Radial Basis Functions and Vortex Methods and their Application to Vortex Dynamics on a Rotating Sphere

by

Lei Wang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Applied and Interdisciplinary Mathematics)
in The University of Michigan
2010

Doctoral Committee:

Professor John P. Boyd, Co-Chair
Professor Robert Krasny, Co-Chair
Associate Professor Divakar Viswanath
Assistant Professor Christiane Jablonowski

To the memory of Meