}}(\delta_1)$ is greater than $\epsilon$, then the calculation repeats the procedure with a new $\delta_1$ larger than the previously chosen $\delta_1$.

The bracket tables for Krasny's, Baker's, and Beale's kernels are shown in Tables 2.10 to 2.12 with an accuracy tolerance of 5 digits. Due to the remarkable similarities in results from Beale's first and second kernels, we only display the one bracket table to represent results for both of Beale's kernels. For a fixed $\delta$, especially when it is small, the value of $N$ needed for Krasny's kernel is less than that of Baker's, and the latter is less than that for Beale's kernel. More importantly, the smaller $\delta$ becomes, the more difficult it is to obtain a profile with 5 digits of accuracy. For 9 digits of
Table 2.10: Bracket table for Krasny's kernel when $\epsilon = 10^{-5}$. Although the result for 4096 points doesn't satisfy the criterion, we include the result to show how difficult it is to generate an accurate profile at small $\delta$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>0.1</td>
<td>$8.69 \times 10^{-7}$</td>
<td>$8.73 \times 10^{-6}$</td>
</tr>
<tr>
<td>512</td>
<td>0.07</td>
<td>$1.65 \times 10^{-6}$</td>
<td>$4.51 \times 10^{-6}$</td>
</tr>
<tr>
<td>1024</td>
<td>0.05</td>
<td>$1.47 \times 10^{-7}$</td>
<td>$1.82 \times 10^{-6}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.042</td>
<td>$4.88 \times 10^{-7}$</td>
<td>$8.43 \times 10^{-7}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.039</td>
<td>$3.10 \times 10^{-6}$</td>
<td>$7.87 \times 10^{-6}$</td>
</tr>
<tr>
<td>8192</td>
<td>0.0355</td>
<td>$9.68 \times 10^{-6}$</td>
<td>$2.29 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

Table 2.11: Bracket table for Baker's kernel with $\epsilon = 10^{-5}$. Although the result for 4096 points doesn't satisfy the criterion, we include the result to show how quickly the resolution requirements increase with decreasing $\delta$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>0.2</td>
<td>$1.20 \times 10^{-11}$</td>
<td>$5.93 \times 10^{-7}$</td>
</tr>
<tr>
<td>512</td>
<td>0.095</td>
<td>$4.15 \times 10^{-6}$</td>
<td>$6.10 \times 10^{-6}$</td>
</tr>
<tr>
<td>1024</td>
<td>0.07</td>
<td>$3.14 \times 10^{-6}$</td>
<td>$1.97 \times 10^{-6}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.055</td>
<td>$4.21 \times 10^{-6}$</td>
<td>$6.36 \times 10^{-6}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.051</td>
<td>$3.67 \times 10^{-6}$</td>
<td>$1.14 \times 10^{-5}$</td>
</tr>
<tr>
<td>8092</td>
<td>0.0493</td>
<td>$9.0412 \times 10^{-6}$</td>
<td>$3.3795 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

Table 2.12: Bracket table for Beale's kernel with $\epsilon = 10^{-5}$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>0.3</td>
<td>$3.35 \times 10^{-12}$</td>
<td>$1.54 \times 10^{-7}$</td>
</tr>
<tr>
<td>512</td>
<td>0.2</td>
<td>$2.63 \times 10^{-11}$</td>
<td>$1.54 \times 10^{-7}$</td>
</tr>
<tr>
<td>1024</td>
<td>0.09</td>
<td>$1.91 \times 10^{-9}$</td>
<td>$3.86 \times 10^{-7}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.075</td>
<td>$1.96 \times 10^{-7}$</td>
<td>$5.16 \times 10^{-7}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.070</td>
<td>$4.24 \times 10^{-6}$</td>
<td>$1.20 \times 10^{-6}$</td>
</tr>
<tr>
<td>8192</td>
<td>0.067</td>
<td>$8.43 \times 10^{-7}$</td>
<td>$1.36 \times 10^{-6}$</td>
</tr>
</tbody>
</table>

50
Table 2.13: Bracket table for Krasny's kernel with $\epsilon = 10^{-8}$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>0.15</td>
<td>$1.14 \times 10^{-9}$</td>
<td>$1.279 \times 10^{-13}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.07</td>
<td>$6.544 \times 10^{-11}$</td>
<td>$6.335 \times 10^{-9}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.055</td>
<td>$5.194 \times 10^{-9}$</td>
<td>$5.862 \times 10^{-9}$</td>
</tr>
</tbody>
</table>

Table 2.14: Bracket table for Baker's kernel with $\epsilon = 10^{-8}$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>0.15</td>
<td>$1.221 \times 10^{-12}$</td>
<td>$8.396 \times 10^{-9}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.09</td>
<td>$9.048 \times 10^{-9}$</td>
<td>$2.873 \times 10^{-8}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.055</td>
<td>$1.621 \times 10^{-9}$</td>
<td>$4.496 \times 10^{-9}$</td>
</tr>
</tbody>
</table>

Table 2.15: Bracket table for Beale's kernel with $\epsilon = 10^{-8}$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\delta$</th>
<th>$E_{space}$</th>
<th>$E_{temporal}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>0.20</td>
<td>$4.193 \times 10^{-11}$</td>
<td>$4.522 \times 10^{-9}$</td>
</tr>
<tr>
<td>2048</td>
<td>0.13</td>
<td>$3.476 \times 10^{-11}$</td>
<td>$2.924 \times 10^{-9}$</td>
</tr>
<tr>
<td>4096</td>
<td>0.095</td>
<td>$2.054 \times 10^{-9}$</td>
<td>$9.138 \times 10^{-9}$</td>
</tr>
</tbody>
</table>
accuracy, as compared to the case of 5 digits of accuracy. Tables 2.13 to 2.15 show that a much larger $N$ is needed at a fixed $\delta$.

Another method to estimate error is to keep track of an energy term. At every time step, Krasny [10] calculates the Hamiltonian, and uses it to adaptively increase grid sizes. If the Hamiltonian is not within some prescribed tolerance, either the spatial or the temporal grid must be refined. Our bracket technique controls error in a stricter manner. Error is controlled pointwise.

2.6 Other Improvements

A rapid change of a function in a small region creates difficulty for accurate numerical evaluation of the integral. In Figure 2.18, we plot the $x$- and $y$-components of Krasny’s kernel as a function of $q$ for a fixed $p$. There are small regions for which both $x$- and $y$-components of Krasny’s kernel change very rapidly.

The $x$ and $y$-components of Krasny’s kernel are plotted on a uniform grid in Figure 2.18. Note that there are only a small number of points to represent those portions of the function which change rapidly. Any numerical integration technique to integrate such functions would not be highly accurate.

Refining the grid resolves the region in question. But most of the points are used to resolve places where the function does not change rapidly. Since decreasing $\delta$ will make the region of rapid change smaller, one can expect increasing computational complexity with decreasing $\delta$. This phenomena was hinted at earlier by examining the bracket tables for all four smoothed kernels.

Instead of searching for faster algorithms, why not try to evaluate the integral more accurately. In this section, we examine four techniques: (1) the filter technique.
Figure 2.18: How points are actually clustered about a region that the function changes rapidly.

(2) the subtracted kernel method. (3) Beale's third-order correction. and (3) the mapping technique.

2.6.1 The Spectrum of the vortex sheet position

Krasny [9] (and many authors since) have applied a spectral filter to suppress the growth of accumulated round-off errors that enter the time-dependent computation with each time step. The idea is to examine the spectrum of the sheet location. For a sheet parameterized by \( N \) markers, its \( x \)-profile and \( y \)-profile, at any fixed time, can
be represented by a truncated Fourier series

\[
x(\xi) - \xi = \frac{a_0}{2} + \sum_{k=1}^{N/2-1} a_k \cos k\xi + \frac{a_{N/2}}{2} \cos \frac{N}{2}\xi + \sum_{k=1}^{N/2-1} b_k \sin k\xi, \quad (2.69)
\]

\[
y(\xi) = \frac{a_0}{2} + \sum_{k=1}^{N/2-1} \alpha_k \cos k\xi + \frac{\alpha_{N/2}}{2} \cos \frac{N}{2}\xi + \sum_{k=1}^{N/2-1} \beta_k \sin k\xi. \quad (2.70)
\]

First, we examine the evolution of the spectrum of the sheet's position in time. It is computed without the filtering technique. We use Beale's first kernel with a spatial and temporal grid size of \( N = 1024, \delta = 0.02, \) and a final time of \( T = 2\pi. \) For this \( \delta, \) referring back to the bracket tables, the grid size \( N = 1024 \) is not large enough to resolve the vortex sheet location to five digits of accuracy. In fact, the ratio \( \delta/h = 3.3 \) indicates that only about 3 grid points are used to resolve one vortex blob. For this reason, we expect to observe a lack of resolution as time progresses.

Since the initial condition contains only one sine mode for both the \( x \) and \( y \)-profiles, the sheet should contain only a small number of sine modes at the beginning of the evolution. The cosine spectrum should be near round-off error values for small time calculations because the initial conditions contain no cosine modes.

We examine the base-10 logarithm of the spectrum, and set \( \log_{10}(0) = -19 \) to prevent excessively large values resulting from applying the logarithm function. Figure 2.19 shows the spectrum of the sheet after the first 10 time steps. The level of round-off error is clear. At time corresponding to 400 time steps, the cosine spectrum is still in the round-off error regions, but the sine spectrum is beginning to develop higher modes, as shown in Figure 2.20. In Figure 2.21, this effect is more pronounced after 600 time steps.

Figure 2.20 shows indication that the cosine spectrum is starting to develop. At time equaling 900 time steps, the cosine spectrum has modes of order \( 10^{-11}. \) At time
step 1010, the sheet's spectrum is shown in Figure 2.23. There the sine spectrum's highest mode is of order $10^{-11}$, which is no longer in the round-off error regions. The sheet position therefore can no longer be resolved completely.

The fact that the highest modes of the sine spectrum fails to be in round-off error values indicates a lack of resolution. The reader may recall that the modified vortex sheet position $(x(\xi) - \xi, y(\xi))$ is an odd function with $\xi$. This means that the cosine spectrum should be all zeros. The development of the cosine spectrum beyond round-off error values indicates accumulation of round-off errors.

A remedy to these two symptoms is a filtering method called the adaptive point-insertion technique used by several authors, for example, Shelley, Hou, Lowengrub, Shelley, Baker, and Nachbin. Whenever the amplitudes of the highest modes rise above round-off error, the number of spatial points used in the calculation is doubled so that the sheet position is again well-resolved. Whenever the cosine spectrum elements are above some round-off level tolerance, say $10^{-13}$, suppress them to zero.

The idea of adaptive point-insertion technique is certainly useful because the number of points needed to resolve the sheet's position completely is not uniform with time. First we perform a calculation with $\delta = 0.07$ on a spatial and temporal grid size of $N = 2048$ points without the adaptive point-insertion technique (or filtering). At these resolutions, we are confident that the sheet's position is accurate up to seven digits. Secondly, the same calculation is performed with the application of the filter at a level of $10^{-13}$ starting with only 64 points. With the filtering technique, Table 2.16 demonstrates that a small grid is enough to resolve the sheet location for many time steps. This is certainly good news.
Figure 2.19: Spectrum of the position at $t = 10\Delta t$.

<table>
<thead>
<tr>
<th>Time Step Range</th>
<th>Spatial Grid Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-783</td>
<td>64</td>
</tr>
<tr>
<td>784-1089</td>
<td>128</td>
</tr>
<tr>
<td>1090-1248</td>
<td>256</td>
</tr>
<tr>
<td>1249-1483</td>
<td>512</td>
</tr>
<tr>
<td>1484-2007</td>
<td>1024</td>
</tr>
<tr>
<td>2008-2048</td>
<td>2048</td>
</tr>
</tbody>
</table>

Table 2.16: Filtering grid size as a function of time step for a calculation using the adaptive point-insertion technique.
Figure 2.20: Spectrum of the position at $t = 400\Delta t$. 
Figure 2.21: Spectrum of the position at $t = 600\Delta t$. 
Figure 2.22: Spectrum of the position at $t = 900\Delta t$. 
Figure 2.23: Spectrum of the position at $t = 1010 \Delta t$. 
Figure 2.24: Filtering Error.
Figure 2.25: Filtering error when an initial grid size of 64 is applied.
Figure 2.26: Filtering error when an initial grid size of 128 is applied.
Is there a tradeoff with using a 64-point grid for the first 783 time steps? Surely 64 points are enough to resolve the vortex sheet location for the first 86 time steps, but are 64 points large enough a resolution to properly evaluate the integral to obtain the instantaneous velocity? Certainly this is not the case. Figure 2.24 shows the negative of the base-10 logarithm of the discrepancy between the vortex sheet position generated by the filtering method starting with 64 points and the sheet generated without the filter on a uniform 2048-point grid for each time step. We observe a discrepancy on the order of $10^{-3}$ for the first 87 time steps.

As the grid is refined with time, does the initial error committed by using only 64 points to evaluate the integral decay with time? Convincing evidence is illustrated in Figure 2.25 which shows the difference between the sheet's position generated by a filtering technique, and one that is generated by a nonfiltering technique using uniform 2048 points at a final time of $T = 2\pi$. The initial error in the integral calculation does not decrease in time. In fact, it worsens to an error of order $10^{-4}$. Starting the adaptive point-insertion technique with 128 points gives an improvement in accuracy as shown in Figure 2.26, but the initial error in the integral calculation is still carried forward in time.

We have now three vortex sheets generated by three different cases: (1) uniform grid of 2048 points without filtering, (2) filtering with tolerance $10^{-13}$ starting with 64 points, and (3) filtering with the same tolerance starting with 128 points. The vortex sheet locations are graphically identical in all three cases since a discrepancy of order $10^{-4}$ cannot be detected by the human eyes.

The fact that an initial error from the velocity calculation is carried forward in time creates doubts regarding the application of the filter method for cases when $\delta$ is
very small. But not all is lost. Filtering does give a fast method to get a rough idea of what the sheet's location is like without doing a more accurate calculation. If we are willing to live with five or six digits of accuracy, then using a filtering technique starting with 128 points is satisfactory.

2.6.2 The Subtracted Kernel

Summary

The exact periodic kernel $K_{\pi, 2\pi}(\xi, \zeta)$ has a singularity as $\zeta \to \xi$. The integral which gives the conjugate of the velocity $\bar{Q}(\xi)$ at any point $\xi$

$$\bar{Q}(\xi) = -\frac{1}{4\pi} \int_0^{2\pi} K_{\pi, 2\pi}(\xi, \zeta) \, d\zeta$$

must be viewed as a principal-valued integral. The idea of the subtracted kernel is to find some function $\phi(\xi, \zeta)$ such that

$$\bar{Q}(\xi) = -\frac{1}{4\pi} \int_0^{2\pi} [K_{\pi}(\xi, \zeta) - \phi(\xi, \zeta)] \, d\zeta$$

where the integrand $[K_{\pi}(\xi, \zeta) - \phi(\xi, \zeta)]$ no longer possesses a singularity as $\zeta \to \xi$.

This technique is called the subtraction technique because the singularity is removed from the kernel. There are two ways proceed. One can apply the subtraction to the exact kernel $K_{\pi}$, derive its periodic version, and smooth it by a cut-off function. Alternatively, the subtraction can be done to Beale's second kernel. These two methods do not produce identical results. The first method results in irregular motion. The second method does not.
Deriving the Subtracted Kernel, Method 1

Recall the instantaneous velocity at each vortex marker point with parameter $\xi$ is given by

$$\tilde{Q}(\xi) = u(\xi) - iv(\xi) = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \frac{\gamma'(\zeta)}{z(\xi) - z(\zeta)} \, d\zeta. \quad (2.73)$$

Following an idea suggested by Baker & Beale [1], the integrand can be recasted into the following form ($\gamma = 1$)

$$2\pi i \tilde{Q}(\xi) = \int_{-\infty}^{\infty} \left[ 1 - \frac{z'(\zeta)}{z'(\xi)} \right] \cdot \frac{d\zeta}{z(\xi) - z(\zeta)} \quad (2.74)$$

$$= \int_{-\infty}^{\infty} \left[ 1 - \frac{z'(\zeta)}{z'(\xi)} \right] \cdot \frac{d\zeta}{z(\xi) - z(\zeta)} + \frac{1}{z'(\xi)} \int_{-\infty}^{\infty} \frac{z'(\zeta)}{z(\xi) - z(\zeta)} d\zeta. \quad (2.75)$$

In a $2\pi$-periodic geometry, the second integral

$$\frac{1}{z'(\xi)} \int_{-\infty}^{\infty} \frac{z'(\zeta)}{z(\xi) - z(\zeta)} = 0. \quad (2.76)$$

As $\zeta \to \xi$, the remaining term

$$\left[ 1 - \frac{z'(\zeta)}{z'(\xi)} \right] \cdot \frac{1}{z(\xi) - z(\zeta)} = \frac{1}{z'(\xi)} \cdot \left[ \frac{z'(\xi) - z'(\zeta)}{z(\xi) - z(\zeta)} \right] \quad (2.77)$$

has no singularity. Now the task is to obtain the periodic version of equation (2.77), and then smooth the periodic version. These two steps are similar to the process by which Beale’s first kernel was constructed. Let’s tackle the periodic version. Writing the kernel as

$$K_{Sub, N} = \left[ 1 - \frac{z'(\zeta)}{z'(\xi)} \right] \cdot \frac{1}{z(\xi) - z(\zeta)} \quad (2.78)$$

and in light of the fact that $z'(\xi + 2\pi n) = z'(\xi)$, it is not too difficult to see that the periodic version of $K_{Sub, N}$, which we now denote as $K_{Sub, P}$, is just given by

$$K_{Sub, P} = \left[ 1 - \frac{z'(\zeta)}{z'(\xi)} \right] \cdot K_{0.2\pi} \quad (2.79)$$
where \( K_{0,2\pi} \) was defined earlier to be the periodic version of the exact kernel. Let’s now use the short-hand notations \((x_\xi, y_\xi) = (x(\xi), y(\xi)), (x'_\xi, y'_\xi) = (x'(\xi), y'(\xi))\), and \(D = (x'_\xi)^2 + (y'_\xi)^2\). Equation (2.79) becomes

\[
K_{Sub,P} = \left[ 1 - \frac{(x'_\xi + iy'_\xi)(x'_\xi - iy'_\xi)}{D} \right] \frac{\sinh (y_\xi - y_\zeta) + i \sin (x_\xi - x_\zeta)}{\cosh (y_\xi - y_\zeta) - \cos (x_\xi - x_\zeta)}
\]

Now write the term in the bracket in real and imaginary parts, and call them \(\alpha\) and \(\beta\) respectively. We obtain

\[
1 - \frac{(x'_\xi + iy'_\xi)(x'_\xi - iy'_\xi)}{D} = \left( 1 - \frac{x'_\xi x'_\xi - y'_\xi y'_\xi}{D} \right) - i \frac{x'_\xi y'_\xi - x'_\xi y'_\xi}{D}
\]  

\[
(2.81)
\]

Multiply \((\alpha - i \beta)\) by \(K_{0,2\pi}\) to get

\[
\text{Re} \ K_{Sub,P} = \frac{\alpha \cdot \sinh (y_\xi - y_\zeta) + \beta \cdot \sin (x_\xi - x_\zeta)}{\cosh (y_\xi - y_\zeta) - \cos (x_\xi - x_\zeta)}
\]

\[
(2.83)
\]

\[
\text{Im} \ K_{Sub,P} = \frac{-\beta \cdot \sinh (y_\xi - y_\zeta) + \alpha \cdot \sin (x_\xi - x_\zeta)}{\cosh (y_\xi - y_\zeta) - \cos (x_\xi - x_\zeta)}
\]

\[
(2.84)
\]

To illustrate typical results, we smooth \(K_{Sub,P}\) as in Beale’s second kernel by multiplying it with a Gaussian profile to obtain

\[
K_{Sub} = K_{Sub,P} \cdot \left( 1 - e^{-2t[\cosh (y_\xi - y_\zeta) - \cos (x_\xi - x_\zeta)]} \delta^2 \right).
\]

\[
(2.85)
\]

We perform calculations with \(\delta = 0.25\), and show that as the grid is refined, the solution does not converge in Figure 2.27. Note that a grid of 1024 points is more than sufficient to generate a profile with six digits of accuracy using Beale’s second kernel without the subtraction technique. With singularity removal, plots in the figures show that a 1024-point grid is still not sufficient. Moreover, since the profile shows some irregularity, it is likely that the kernel obtained by singularity removal does not regularize vortex sheet motion.
Figure 2.27: As the grid is refined, the vortex sheet position generated by the smoothed $K_x$ appears to converge. This profile was generated at $\delta = 0.25$. Slow convergence may be due to irregular motion.
Derivation of the Subtracted Kernel, Method 2

We follow the same idea that was applied in the previous section to obtain a subtracted form of the kernel in the full domain \( \mathbb{R} \). We work now with \( K_{[0,2\pi]} \) over the finite domain \([0,2\pi]\). The idea is to find an \( 2\pi \)-periodic integrand which is a perfect derivative. The starting point is Beale’s simple kernel

\[
K_{\text{Beale}}(\xi, \zeta) = \frac{\sinh \Delta y + i \sin \Delta x}{\cosh \Delta y - \cos \Delta x} \cdot \left[ 1 - e^{-2(\cosh \Delta y - \cos \Delta x) \cdot \delta^2} \right].
\]  
(2.86)

where we use the notation \( \Delta x = x(\xi) - x(\zeta) \), and \( \Delta y = y(\xi) - y(\zeta) \). For simplicity, define

\[
F(\Delta x, \Delta y) = 1 - e^{-2(\cosh \Delta y - \cos \Delta x) \cdot \delta^2}.
\]  
(2.87)

and

\[
S(\xi, \zeta) = \frac{y'(\zeta) \sinh \Delta y + x'(\zeta) \sin \Delta x}{\cosh \Delta y - \cos \Delta x}.
\]  
(2.88)

It can be easily checked that \( S(\xi, \zeta) \cdot F(\Delta x, \Delta y) \) is the \( \zeta \)-derivative of another function. Therefore

\[
-\frac{1}{4\pi} \int_0^{2\pi} S(\xi, \zeta) \cdot F(\Delta x, \Delta y) \, d\zeta = 0.
\]  
(2.89)

For any real constants \( \alpha \) and \( \beta \), it is still the case that

\[
\frac{\partial \xi}{\partial t} = -\frac{1}{4\pi} \int_0^{2\pi} \left[ K_{\text{Beale}}(\xi, \zeta) - (\alpha + i\beta) \cdot S(\xi, \zeta) \cdot F(\Delta x, \Delta y) \right] \, d\zeta.
\]  
(2.90)

Now the goal is to find \( \alpha \) and \( \beta \) so that the integrand has no singularity as \( \zeta \to \xi \). Taylor expansions can be applied to find the leading order of each term in equation (2.90).

Let’s examine the term

\[
K_{[0,2\pi]}(\xi, \zeta) = \frac{\sinh \Delta y + i \sin \Delta x}{\cosh \Delta y - \cos \Delta x}.
\]  
(2.91)

Now perform a Taylor expansion in the limit as \( \zeta \to \xi \). Let us write \( \zeta = \xi + \epsilon \) where \( \epsilon \) is a small number in magnitude. Since \( \Delta x = x(\xi) - x(\zeta) \), we would rather like to
work with $-\Delta x = x(\zeta) - x(\xi)$ which we write in short-hand as $\tilde{x} - x$. We make use of the following Taylor expansions

\[- \sinh \Delta y = \sinh (\tilde{y} - y) = [(\tilde{y} - y) + \frac{1}{3!}(\tilde{y} - y)^3 + \cdots] \quad (2.92)\]

\[= \epsilon \cdot y'(\xi) + \epsilon^2 \cdot \frac{y''(\xi)}{2} + \epsilon^3 \cdot \frac{y'''(\xi)}{6} + O(\epsilon^4) \quad (2.93)\]

\[\cosh \Delta y = \cosh (\tilde{y} - y) = 1 + \frac{1}{2!}(\tilde{y} - y)^2 + \frac{1}{4!}(\tilde{y} - y)^4 + \cdots \quad (2.94)\]

\[= 1 + \epsilon^2 \cdot \frac{y'(\xi)^2}{2} + \epsilon^3 \cdot \frac{y'(\xi)y''(\xi)}{2} + O(\epsilon^4) \quad (2.95)\]

\[\cos \Delta x = \cos(\tilde{x} - x) = 1 - \frac{1}{2!}(\tilde{x} - x)^2 + \frac{1}{4!}(\tilde{x} - x)^4 + \cdots \quad (2.96)\]

\[= 1 - \epsilon^2 \cdot \frac{x'(\xi)^2}{2} - \epsilon^3 \cdot \frac{x'(\xi)x''(\xi)}{2} + O(\epsilon^4) \quad (2.97)\]

\[- \sin \Delta x = \sin(\tilde{x} - x) = [(\tilde{x} - x) - \frac{1}{3!}(\tilde{x} - x)^3 + \cdots] \quad (2.98)\]

\[= \epsilon \cdot x'(\xi) + \epsilon^2 \cdot \frac{x''(\xi)}{2} + \epsilon^3 \cdot \frac{x'''(\xi) - x'(\xi)^3}{6} + O(\epsilon^4). \quad (2.99)\]

To the leading order, we obtain

\[K_{\alpha,2\pi}(\xi, \zeta) = \frac{-\epsilon \cdot [y'(\xi) + ix'(\xi)] + O(\epsilon^2)}{\frac{\epsilon}{2}[x'(\xi)^2 + y'(\xi)^2] + O(\epsilon^3)} \quad (2.100)\]

\[= -\frac{1}{\epsilon} \cdot \frac{2[y'(\xi) + ix'(\xi)]}{x'(\xi)^2 + y'(\xi)^2} + O(1). \quad (2.101)\]

Now we examine the leading order of the integrand that sets up the subtracted kernel

\[S = \frac{y'(\zeta) \sinh \Delta y + x'(\zeta) \sin \Delta x}{\cosh \Delta y - \cos \Delta x}. \quad (2.102)\]

By Taylor expansion, we obtain

\[S = \frac{-\epsilon[y'(\xi)^2 + x'(\xi)^2]}{\frac{\epsilon}{2}[x'(\xi)^2 + y'(\xi)^2] + O(\epsilon^3)} \quad (2.103)\]

\[= -\frac{1}{\epsilon} \cdot 2 + O(1) \quad (2.104)\]

The subtracted kernel form (2.90) becomes

\[K_{\text{Beurling}}(\xi, \zeta) = (\alpha + i\beta) \cdot S(\xi, \zeta) \cdot F(\Delta x, \Delta y) \quad (2.105)\]

\[= \left[ -\frac{1}{\epsilon} \cdot \frac{2[y'(\xi) + ix'(\xi)]}{x'(\xi)^2 + y'(\xi)^2} + \frac{2(\alpha + i\beta)}{\epsilon} \right] \cdot F(\Delta x, \Delta y). \]

70
With the choice

\[ \alpha = \frac{y'(\xi)}{x'(\xi)^2 + y'(\xi)^2} \quad \text{and} \quad J = \frac{x'(\xi)}{x'(\xi)^2 + y'(\xi)^2}, \]

we remove the $1/\epsilon$ behavior in the expression in the brackets in equation (2.105).

Now substitute the values of $\alpha$ and $J$ above back into equation (2.90) to obtain the subtracted kernel. In particular, let us pay attention to the $x$-component. We have

\[
-4\pi u(\xi) = \int_0^{2\pi} \left[ \frac{\sinh \Delta y}{\cosh \Delta y - \cos \Delta x} \left( \frac{\alpha y'(\xi)}{x'(\xi)^2 + y'(\xi)^2} \cdot \frac{y'(\zeta) \sinh \Delta y + x'(\zeta) \sin \Delta x}{\cosh \Delta y - \cos \Delta x} \right) \right] \cdot F(\Delta x, \Delta y) \, d\zeta. \tag{2.107}
\]

Writing $D = x'(\xi)^2 + y'(\xi)^2$. we obtain

\[
-4\pi D \cdot u = \int_0^{2\pi} \left[ \frac{[x'(\xi)^2 + y'(\xi)^2] \sinh \Delta y}{\cosh \Delta y - \cos \Delta x} \left( \frac{\alpha y'(\xi) y'(\zeta) \sinh \Delta y + x'(\zeta) y'(\xi) \sin \Delta x}{\cosh \Delta y - \cos \Delta x} \right) \right] \cdot F(\Delta x, \Delta y) \, d\zeta, \tag{2.108}
\]

\[
= \int_0^{2\pi} \left[ \frac{y'(\xi)[x'(\xi) - x'(\zeta)] \sinh \Delta y}{\cosh \Delta y - \cos \Delta x} \left( \frac{x'(\xi) y'(\xi) \sin \Delta x}{\cosh \Delta y - \cos \Delta x} \right) \right] \cdot F(\Delta x, \Delta y) \, d\zeta. \tag{2.109}
\]

In the same manner, the instantaneous velocity $v$ along the $y$-direction is

\[
-4\pi D \cdot v = \int_0^{2\pi} \left[ \frac{x'(\xi)[x'(\xi) - x'(\zeta)] \sinh \Delta y}{\cosh \Delta y - \cos \Delta x} \right] \cdot F(\Delta x, \Delta y) \, d\zeta. \tag{2.110}
\]

We performed calculations with the same $\delta = 0.25$ as in the previous section where irregular motion was observed. On a grid of only 256 points, the vortex sheet appears to be well-behaved for the form of the subtracted kernel derived by the second method as seen in Figure 2.28.
Figure 2.28: This figure shows no irregular motion associated with the subtracted kernel derived by method 2.

Since numerical solutions obtained from the subtracted kernel derived in this section are well-behaved, the issue of its performance and accuracy become relevant, the topic of the next section.

2.6.3 Beale's Correction

Summary

When $h \ll \delta$, many points are used to resolve one vortex blob, and the accuracy of velocity calculation is spectrally accurate. When $h \gg \delta$, the dominant error is $O(h)$. Baker and Beale [1] provide a correction that improves the error to $O(h^3)$. Results are shown to prove that the correction is indeed third-order. Calculations are made for several vortex sheet profiles to test the effectiveness of the method.
Error of the Subtracted Kernel

The smoothed subtracted kernel has the form

\[ K_{\text{Sub}, P}(\xi, \zeta) \cdot F(\Delta x, \Delta y) \]  

(2.111)

where \( \Delta x = x(\xi) - x(\zeta) \), \( \Delta y = y(\xi) - y(\zeta) \), where the real and imaginary parts of \( K_{\text{Sub}, P} \) are obtained by equations (2.109) and (2.110) respectively. The instantaneous velocity is calculated by

\[ \tilde{Q}(\xi) = -\frac{1}{4\pi} \int_{0}^{2\pi} K_{\text{Sub}, P}(\xi, \zeta) \cdot F(\Delta x, \Delta y) \quad d\zeta. \]  

(2.112)

Numerical evaluation of \( \tilde{Q}(\xi) \) involves replacing the integral by a sum of \((N - 1)\) terms as follows

\[ \tilde{Q}(\xi_i) = -\frac{1}{4\pi} \cdot \frac{2\pi}{N} \sum_{j \neq i}^{N-1} K_{\text{Sub}, P}(\xi_i, \xi_j) \cdot F(\Delta x_j, \Delta y_j) \]  

(2.113)

We are interested in calculating the largest part of the quadrature error \( e^{(h)} \) so it can be added to the sum in equation (2.113) to obtain better accuracy. The error \( e^{(h)} \) is obtained by replacing \( K_{\text{Sub}}(\xi, \zeta) \) by its linear approximation at \( \zeta = \xi \) in equation (2.112).

Let’s first examine the real part of \( K_{\text{Sub}, P} \). It has two terms, and we can write

\[ \text{Re}[K_{\text{Sub}}(\xi, \zeta)] = T_{\xi}^{(1)} + T_{\xi}^{(2)}. \]

To further simplify notation, introduce \( \tau_{\xi}^{(1)} \) and \( \tau_{\xi}^{(2)} \) which are defined by

\[ T_{\xi}^{(1)} + T_{\xi}^{(2)} = \frac{\tau_{\xi}^{(1)} + \tau_{\xi}^{(2)}}{\cosh \Delta y - \cos \Delta x} \cdot F(\Delta x, \Delta y). \]  

(2.114)

With \( h = \zeta - \xi \). Taylor expansions of \( \tau_{\xi}^{(1)} \) and \( \tau_{\xi}^{(2)} \) are given by

\[ \tau_{\xi}^{(1)} = (y'(\xi)[y'(\xi) - y'(\zeta)] + (x'(\xi)^2)) \cosh \Delta y \]  

and

\[ \tau_{\xi}^{(2)} = (x'(\xi)^2)(y'(\xi) - y'(\zeta)) \cosh \Delta y \]  

(2.115)
The addition of these two terms and division by the leading order term of \( \cosh \Delta y - \cos \Delta x \), which is \( h^2[x'(\xi)^2 + y'(\xi)^2] \), gives

\[
E_{\xi} = \frac{1}{4\pi} \frac{y'(\xi) \cdot x'(\xi)^2 - 2y'(\xi)^2y''(\xi) - 3x'(\xi)\cosh(\xi)y'(\xi)}{x'(\xi)^2 + y'(\xi)^2} + O(h). \tag{2.118}
\]

In a similar fashion, we obtain

\[
E_{\xi} = \frac{1}{4\pi} \frac{3x'(\xi)y'(\xi)y''(\xi) + 2y'(\xi)^2x''(\xi)}{x'(\xi)^2 + y'(\xi)^2} + O(h). \tag{2.119}
\]

**Third-Order Correction**

It is not too difficult (but tedious) to show that equations (2.118) and (2.119) can be written as

\[
E_{\xi} = E_{\xi}^{(x)} + iE_{\xi}^{(y)} = \frac{1}{2\pi i} \left[ \frac{1}{2z'(\xi)} \left( \frac{z''(\xi)}{z'(\xi)} + \text{Re} \left\{ \frac{z''(\xi)}{z'(\xi)} \right\} \right) \right] \tag{2.120}
\]

Baker and Beale [1] shows that the largest part of the quadrature error \( e^{(h)} \) with \( \delta/h \) fixed and \( h \to 0 \), occurs when \( K_{s_{ab}}(\xi, \zeta) \) is replaced by its linear approximation. In fact, \( e^{(h)} \) can be shown to be in terms of the previous equation to be

\[
e^{(h)}(\xi) = \frac{1}{2\pi i} \left[ \frac{1}{2z'(\xi)} \left( \frac{z''(\xi)}{z'(\xi)} + \text{Re} \frac{z''(\xi)}{z'(\xi)} \right) \right] h\epsilon^{(0)} + O(h^3) \tag{2.121}
\]

where

\[
h\epsilon^{(0)} = \sqrt{2\pi} \sigma \sum_{n \neq 0} \hat{g}(2\pi n\sigma). \tag{2.122}
\]

The function \( \hat{g} \) is given as \( \hat{g}(k) = -\sqrt{2}e^{-k^2/4}/2 \). The variable \( \sigma \) is given by \( \sigma = h/D\delta \).

After the velocity is calculated at each grid point by using equation (2.113), add the first term of (2.121) to the velocity. The result should give an error of \( O(h^3) \).
To make sure that our code works correctly, numerical results must indicate that the order of the method is indeed three. Since the correction is third-order, the error obtained by comparing the velocity generated by the third-order correction method and by Beale's simple kernel using a large enough grid must have the form

$$E_h = C h^3 + O(h^4). \quad (2.123)$$

Taking the logarithm gives

$$\log_{10} |E_h| \approx \log_{10} |C| + 3 \log_{10} h. \quad (2.124)$$

Now compare the differences of the errors obtained from using two different-sized grids. It is given by

$$\log_{10} |E_h| - \log_{10} |E_{h,2}| = 3 \log_{10} 2 = 0.90. \quad (2.125)$$

Equation (2.125) says that refining the grid by doubling the number of points shifts the number of accurate digits by a fixed amount.

**Presentation of Results**

For an \(N\)-point grid, the accuracy of the method is determined by first evaluating the velocity at each grid point via the subtracted kernel plus the third-order correction. Then evaluate the velocity at these same points, but use a large enough grid to assure an accuracy at round-off error level using Beale's first kernel. Finally, the error is obtained from the discrepancy between the results obtained from the two methods.

Before the numerical integration scheme of the subtracted kernel is performed, a position profile must be specified since the kernel is a function of the position. We apply four profiles to test the accuracy of the third-order correction method. The first
Figure 2.29: The second and third position profiles to be used in testing the accuracy of the subtracted kernels.
profile is the initial condition, the second, third, and fourth profiles are positions of the sheet at later times as calculated by Beale's kernel with $\delta = 0.1$. All four profiles are shown in Figure 2.29. Note that only the first profile is known analytically.

We first show that the method is correct by appealing to the fact that difference in digits of accuracy between a grid of $N$ and $2N$ points is roughly 0.9. Using the first profile, we plot the accuracy for grids of 32, 64, 128, and 256 points using Beale's 3rd-order correction method in Figure 2.30. The bottom plot zooms into a small region of the top graph to show a gap between the curves of approximately 0.9.

Recall that $\delta/h$ gives the number of grid points per blob. If Beale's third-order correction is to work effectively, high accuracy is desired for moderate values of $\delta/h$. Thus it makes sense to study the accuracy as a function of $\delta/h$. As before, the accuracy of a method on an $N$-point grid is measured by the discrepancy of the results obtained from an $N$-point grid and the results obtained from Beale's first kernel with a fine enough grid. The latter results should have an accuracy level on order of round-off errors.

We study the accuracies of both Beale's first kernel and the 3rd-order correction method on an $N$-point grid for a range of $N$. More specifically, we study the negative of the base-10 logarithm of the accuracies. The calculated accuracies are shown in Figures 2.31 to 2.34 for profiles 1 to 4, respectively. The curve in solid is the accuracy obtained from using Beale's first kernel, and the dotted curve, the accuracy obtained from using Beale's 3rd-order method.

For profile 1, the analytic profile, the 3rd-order method works very well. As the vortex sheet position becomes more complicated, first seen in profile 2, $N$ now must be at least 128 for the 3rd-order method to outperform Beale's first kernel. This
Figure 2.30: Debugging the subtraction plus third-order correction method.
Figure 2.31: Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 1.
Figure 2.32: Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 2.
Figure 2.33: Comparing the velocity evaluation of Beale’s first kernel and the subtraction + 3rd-order correction for profile 3.
Figure 2.34: Comparing the velocity evaluation of Beale's first kernel and the subtraction + 3rd-order correction for profile 4.
claim is supported by the graphs obtained for the third and fourth profiles which
have roll-ups.

**Conclusion**

Compared with Beale's first kernel, the subtraction method in conjunction with
Beale's third-correction provides more accuracy for velocity evaluation. The com­
putational complexity of the former is more intense than the latter. Moreover, as
the vortex sheet develops roll-up structures, the grid size $N$ must increase for the
3rd-order correction method to be more accurate than Beale's first kernel. This sug­
gests a time-adaptive procedure to increase $N$ accordingly so that the accuracy of
the 3rd-order method remains fixed. It is presently not clear how $N$ should vary as
a function of time. Further research is needed to develop a time-dependent adaptive
method using Beale's 3rd-order correction method.

**2.6.4 The Mapping Technique**

As observed from Figure 2.18 earlier, most of the data points are not clustered
about the region where the function changes rapidly. This leads us to question
whether the vortex sheet position is uniformly parameterized by the point vortex
markers. In other words, are the markers uniformly distributed along the sheet? The
issue of parameterization is important since ideally we would like to use many markers
to resolve the portion of the sheet that contains the roll-up.

Unfortunately, most of the markers are used to parameterize the part of the vortex
sheet that does not change rapidly. By symmetry, we examine half of the vortex
sheet spiral, and inquire into the distribution of vortex markers along the sheet. We
examine a typical vortex sheet generated by Beale's kernel at $\delta = 0.1$ using 1024
points. Half of the parameter space \([0, \pi]\) is divided into eight equal-sized partitions giving nine equally-spaced points \(\xi_k = k(\pi/8)\) for \(k \in \mathbb{Z}[0,8]\). To study vortex marker distribution along the sheet, we map each point \(\xi_k\) onto the sheet and label it by a star. The result is shown in Figure 2.35.

About half of all the markers parameterize the section of the sheet that is relatively uninteresting. Only \(3/8\) of the total points are used to parameterize the most interest portion, the region of roll-up. We are interested in redistributing the markers along the sheet so more points are used to parameterize the roll-up section. Baker (private communication) suggested a mapping technique. The basic idea is the following. Suppose we need to numerically evaluate

\[
I = \int_0^{2\pi} f(t) \, dt. \tag{2.126}
\]

A change of variable \(t = \phi(\zeta)\) can be applied such that \(\phi(0) = 0\) and \(\phi(2\pi) = 2\pi\). On a uniform grid of the space \([0, 2\pi]\), the change of variable is equivalent to a rearrangement of the grid points. The integral then has the form

\[
I = \int_0^{2\pi} f(\phi(\zeta)) \phi'(\zeta) \, d\zeta. \tag{2.127}
\]

The function \(\phi\) should be chosen so that \(\phi'(\zeta)\) is never zero. More importantly, \(\phi\) should redistribute points such that the important portion of the sheet is parameterized by many markers. How should this be done? Equation (2.126) can be thought of as equation (2.127) with \(\phi(\zeta) = \zeta\). When \(\phi = \zeta\), the grid points are uniformly distributed. If more points are to cluster about a certain region, then \(\phi(\zeta)\) is equal to \(\zeta + \lambda(\zeta)\) for some nonzero smooth function \(\lambda(\zeta)\).

From Figure 2.35, a function \(\phi\) can be created by relocating the markers \(\xi_k\) to \(\xi_i\) according to the wishes of the user. For instance, if the user wants only one-eighth of
Figure 2.35: This figure the distribution of points used to parameterize the sheet.
the total point to parameterize part of the curve that half of the points did before. then \( \phi \) must take \( \pi/8 \) to \( 4\pi/8 \), or \( \phi(\pi/8) = 4\pi/8 \). The user may want another eighth of all the points to parameterize the section of the curve that was before parameterized [\([4\pi/8, 6\pi/8]\)]. The function \( \phi \) must take \( 2\pi/8 \) to \( 6\pi/8 \), or \( \phi(2\pi/8) = 6\pi/8 \). In the same manner, the user can specify how the rest of the markers \( \xi_k \) maps onto \([0, \pi]\) with the condition that \( \phi(\pi) = \pi \). Once \( \phi \) is defined on half of the parameter space \([0, \pi]\), it can be extended to a function defined on \([0, 2\pi]\) by insisting that \( \phi(\zeta) = \zeta + \lambda(\zeta) \) and \( \lambda \) is an odd \( 2\pi \)-periodic function. We completed the mapping \( \phi \) with the resulting function \( \lambda \) given by

\[
\lambda(\zeta) = \begin{cases} 
3.5\zeta & 0 \leq \zeta < \pi/8 \\
7\pi/16 & \pi/8 \leq \zeta < \pi/4 \\
-\frac{7}{12}(\zeta - \pi) & \pi/4 \leq \zeta < 7\pi/4 \\
-\frac{7\pi}{16} & 7\pi/4 \leq \zeta < 15\pi/8 \\
3.5(\zeta - 2\pi) & 15\pi/8 \leq \zeta \leq 2\pi 
\end{cases}.
\] (2.128)

The function \( \phi \) just created is not differentiable. However, \( \phi \) and therefore the function \( \lambda \), can be approximated by a truncated sine series \( \lambda_{app}(\zeta) \). In Figure 2.36, the first plot shows the construction of \( \phi \) by this technique. We show the plot of \( \lambda \) in the second figure, and the plot of \( \lambda_{app}(\zeta) \) in the third. The fourth plot shows a graph of \( \phi'(t) \) which is nonzero for all \( \zeta \). The function \( \lambda_{app}(\zeta) \) is calculated using 5 sine modes to be

\[
\lambda_{app}(\zeta) = 1.12 \sin \zeta + 0.49 \sin 2\zeta + 0.26 \sin 3\zeta + 0.14 \sin 4\zeta + 0.07 \sin 5\zeta. \quad (2.129)
\]

Now that we have constructed \( \phi_{app} \), we can apply the change of variable to the Birkhoff-Rott equation. The vortex sheet is then parameterized by \( \hat{p}(p) = \phi_{app}(p) \) for \( p \in [0, 2\pi] \). We can also view the vortex sheet as parameterized by \( p \in [0, 2\pi] \) with the equation of motion modified to

\[
\frac{\partial \hat{z}}{\partial t} = -\frac{1}{4\pi} \int_0^{2\pi} K_\delta (z(\phi(\xi)), z(\phi(\zeta))) \cdot \phi_{app}(\zeta) \, d\zeta. \quad (2.130)
\]
Figure 2.36: How we generate $\varphi$ by hand and use Fourier analysis to construct a smooth approximation. Also shown is the derivative of the function $\varphi$ to make sure that it is not zero.
We solve equation (2.130) numerically and compare the result with the numerical solution when $o(\zeta) = \zeta$ in Figure 2.37. In the former case, note that the vortex sheet is better parameterized as more points are used to parameterize the roll-up. The mapping technique is certainly more computationally expensive than its non-mapping counterpart. We do not expect the mapping method to produce better accuracy than the non-mapping method. In Figure 2.38, the top and bottom plots shows the accuracy in the sheet's position for the non-mapping and mapping cases, respectively. Since the mapping technique is more computationally expensive than the non-mapping version, the former method is expected to produce a result with more accumulated round-off errors.

### 2.6.5 Conclusion

We have examine four methods in an attempt to improve the accuracy of the integral calculation to obtain the instantaneous velocity. However, due to the experimental nature of these methods, for instance the subtracted kernel case, it is not clear how to control the grid size adaptively as a function of time step. Moreover, since the computational complexity of these technique is very high, in practice we do not use them to produce results. More research is needed to apply these more accurate integral evaluation to the time-evolution problem.

Before we proceed to a presentation of results, we note that there are other methods to evaluate the integral of a smoothed kernel efficiently. One such method is Draghicescu's fast summation method. Okamoto and Sakjo [19] has modified this method so the complexity is $O(N \log N)$. This method uses the spectral filter.
Figure 2.37: Redistribution of markers which parameterize the vortex sheet. The top figure shows result for the non-mapping technique, i.e., $\lambda = 0$. The bottom figure shows the result after the mapping.
Figure 2.38: Top figure: accuracy of the sheet position obtain from the non-mapping kernel. Bottom figure: accuracy of the sheet position obtain from the mapping kernel.
In the next section, we take a quick tour in the gallery of double spirals generated numerically by the four smoothed kernels.

2.7 Some Results

In this section, we examine some results generated by the four smoothed kernels considered in this thesis. It is well-known that the time-evolution of the perturbed flat vortex sheet rolls up into a double spiral. Figure 2.39 shows the changes in the sheet profile at $T = 2\pi$ as $\delta$ is decreased from $\delta = 0.4475$ to $\delta = 0.242$. There is a critical value $\delta_c$ at which the sheet first begin to form the spiral. As $\delta$ is decreased below $\delta_c$, the roll-up of the sheet begins to develop. The roll-up is clearly developed at $\delta = 0.242$. Decreasing $\delta$ further creates more turns of the spiral as evident in Figure 2.40 for $\delta$ ranging from 0.2 down to 0.064.

We have so far examined four smoothed kernels. We now investigate similarities and differences between the results they each produce. Vortex sheet profiles are generated using all four kernels at a fixed $\delta = 0.2$ for a final time $T = 2\pi$. The results are displayed in Figure 2.41. As observed from the plots, it would take Krasny's kernel a smaller $\delta$, namely a $\delta < 0.2$, to reach the result obtained by Baker's kernel. We say that Krasny's kernel produced a result much less developed than Baker's kernel for the same $\delta$. Baker's kernel, on the other hand, produces a result much less developed than either of Beale's kernels. More importantly, it appears that there is not much visual differences between Beale's first and second kernel. There is in fact a small difference between Beale's two kernels which are plotted in Figure 2.42 on an enlarged scale.
Figure 2.39: How the vortex sheet changes as $\delta$ is decreased.
Figure 2.40: Generation of more spirals as for smaller and smaller $\delta$. 
Figure 2.41: Comparing vortex sheets generated by the four kernels.
Figure 2.42: Comparing vortex sheet generated by the four kernels.
2.8 Summary of Chapter 2

The curvature singularity that forms during vortex sheet motion is remedied by use of the blob method. Chorin's blob method leads to Baker's kernel in a periodic geometry. Krasny uses Chorin's idea to produce a smoothing of $K_{[0,2\pi]}$. Beale provided another regularization of $K_{\mathbb{R}}$, and also a simpler smoothing of $K_{[0,2\pi]}$.

Numerical implementation of the desingularized equation is introduced. Resolution studies are done to show that the numerical algorithm works correctly. The brackets tables determine the size of the spatial and temporal grids that are necessary to generate a sheet profile accurate to five digits.

Different algorithmic approaches are introduced in an attempt to take advantage of cache architecture to speed-up the code. These approaches involve reordering terms in the sums, and regrouping the sum operations to take advantage of temporal and cache coherence. Besides speeding up the code, attempts are made to obtain more accurate evaluation of the velocity. The filter method, the subtracted kernel, Beale's 3rd-order correction, and the mapping method are four methods examined.

Finally, some results of vortex sheet profiles are shown. As the smoothing $\delta$ is decreased, the vortex sheet profile develops more spirals at the center. With the aid of the bracket tables, it has been shown that an accurate profile at small $\delta$ is very difficult to generate.

2.9 Preludes to Chapter 3 and 4

With the foundation firmly established, the studies of vortex sheets shifts to analyzing their behavior in both the outer and inner regions of the spirals. Chapter 3
examines the sheet's outer region by investigating convergence issue. Chapter 4 studies the inner region of the roll-up by searching for a similarity connection between variables.
CHAPTER 3

CONVERGENCE TO WEAK LIMIT

3.1 Summary

Employing Krasny's initial condition, vortex sheets are generated using the four smoothed kernels up to a final time \( T = 2\pi \) for a range of the smoothing parameter \( \delta > 0 \). In studying the pointwise convergence of a sheet location \( P \) as \( \delta \to 0 \) for a smoothed kernel, the point \( P \) must be chosen so that it can be identified among vortex sheets for any \( \delta \). This remark is important since vortex sheet profiles develop spirals, and thus change shape, as \( \delta \) tends to zero. In our studies of pointwise convergence of vortex sheets, the collection of points \( P \) is the set of intersection points of the sheet profile with the segment \([\pi, 2\pi] \times \{0\}\). Associated to each \( \delta \) is a set \( S(\delta) \) of sorted intersection points (in decreasing order). Let \( s_1(\delta) \) be the maximum element of the set \( S(\delta) \). Numerical evidence indicates that as \( \delta \to 0 \), \( s_1(\delta) \) approaches a unique limit for all four smoothed kernels. A preliminary polynomial form-fit study of the curve \( s_1(\delta) \) is conducted.
3.2 Convergence and Theoretical Issues

There is a huge body of literature on the subject of convergence of vortex methods for discontinuous initial data of which the vortex sheet problem is a candidate. For a small perturbation of a flat vortex sheet, Caffin and Lowengrub [4] proved that the vortex blob approximation converges strongly before the critical time of singularity formation which is estimated by Krasny to be roughly $t_c \approx \frac{3}{4}(2\pi)$. Krasny [9] shows that for $t < t_c$, the point-vortex method converges as the number of point vortices used to describe the sheet increases.

The final time of $T = 2\pi$ is assured to be after the time of singularity. There are theoretical results for convergence of vortex sheet positions for both $T < t_c$ and $T > t_c$. The following important results were derived from a paper by Liu and Xin [12]. It details the convergence of solutions of vortex blob methods in the limit that the grid size and the blob size are approaching zero to a classical weak limit solution of the 2D incompressible Euler equation. For the sake of completion, let us recall the definition of a weak solution of Euler’s equation from a paper by DiPerna and Majda [7].

A vector function $\bar{u}(\vec{x}, t) \in L^{\infty}([0, T), L^2_{x}(\mathbb{R}^2))$ is a classical weak solution of the 2D Euler equations

$$\frac{D}{Dt} \vec{u} = -\frac{1}{\rho} \vec{\nabla} p + \vec{g}. \quad (3.1)$$

$$\vec{\nabla} \cdot \vec{u} = 0. \quad (3.2)$$

on $[0, T]$ with initial velocity $\bar{u}_0$ if

1. For all test function $\theta(\vec{x}, t) \in C^\infty_c(\mathbb{R}^2 \times (0, T))$,

$$\int \int (\vec{\nabla} \cdot \vec{u} + (\vec{\nabla} \otimes \vec{\nabla}) : (\vec{u} \otimes \vec{u})) \ d\vec{x} \ dt = 0. \quad (3.3)$$
2. \( \nabla \cdot \vec{u} = 0 \) in the sense of distribution, and

3. \( \vec{u}(\vec{x}, t) \in \text{Lip}([0, T), H_{\text{loc}}^{m}(\mathbb{R}^2)) \) for some \( m > 0 \) and \( \vec{u}(\vec{x}, 0) = \vec{u}_0(\vec{x}). \)

Here, \( \nabla \cdot = (-\partial_y, \partial_z) \) and \( \mathbf{A} : \mathbf{B} \) denotes the inner product.

In the next section, we shall present numerical results supporting the conjecture that solutions from different smoothed kernels, even those not arising from the convolution of \( K_2 \) with a blob function, converge to the same weak limit solution.

### 3.2.1 The Outer Arms

We analyze the outer region of the vortex spiral. For each \( \delta \), the intersection of the vortex sheet with the \( x \)-axis is calculated. Due to the sheet's symmetry, only intersection points to the right of the sheet's center \((\pi, 0)\) are considered. Thus the vortex sheet profile is intersected with the segment \([\pi, 2\pi) \times \{0\}\) to obtain a nonempty set \( S(\delta) \) containing the \( x \)-coordinates of intersection points. The set \( S(\delta) \) always contain the element \( \pi \) for any \( \delta > 0 \) and therefore \( S(\delta) \neq \emptyset \). The set \( S(\delta) \) can be ordered from large to small. The largest element \( s_1(\delta) \) of \( S(\delta) \) is the outermost intersection point of the sheet with \([\pi, 2\pi) \times \{0\}\). For a lack of a better name, we call \( s_1(\delta) \) the **first arm** point. The next outermost intersection point is the **second arm**, and so on. Figure 3.1 illustrates the positions of the arms.

To obtain the intersection of the sheet profile with the \( x \)-axis, the \( y \)-profile of the sheet must be zero. Therefore, \( y(\xi) = 0 \), and roots of \( y(\xi) \) must be extracted numerically. The intersection points are given by the set of all \((x(\xi'), 0)\) for all roots \( \xi' \) of \( y(\xi) \). The set \( S(\delta) \) only records \( x(\xi') \).
Since the positions $x(\xi) - \xi$ and $y(\xi)$ are odd functions in $\xi$, they can be represented by a truncated sine series

\[
x(\xi) - \xi \approx \sum_{n=1}^{(N-1)/2} a_n \sin n\xi.
\]
\[
y(\xi) \approx \sum_{n=1}^{(N-1)/2} c_n \sin n\xi.
\] (3.4)

The coefficients $a_n$ and $c_n$ are determined by the fast Fourier transform using the discrete profile data. The derivative $y'(\xi)$ is then given by

\[
y'(\xi) \approx \sum_{n=0}^{(N-2)/2} n \cdot c_n \cos n\xi.
\] (3.6)

The bisection rule is applied to perform a preliminary search for possible zeroes of $y(\xi)$. Newton’s algorithm is then used to obtain the list $Z(\delta)$ of $m(\delta)$ zeroes of $y(\xi)$

\[
Z(\delta) = \{ \xi_1(\delta), \xi_2(\delta), \ldots, \xi_{m(\delta)}(\delta) \}
\] (3.7)

to some degree of accuracy. Newton’s algorithm searches for the roots of $y(\xi)$ by the following recursion:

\[
\xi_0 : \text{Initial guess of a root of } y(\xi) \text{ from the bisection method.} \quad (3.8)
\]
\[
\xi_{n-1} = \xi_n - \frac{y(\xi_n)}{y'(\xi_n)}. \quad (3.9)
\]

The algorithm terminates when $|\xi_{n-1} - \xi_n| < \epsilon$ for some tolerance $\epsilon$. Since vortex sheet location data are five digits in accuracy, I took $\epsilon = 10^{-5}$. With the zeroes of $y(\xi)$ determined by the set $Z(\delta)$ given in equation (3.7), the $x$-coordinates of the intersection points are determined by equation (3.4), which are collected in the set $S(\delta)$

\[
S(\delta) = \{ x(\xi_1(\delta)), x(\xi_2(\delta)), \ldots, x(\xi_{m(\delta)}(\delta)) \}
\]
\[
= \{ s_1(\delta), s_2(\delta), \ldots, s_{m(\delta)}(\delta) \}
\] (3.10)
where $s_1(\delta) > s_2(\delta) > \ldots > s_m(\delta)$. 

### 3.3 Results

The general pattern of the function $s_1(\delta)$ for the four smoothed kernels is the same. For large enough $\delta$, for example $\delta = 0.5$, there are no arms in the spiral. As $\delta$ is decreased there is a critical value of $\delta$ when a spiral with one arm first appears. Subsequently, there is a sequence of $\delta$'s for which new arms are generated. As shown in Figure 3.2, Beale's kernel produces the first arm for the largest $\delta$, followed by Baker's kernel, then Krasny's. These results agree with the aforementioned observation that the influence of $\delta$ is much more local for Beale's kernels, moderate for Baker's kernel, and is long-range for Krasny's kernel.

Curiously, we observe some weak oscillatory features in the outer arms as shown in Figure 3.2. We conjecture that these oscillatory patterns arise from the sudden formation of inner arms close enough to the outer arms to produce some nonnegligible influence. In the plot of the second and third arms in Figure 3.3 and 3.4, respectively, the oscillatory behavior is more pronounced, and quite noticeable in Beale's kernels.

Beale's first and second kernels produce first arms that are virtually on top of one another. The differences are shown in Figure 3.5 where a greatly expanded scale has been used. The differences are typically 0.1% and so are invisible in Figure 3.2.

Lastly and perhaps most importantly, the first, second, and third arms from all four kernels do seem to converge to the same point as $\delta \to 0$. More rigorously, the results suggest

$$\lim_{\delta \to 0} s_{1}^{(Krasny)}(\delta) = \lim_{\delta \to 0} s_{1}^{(Baker)}(\delta) = \lim_{\delta \to 0} s_{1}^{(Beale)}(\delta) = s_{1}$$  \hspace{1cm} (3.12)
Figure 3.1: Measuring the arms.
Figure 3.2: Measuring the first outer arm $s_1(\delta)$. 
Figure 3.3: Measuring the second outer arm $s_2(\delta)$.
Figure 3.4: Measuring the third outer arm $s_3(\delta)$. 
Figure 3.5: Beale's first compared to Beale's second Kernel.
Figure 3.6: Wiggles generated by the first outer arms from Beale's second kernel.
for \( i = 1, 2 \) and \( 3 \), and \( s_i \) are real constants.

Unfortunately, due to present computer limitations, we are unable to pick \( \delta \) smaller than 0.03. Already with \( \delta = 0.03 \), Beale's second kernel requires a grid of sixteen thousand points to obtain an accuracy of about nine digits. Of course \( \delta = 0.03 \) may not be small enough to detect small differences in the limits of the arms, but then again, numerical methods can never provide exact results. Nevertheless, one can make the following conjecture that is supported by present numerical results: *In the limit as the smoothing parameter \( \delta \) tends to zero, solutions from different regularizations of the Birkhoff-Rott equation by smoothed kernels converge to a unique weak limit solution of the Euler equation.*

The author is aware of the work by Rottman and Stansby [17] in which circular vortex sheets were studied. In this work, irregular motions were found which dispute our claim of the conjecture above. We also note that their calculations are with the aid of the Fourier filter. In our work, data supporting the hypothesis regarding the uniqueness of a weak limit were done without the use of the filter. We were concerned about initial integration error which propagates in time. Such error may cause instabilities resulting in irregular motion.

Indications of instabilities were observed at \( \delta = 0.043 \) using Baker's kernel. They appear as small oscillations in the outer portion of the vortex sheet calculated on a grid of 4096 points. However, as the grid size is doubled, the oscillations seem to weaken considerably as shown in Figure 3.7. This leads us to believe that irregular motions may not be an inherent feature of vortex sheet motion. They may arise from a lack of grid resolution, or from propagated initial integration error by the spectral filter.
Figure 3.7: Iregular motion on the outer portion of the vortex sheet calculated on a 4096 point grid. Oscillations weaken when the vortex sheet is calculated on a grid of 8196 points.
3.4 Form-Fit of The Outer Arms

At this point, we try a form fit of the curve \( s_1(\delta) \) detailing the position of the first outer arm as a function of the smoothing parameter \( \delta \). Krasny [9] suggested a polynomial form fit but did not actually calculate the coefficients. The polynomial form-fit of degree two is given by

\[
c(\delta) = c_0 + c_1 \delta + c_2 \delta^2.
\]

Certainly one difficulty in determining a polynomial form fit arises from the oscillatory features in the graph of the outer arm. Secondly, as \( \delta \to 0 \), the location of the first arm varies slowly.

We now present the technicalities involved in obtaining the coefficients \( c_0, c_1, \) and \( c_2 \). We examine three values of the smoothing parameters \( \delta_1, \delta_2, \delta_3 \). Then \( s_1(\delta_i) \) is the value of the first arm obtained when in the case of smoothing parameter \( \delta_i \). We have three equations and three unknowns in \( c_0, c_1, \) and \( c_2 \), given by

\[
\begin{align*}
  s_1(\delta_1) & = c_0 + c_1 \delta_1 + c_2 \delta_1^2, \\
  s_1(\delta_2) & = c_0 + c_1 \delta_2 + c_2 \delta_2^2, \\
  s_1(\delta_3) & = c_0 + c_1 \delta_3 + c_2 \delta_3^2.
\end{align*}
\]

This is equivalent to a linear system

\[
\begin{pmatrix}
  1 & \delta_1 & \delta_1^2 \\
  1 & \delta_2 & \delta_2^2 \\
  1 & \delta_3 & \delta_3^2
\end{pmatrix}
\begin{pmatrix}
  c_1 \\
  c_2 \\
  c_3
\end{pmatrix}
= 
\begin{pmatrix}
  s_1(\delta_1) \\
  s_1(\delta_2) \\
  s_1(\delta_3)
\end{pmatrix}.
\]

We can then view the equation above as a matrix problem \( A x = b \) where \( x \) is solved by inverting \( A \) and multiplying the result by the vector \( b \). Before we obtain any numerical results, we must pay attention to the condition number of the matrix \( A \).
The matrix $A$ in equation (3.18) is an example of a Vandermonde matrix which is known to be ill-posed. Nevertheless, we compute the coefficients with the hope that a clear pattern would emerge.

To numerically implement the form fit, our data of the first arm points can be viewed as a list

$$\{ (\delta_i, c(\delta_i)) : i = 1, 2, \ldots, K \} \quad (3.19)$$

where $\delta_i > \delta_{i-1}$. For $K$ values of $\delta$, we have $K - 2$ values of the coefficients $c_0, c_1,$ and $c_2$. We perform a local sliding fit by choosing

$$(\delta_1, \delta_2, \delta_3) = (\delta_i, \delta_{i-1}, \delta_{i-2}) \quad (3.20)$$

at the $i^{th}$ step of the calculation for $i \in \mathbb{Z}[0, K - 2]$. Figure 3.6 presents a graph of the three coefficients as a function of step $i$. Also shown is the logarithm of the condition number. In Figure 3.9 we attempted a form fit using a third degree polynomial. Due to the ill-posed nature of the matrix $A$, the values of the polynomial coefficients vary over a wide region. The condition number of $A$ is larger in the case of the third-order polynomial by a factor of 10 as seen in Figure 3.10

### 3.5 Future Work

Clearly the polynomial form-fit does not produce coefficients that converge. More research are needed to find a better form-fit model for which the form-fit procedure is not ill-posed. It is not at all clear at the moment how to proceed.
Figure 3.8: Degree 2 polynomial form fit $a + bx + cx^2$. 
Figure 3.9: Degree 3 polynomial form fit $a + bx + cx^2 + dx^3$. 
Figure 3.10: Condition number of the matrix $A$ for degree 3 polynomial form fit $a + b x + c x^2 + d x^3$ as a function of $\delta$. 
CHAPTER 4

SIMILARITY STUDIES

4.1 Motivation For Similarity Studies

Using any one of the four smoothed kernels considered earlier, there are two approaches that can be applied to generate vortex sheet roll-up. In one method, the final time $T = 2\pi$ (with $T > t_c$) is held fixed, and the smoothing parameter $\delta$ is decreased. We refer to this method as method (A). Alternatively, in method (B), let $\delta$ stay a constant, and increase final time $T$.

In Figure 4.1, I apply Beale's first kernel with a fixed final time of $T = 2\pi$, and $\delta$ decreasing from 0.5 to 0.2. As $\delta$ is decreased, the sheet rolls up into a double spiral with the number of turns increasing with decreasing $\delta$. The same phenomenon takes place when $\delta$ is held fixed at a value of 0.1, and time is increased from $T = 1.0$ to $T = 4.0$. Beale's first kernel was also used in this case with the results shown in Figure 4.2.

These two apparently different mechanisms seem to produce results that appear to be similar, though not identical. The roll-up region in the first case appears to be larger than that in the second case. Such similarity in the sheet's shape and structure generated by two different methods motivates the idea of variable rescaling.
rescaled variables. It is the hope that both methods (A) and (B) produce identical results in some region of the sheet. More specifically, we search for a similarity relation between the smoothing variable $\delta$ and final time $T$.

In order for the vortex sheet to develop spirals, either from method (A) or (B), the sheet's tangent line $\ell$ at the center must be rotating as spirals are formed. As the roll-up phenomenon develops, the angle $\theta$ that $\ell$ makes with the positive $x$-axis can be tracked (see Figure 4.3). It is important that $\theta$ is an increasing function. In other words, for method (A), $\theta^{(A)}(\delta, T)$ is an increasing function with decreasing $\delta$. The function $\theta^{(B)}(\delta, T)$ is an increasing function with increasing $T$ for method (B). The superscript on the variable $\theta$ denotes the method.

For the vortex sheet profiles generated from method (A) and (B) to look similar, at any angle $\alpha$, there exists final times $T_A$ and $T_B$, smoothing parameters $\delta_A$ and $\delta_B$ such that

$$\alpha = \theta^{(A)}(\delta_A, T_A) = \theta^{(B)}(\delta_B, T_B).$$

(4.1)

We like to understand the connection between the five variables $T_A$, $T_B$, $\delta_A$, $\delta_B$, and $\alpha$. But first, the angle $\theta$ must be measured numerically at each time step.

At any time $t$, the vortex sheet location was observed to be symmetric (Chapter 1). In particular, the functions $(x(\xi) - \xi)$ and $y(\xi)$ are odd functions of $\xi$. On an $N$-point grid, they can be represented as truncated Fourier sine series given by

$$x(\xi) = \xi + \sum_{k=1}^{N/2-1} b_k \sin k\xi.$$  \hspace{1cm} (4.2)

$$y(\xi) = \sum_{k=1}^{N/2-1} J_k \sin k\xi.$$  \hspace{1cm} (4.3)

The coefficients $b_k$ and $J_k$ are numerically computed using the FFTW libraries. The derivatives $x'(\xi)$ and $y'(\xi)$ are obtained by differentiating the finite sum (4.2) and
Figure 4.1: Evolution of the vortex sheet for fixed final time as the smoothing parameter $\delta$ is decreased.
Figure 4.2: Evolution of the vortex sheet for fixed $\delta$ and increasing final time.
Figure 4.3: Measuring the angle at the core.
respectively. These tasks are equivalent to modifying the coefficients $b_k$ and $J_k$ by a multiplicative factor $k$ ($0 < k < \pi/2$), and performing an inverse cosine transform. The ratio $y'(\pi)/x'(\pi)$ gives the slope at the point $(\pi, 0)$ on the sheet when $x'(\xi) \neq 0$. When $x'(\xi) = 0$, the tangent line is a vertical line. In both cases, the time-dependent angle $\theta(t)$ can be computed at each time step. To make sure the program works correctly, the angle at the zeroth time step can be checked with the exact angle value calculated from the initial condition $z_0(\xi) = x_0(\xi) + iy_0(\xi)$ to be

$$\theta(0) = \tan^{-1}\left(\frac{y'_0(\pi)}{x'_0(\pi)}\right) = \tan^{-1}\left(\frac{\pi}{50 - \pi}\right) = 0.0324235 \text{ radian.} \quad (4.4)$$

4.2 Similarity Studies with Baker’s Kernel

Our similarity studies make use of Baker’s kernel due to the fact that there are asymptotics studies currently being developed for this kernel. Vortex sheet profiles are examined for a range of smoothing parameters from $\delta = 0.01$ to $\delta = 0.10$ in steps of $\Delta \delta = 0.05$. Due to the small magnitude of $\delta$, the calculation is done using the adaptive point-insertion technique with the filtering level set at $10^{-13}$. Initial grid size is set at $N = 1024$. Given a time step size $\Delta t$, the program computes at each $t_k = k \Delta t$ the angle $\theta(t_k)$. The program terminates when $\theta(t_w) > 6\pi$ for some time step $w$. The function $T_\delta(\theta)$ returns the time the tangent to the sheet center makes an angle $\theta$ with the positive $x$-axis for a smoothing parameter $\delta$.

The graph of $T_\delta(\theta)$ for $0.01 \leq \delta \leq 0.10$ (in steps of 0.005) is shown in Figure 4.4. There are two prominent features to observe. First, the smaller the value of $\delta$, the shorter the time it takes for the sheet center to rotate to an angle $\theta$. Secondly,
Figure 4.4: Time as a function of the sheet angle at the center for $\delta$ ranging from 0.01 to 0.10 in steps of $\Delta \delta = 0.05$. 
Figure 4.5: Studying the time the sheet center angle $\theta$ achieves an angle $\theta$ as a function of smoothing variable $\delta$ for $\theta = \pi, 3\pi/2, \ldots, 6\pi$. 
Figure 4.6: The slope and intersection of each of the lines in the previous plot using a least-square linear fit.
beyond a threshold angle of approximately 2.5 radians, the function \( T_\delta(\theta) \) appears to be a linear function of \( \theta \) for each \( \delta \).

Another way of viewing the result shown in Figure 4.4 is to let \( \theta \) be fixed at some constant angle \( \alpha \), and study the dependence of \( T_\delta(\alpha) \) on \( \delta \). This is equivalent to studying vertical slices of the graph in Figure 4.4. The result is shown in Figure 4.5 for \( \alpha \) ranging from \( \pi \) to \( 6\pi \) in steps of \( \pi/2 \). In our numerical studies, we examine angle \( \alpha \) from \( \pi \) to \( 6\pi \) in step size of \( \pi/8 \), but show only a few curves in Figure 4.5 due to their close spacings.

The observed regularity in the family of lines from Figure 4.5 suggests that there is one equation that can describe them. Each line is a linear function in the variable \( \delta \), with the slope \( m \) and the \( y \)-intercept \( b \) functions of \( \alpha \). The family of lines can thus be described as

\[
T_\delta(\alpha) = m(\alpha) \cdot \delta + b(\alpha). \tag{4.5}
\]

Each line in the Figure 4.5 has a slope and \( y \)-intercept. We apply the least-square linear fit to obtain this slope and \( y \)-intercept, and plot them as functions of \( \alpha \) in Figure 4.6. One can again obtain a least-square linear fit to the functions \( m(\alpha) \) and \( b(\alpha) \) to obtain

\[
b(\alpha) = -0.005\alpha + 2.4576. \tag{4.6}
\]

\[
m(\alpha) = 1.1485\alpha + 18.6725. \tag{4.7}
\]

Now the equation 4.5 can be described by

\[
T_\delta(\alpha) = (1.1485\alpha + 18.6725)\delta - 0.005\alpha + 2.4576. \tag{4.8}
\]

One notes that the constant \(-0.005\) is smaller than the other three constants in the equation. At the moment, we do not have an error estimate on the accuracy of the
least-square linear fit. If the constant $-0.005$ falls within the error range, then it could possibly be zero. In such a case, the equation 4.8 can be rewritten as

$$\frac{T_\delta(\alpha) - 2.4576}{\delta} = 1.1485\alpha + 18.6725. \quad (4.9)$$

Equation 4.9 provides a similarity relation between the variable $\delta$ and final time $T$ when the angle $\alpha$ is fixed. Moreover, the constant $2.4576$ is very close to the value of the critical time of singularity formation given by Krasny to be

$$(\text{Krasny}) \quad t_c = \frac{3}{8}(2\pi) = 2.3561945. \quad (4.10)$$

It is the belief that equation (4.9), to the leading order in $\delta$, has the form

$$\frac{T_\delta(\alpha) - t_c}{\delta} = A\alpha + B \quad (4.11)$$

where $A$ and $B$ are constants.

### 4.3 Confirming The Similarity Relation

Using the similarity relation given in equation (4.9) with $\alpha = 5.0$ radians, we calculate vortex sheet profiles for $\delta = 0.06$ to $\delta = 0.10$ in step of $\Delta\delta = 0.01$. The final time of the vortex sheet profile is different for each $\delta$, and it is calculated by equation (4.9).

The top graph in Figure 4.7 shows the five vortex sheet profiles associated to $\delta$ ranging from 0.06 to 0.10. The regular spacing between the curves suggests that there is a rescaling of $x$ and $y$ variables that can be used to collapse the curves into one. We apply the rescaling

$$\frac{x(p) - \pi}{\delta} \quad \text{and} \quad \frac{y(p)}{\delta} \quad (4.12)$$

126
Figure 4.7: Similarity studies result 1.
Figure 4.8: Similarity studies result 2.
Figure 4.9: Similarity studies result 3.
to each of the five profiles, and replotted them. The results are shown on the bottom graph of Figure 4.7. Near the center of the spiral, all five curves seem to collapse onto one curve. Figure 4.8 provides a closer examination of the plots in Figure 4.7.

Due to the inaccuracy of the constants in the similarity relation (4.9), it is expected that all five curves do not fall perfectly on top of one another in the roll-up region. Figure 4.9 zooms in much closer to the roll-up region, and shows that there is slight gap between the curves.

Further work is needed to obtain a more accurate similarity relation between final time $T$, smoothing parameter $\delta$, and the angle $\alpha$. It may be that due to the higher-order corrections in $\delta$, vortex sheet positions do not necessarily fall on top of one another. The size of the gap between the curves may hint at the form of the asymptotics expansion for higher-order correction terms in $\delta$. 
APPENDIX A

THE METHOD OF IMAGES

The exact equation of motion for a vortex sheet in complex notation is

\[
\frac{\partial \zeta}{\partial t}(\xi, t) = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \frac{dr}{z(\xi, t) - z(r, t)}. \tag{A.1}
\]

In real and imaginary parts, the equation of motion becomes

\[
\frac{\partial \xi}{\partial t}(\xi, t) = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \frac{[x(\xi, t) - x(r, t)] - i[y(\xi, t) - y(r, t)]}{[x(\xi, t) - x(r, t)]^2 + [y(\xi, t) - y(r, t)]^2} \, dr \tag{A.2}
\]

One can sum over all intervals on the real line of length \(2\pi\) of the form \([2\pi k, 2\pi(k + 1))\) to obtain

\[
\frac{\partial \zeta}{\partial t}(\xi, t) = \frac{1}{2\pi i} \sum_{k=-\infty}^{\infty} \int_{2\pi k}^{2\pi(k+1)} \frac{x(\xi, t) - x(r, t)}{[x(\xi, t) - x(r, t)]^2 + [y(\xi, t) - y(r, t)]^2} \, dr. \tag{A.3}
\]

Naturally, one applies a change of variable \(\zeta = r - 2\pi k\). The resulting equation will contain terms of the type \(x(\zeta + 2\pi k)\) and \(y(\zeta + 2\pi k)\). Recall the periodicity conditions of the sheet

\[
x(\zeta + 2\pi) = 2\pi + x(\zeta) \quad \text{and} \quad y(\zeta + 2\pi) = y(\zeta). \tag{A.4}
\]

The resulting equation of motion with conditions given in equation (A.4) becomes

\[
\frac{\partial \zeta}{\partial t}(\xi, t) = \frac{1}{2\pi i} \sum_{k=-\infty}^{\infty} \int_0^{2\pi} \frac{[x(\xi) - x(\zeta) - 2\pi k] - i[y(\xi) - y(\zeta)]}{[x(\xi) - x(\zeta) - 2\pi k]^2 + [y(\xi) - y(\zeta)]^2} \, d\zeta. \tag{A.5}
\]
One can further simplify notation by introducing

\[ X = x(\xi, t) - x(\zeta, t) \quad \text{and} \quad Y = y(\xi, t) - y(\zeta, t). \]  

(A.6)

The equation of motion now becomes

\[
\frac{\partial z}{\partial t}(\xi) = \frac{1}{2\pi i} \int_{0}^{2\pi} \sum_{k=\infty}^{\infty} \frac{(X - 2\pi k) - iY}{(X - 2\pi k)^2 + Y^2} \, d\zeta = \frac{1}{2\pi i} \int_{0}^{2\pi} S(q) \, dq. \tag{A.7}
\]

One would like to evaluate the sum

\[
S(q) = \sum_{k=\infty}^{\infty} \frac{(X - 2\pi k) - iY}{(X - 2\pi k)^2 + Y^2}. \tag{A.8}
\]

To do this, one interprets the sum as a residue of some function \( f \) over a closed contour \( C \) which encloses some singularities. Since the index of the sum is \( k \), the singularities must occur at \( 2\pi k \). Hence for some function \( f \), we like to interpret \( S(q) \) as

\[
S(q) = \sum_{k=\infty}^{\infty} \text{Res}(f, 2\pi k). \tag{A.9}
\]

Now the question becomes, what is \( f \)? \( f \) must have singularities at \( 2\pi k \), say poles of order 1, and its residue there must evaluate to the \( k \)th summand, i.e.,

\[
\text{Res}(f, 2\pi k) = \frac{(X - 2\pi k) - iY}{(X - 2\pi k)^2 + Y^2}. \tag{A.10}
\]

One candidate for \( f \) is

\[
f(z) = \frac{1}{2} \frac{(X - iY) - z}{(X - z)^2 + Y^2} \frac{\cos(z/2)}{\sin(z/2)} = \frac{1}{2} \frac{1}{z - (X + iY)} \frac{\cos(z/2)}{\sin(z/2)}. \tag{A.11}
\]

The poles of \( f \) are

\[
\text{Pole}(f) = \{ 2\pi k : k \in \mathbb{Z} \} \cup \{ X + iY \}. \tag{A.12}
\]
It is clear that $2\pi k$ is a simple pole of $f$, and thus

$$\text{Res}(f : 2\pi k) = \lim_{z \to 2\pi k} (z - 2\pi k) f(z)$$

$$= \lim_{z \to 2\pi k} \frac{1}{2} \frac{(X - iY) - z}{(X - z)^2 + Y^2} (-1)^k \frac{z - 2\pi k}{\sin(z/2)}$$

We massage the $\sin(z/2)$ term by

$$\sin(z/2 - 2\pi k/2 + 2\pi k/2) = \sin((z/2 - 2\pi k/2) + 2\pi k/2)$$

$$= \sin(z/2 - 2\pi k/2) \cos \pi k$$

$$= (-1)^k \sin((z - 2\pi k)/2).$$

Thus we have shown that

$$\text{Res}(f : 2\pi k) = (-1)^k \frac{(X - iY) - 2\pi k}{(X - 2\pi k)^2 + Y^2} \lim_{z \to 2\pi k} \frac{(z - 2\pi k)/2}{(-1)^k \sin((z - 2\pi k)/2)}$$

$$= \frac{(X - iY) - 2\pi k}{(X - 2\pi k)^2 + Y^2}. $$

For a square $C(r)$ centered at the origin of width $r$, it can be shown that

$$\frac{1}{2\pi i} \int_{C(r)} f(z) \, dz = \sum_{k|k| \leq r} \text{Res}(f : 2\pi k)$$

In the limit as $r$ approaches infinity, $C(r)$ will enclose all singularities of $f$, and thus

$$\lim_{r \to \infty} \frac{1}{2\pi i} \int_{C(r)} f(z) \, dz = 0. \quad (A.16)$$

But since

$$\lim_{r \to \infty} \frac{1}{2\pi i} \int_{C(r)} f(z) \, dz = S(q) + \text{Res}(f : X + iY). \quad (A.17)$$

we have,

$$S(p) = -\text{Res}(f : X + iY). \quad (A.18)$$
Since $X + i Y$ is a simple pole, the residue of $f$ at these points are easily computed to be

$$S(p) = \frac{1}{2} \cot \left( \frac{X + i Y}{2} \right). \quad (A.19)$$

The equation of motion now becomes

$$\frac{\partial z}{\partial t}(p) = \frac{1}{4\pi i} \int_0^{2\pi} \cot \left[ \frac{(x(p) - x(q)) + i(y(p) - y(q))}{2} \right] dq \quad (A.20)$$

Since $z$ is periodic, $z$ is also a function of a variable $p$ where $p \in [0, 2\pi)$. Thus we have

$$\frac{\partial z}{\partial t}(p) = \frac{1}{4\pi i} \int_0^{2\pi} \cot \left[ \frac{(x(p) - x(q)) + i(y(p) - y(q))}{2} \right] dq \quad (A.21)$$

Next, we derive a real and imaginary parts of the integrand. We use the following identity

$$\cot(\alpha + i\beta) = \frac{\sin 2\alpha - i \sinh 2\beta}{\cosh 2\beta - \cos 2\alpha} \quad (A.22)$$

to obtain

$$\frac{\partial z}{\partial t}(p) = -\frac{1}{4\pi} \int_0^{2\pi} \frac{\sinh(y(p) - y(q)) + i \sin(x(p) - x(q))}{\cosh(y(p) - y(q)) - \cos(x(p) - x(q))} dq. \quad (A.23)$$

This method is called the method of images. As a reminder, this is an exact formulation of the motion of a vortex sheet in a $[0,2\pi]$-periodic geometry. We will call this kernel $K_{\text{per}}$ as a reminder.

$$K_{\text{per}}(p, q) = \frac{\sinh(y(p) - y(q)) + i \sin(x(p) - x(q))}{\cosh(y(p) - y(q)) - \cos(x(p) - x(q))}. \quad (A.24)$$

the exact periodic kernel, which is nothing other than the periodic version of $K$.

### A.1 Beale’s Second Kernel

In Beale’s second kernel, we need to evaluate one additional term

$$I = \frac{1}{2\pi i} \int_{-\infty}^{-\infty} \frac{(x - \tilde{x}) - i(y - \tilde{y})}{(x - \tilde{x})^2 + (y - \tilde{y})^2} \cdot \exp \left[ - \frac{(x - \tilde{x})^2 + (y - \tilde{y})^2}{\delta^2} \right] \ dq \quad (A.25)$$
Again, using the periodicity of the vortex sheet, we have

\[
l = -\frac{1}{2\pi i} \sum_{k \in \mathbb{Z}} \int_{0}^{2\pi} \frac{\Delta x - 2\pi k - i \Delta y}{(\Delta x - 2\pi k)^2 + (\Delta y)^2} \cdot e^{-\frac{\Delta x - 2\pi k - i \Delta y}{\Delta x - 2\pi k - i \Delta y}} \, dp \quad \text{(A.26)}
\]

\[
= \frac{i}{2\pi} \sum_{k \in \mathbb{Z}} \int_{0}^{2\pi} \frac{\Delta x - 2\pi k - i \Delta y}{(\Delta x - 2\pi k)^2 + (\Delta y)^2} \cdot e^{-\frac{\Delta x - 2\pi k - i \Delta y}{\Delta x - 2\pi k - i \Delta y}} \, dp \quad \text{(A.27)}
\]

\[
= \frac{1}{2\pi} \sum_{k \in \mathbb{Z}} \int_{0}^{2\pi} \frac{i(\Delta x - 2\pi k) + \Delta y}{(\Delta x - 2\pi k)^2 + (\Delta y)^2} \cdot e^{-\frac{\Delta x - 2\pi k - i \Delta y}{\Delta x - 2\pi k - i \Delta y}} \, dp \quad \text{(A.28)}
\]

\[
= -\frac{1}{4\pi} \cdot -2 \sum_{k \in \mathbb{Z}} \int_{0}^{2\pi} \frac{\Delta y + i(\Delta x - 2\pi k)}{(\Delta x - 2\pi k)^2 + (\Delta y)^2} \cdot e^{-\frac{\Delta x - 2\pi k - i \Delta y}{\Delta x - 2\pi k - i \Delta y}} \, dp \quad \text{(A.29)}
\]
APPENDIX B

COMPLETE DEBUG TABLES

In this appendix, we archive the complete debug tables that are used to generated Figures 2.4 and 2.5 from Chapter 2. The following figures record, for all four smoothed kernels, the quantities $c_{x,M}^r(\pi/4; \delta)$ and $c_{x,M}^r(\pi/4; \delta)$ for two values of $\delta$: 0.1 and 0.05. Due to the physical dimensions of the tables, the first table starts on next page.
<table>
<thead>
<tr>
<th>$N$</th>
<th>$c_{N,M}^a(\xi: 0.1)$</th>
<th>$c_{N,M}^b(\xi: 0.1)$</th>
<th>$c_{N,M}^c(\xi: 0.05)$</th>
<th>$c_{N,M}^d(\xi: 0.05)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>7.56974</td>
<td>8.21913</td>
<td>6.85179</td>
<td>7.95699</td>
</tr>
<tr>
<td>24</td>
<td>8.77598</td>
<td>8.36317</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>9.9297</td>
<td>8.98325</td>
<td>10.6239</td>
<td>9.67785</td>
</tr>
<tr>
<td>40</td>
<td>11.0675</td>
<td>9.76439</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>13.3316</td>
<td>11.5512</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>14.4623</td>
<td>12.5072</td>
<td>10.6239</td>
<td>9.67785</td>
</tr>
<tr>
<td>72</td>
<td>15.5929</td>
<td>13.4889</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>16.7335</td>
<td>14.4898</td>
<td>11.7624</td>
<td>10.4601</td>
</tr>
<tr>
<td>88</td>
<td>17.8541</td>
<td>15.5056</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>18.9847</td>
<td>16.5332</td>
<td>12.8959</td>
<td>11.3278</td>
</tr>
<tr>
<td>104</td>
<td>20.1153</td>
<td>17.5704</td>
<td></td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>22.3766</td>
<td>19.6675</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>23.5073</td>
<td>20.7251</td>
<td>15.1592</td>
<td>13.2054</td>
</tr>
<tr>
<td>160</td>
<td>25.7693</td>
<td></td>
<td>17.4216</td>
<td>15.1896</td>
</tr>
<tr>
<td>176</td>
<td>26.8994</td>
<td></td>
<td>18.5528</td>
<td>16.2061</td>
</tr>
<tr>
<td>192</td>
<td>28.0295</td>
<td></td>
<td>19.684</td>
<td>17.2344</td>
</tr>
<tr>
<td>208</td>
<td>29.1596</td>
<td></td>
<td>20.8152</td>
<td>18.2723</td>
</tr>
<tr>
<td>224</td>
<td>30.2897</td>
<td></td>
<td>21.9464</td>
<td>19.3181</td>
</tr>
<tr>
<td>240</td>
<td>31.4198</td>
<td></td>
<td>23.0776</td>
<td>20.3707</td>
</tr>
<tr>
<td>256</td>
<td>32.5499</td>
<td></td>
<td>24.2088</td>
<td>21.4290</td>
</tr>
</tbody>
</table>

Table B.1: Complete debug data for Krasny's kernel.
<table>
<thead>
<tr>
<th>$N$</th>
<th>$c_{N,M}^R (\xi: 0.1)$</th>
<th>$c_{N,M}^R (\xi: 0.05)$</th>
<th>$c_{N,M}^R (\xi: 0.1)$</th>
<th>$c_{N,M}^R (\xi: 0.05)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>7.13671</td>
<td>6.65936</td>
<td>7.26147</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>8.06711</td>
<td>8.51692</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>8.92058</td>
<td>9.61435</td>
<td>9.3250</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>9.74369</td>
<td>10.0722</td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>10.5539</td>
<td>8.76087</td>
<td>9.20869</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>11.3585</td>
<td>9.61435</td>
<td>9.325</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>12.1605</td>
<td>10.6769</td>
<td>11.3121</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>13.7619</td>
<td>11.9699</td>
<td>10.4375</td>
<td>9.7079</td>
</tr>
<tr>
<td>88</td>
<td>14.5621</td>
<td>12.6452</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>15.3622</td>
<td>13.3341</td>
<td>11.2477</td>
<td>10.2041</td>
</tr>
<tr>
<td>104</td>
<td>16.1623</td>
<td>14.0341</td>
<td></td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>16.9623</td>
<td>14.7432</td>
<td>12.0523</td>
<td>10.7637</td>
</tr>
<tr>
<td>120</td>
<td>17.7624</td>
<td>15.4598</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>18.5624</td>
<td>16.1823</td>
<td>12.8543</td>
<td>11.3704</td>
</tr>
<tr>
<td>144</td>
<td>19.3624</td>
<td>16.8856</td>
<td>12.0056</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>14.4556</td>
<td>12.6635</td>
<td></td>
<td></td>
</tr>
<tr>
<td>176</td>
<td>15.2558</td>
<td>13.3388</td>
<td></td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>16.0559</td>
<td>14.0277</td>
<td></td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>16.856</td>
<td>14.7277</td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>17.656</td>
<td>15.4368</td>
<td></td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>18.456</td>
<td>16.1534</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>19.256</td>
<td>16.8765</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B.2: Complete debug data for Baker's Kernel.
<table>
<thead>
<tr>
<th>N</th>
<th>$c_{X,M}^{u}(\xi:0.1)$</th>
<th>$c_{X,M}^{w}(\xi:0.1)$</th>
<th>$c_{X,M}^{u}(\xi:0.05)$</th>
<th>$c_{X,M}^{w}(\xi:0.05)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>6.83389</td>
<td>8.55719</td>
<td>6.18884</td>
<td>6.8332</td>
</tr>
<tr>
<td>24</td>
<td>7.7633</td>
<td>7.773</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>8.89515</td>
<td>7.75834</td>
<td>7.52661</td>
<td>9.25461</td>
</tr>
<tr>
<td>40</td>
<td>10.3312</td>
<td>8.5425</td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>12.0815</td>
<td>9.83081</td>
<td>8.4556</td>
<td>8.46282</td>
</tr>
<tr>
<td>56</td>
<td>14.1451</td>
<td>11.5303</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>16.5207</td>
<td>13.6013</td>
<td>9.58813</td>
<td>8.44883</td>
</tr>
<tr>
<td>72</td>
<td>19.208</td>
<td>16.0221</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>22.2082</td>
<td>18.7792</td>
<td>11.0275</td>
<td>9.23647</td>
</tr>
<tr>
<td>88</td>
<td>25.5244</td>
<td>21.8635</td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>29.1637</td>
<td>25.2685</td>
<td>12.785</td>
<td>10.5323</td>
</tr>
<tr>
<td>104</td>
<td>33.1327</td>
<td>28.9907</td>
<td></td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>37.067</td>
<td>32.9939</td>
<td>14.8609</td>
<td>12.2452</td>
</tr>
<tr>
<td>120</td>
<td>39.1039</td>
<td>35.2053</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>38.5031</td>
<td>35.484</td>
<td>17.2547</td>
<td>14.3372</td>
</tr>
<tr>
<td>144</td>
<td>38.5031</td>
<td>35.484</td>
<td>17.2547</td>
<td>14.3372</td>
</tr>
<tr>
<td>160</td>
<td></td>
<td></td>
<td>19.9663</td>
<td>16.7886</td>
</tr>
<tr>
<td>176</td>
<td></td>
<td></td>
<td>22.996</td>
<td>19.5875</td>
</tr>
<tr>
<td>192</td>
<td></td>
<td></td>
<td>26.3444</td>
<td>22.7261</td>
</tr>
<tr>
<td>208</td>
<td></td>
<td></td>
<td>30.0126</td>
<td>26.1988</td>
</tr>
<tr>
<td>224</td>
<td></td>
<td></td>
<td>33.9701</td>
<td>29.9924</td>
</tr>
<tr>
<td>240</td>
<td></td>
<td></td>
<td>36.9444</td>
<td>34.4563</td>
</tr>
<tr>
<td>256</td>
<td></td>
<td></td>
<td>38.1231</td>
<td>39.5094</td>
</tr>
</tbody>
</table>

Table B.3: Complete debug data for Beale's first Kernel.
<table>
<thead>
<tr>
<th>N</th>
<th>$c_{X,M}^N(\xi: 0.1)$</th>
<th>$c_{X,M}^N(\xi: 0.05)$</th>
<th>$c_{X,M}^N(\xi: 0.05)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>6.83303</td>
<td>8.5555</td>
<td>6.83316</td>
</tr>
<tr>
<td>24</td>
<td>7.76142</td>
<td>7.7716</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>8.89316</td>
<td>7.75564</td>
<td>9.25413</td>
</tr>
<tr>
<td>40</td>
<td>10.3322</td>
<td>8.54272</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>12.0909</td>
<td>9.83935</td>
<td>8.46247</td>
</tr>
<tr>
<td>56</td>
<td>14.1708</td>
<td>11.5552</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>16.5739</td>
<td>13.6534</td>
<td>8.4818</td>
</tr>
<tr>
<td>72</td>
<td>19.3039</td>
<td>16.1161</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>22.3664</td>
<td>18.9337</td>
<td>9.23653</td>
</tr>
<tr>
<td>88</td>
<td>25.7708</td>
<td>22.1016</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>29.5325</td>
<td>25.6186</td>
<td>10.5345</td>
</tr>
<tr>
<td>104</td>
<td>33.679</td>
<td>29.489</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>37.4256</td>
<td>33.9036</td>
<td>12.2515</td>
</tr>
<tr>
<td>120</td>
<td>38.8162</td>
<td>35.484</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>37.8046</td>
<td>36.3313</td>
<td>14.3505</td>
</tr>
<tr>
<td>144</td>
<td>39.9906</td>
<td>19.6127</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>23.036</td>
<td>19.6272</td>
<td></td>
</tr>
<tr>
<td>176</td>
<td>26.4062</td>
<td>22.7875</td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>30.095</td>
<td>26.2891</td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>33.7333</td>
<td>25.818</td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>34.6754</td>
<td>33.7118</td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>34.8044</td>
<td>34.3389</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>38.0053</td>
<td>34.5894</td>
<td></td>
</tr>
</tbody>
</table>

Table B.4: Complete debug data for Beale's second Kernel.
APPENDIX C

COMPUTER CACHE ARCHITECTURE

Present-day computer memory architecture can be described in a hierarchy. Inside the CPU are 32 registers. These are very fast memory units used to store the most fundamental computer instructions at the machine language level.

Most of today’s CPU have cache memory built-in. The first level cache (L1 cache) is a small but fast memory unit with a typical capacity of 32 Kbytes and access time several clock cycles. Data needed by the CPU is searched first in the L1 cache. If it is not found, then the second level L2 cache is accessed. Typical L2 cache capacity is 256 to 512 Kbytes, and access time about 30-60 clock cycles. Server-type CPUs, such as the Intel Xeon processors, have a large L2 cache with capacity up to 2 Mbytes.

Data that resides in the L1 cache must also reside in the L2 cache. If the CPU needs data that is not in the L2 cache, then it access main memory. Main memory access time is on the order of hundreds of clock cycles. In Figure C.1, we show a typical memory hierarchy that is common to today’s computers.
Figure C.1: Memory hierarchy of today’s computers.
BIBLIOGRAPHY


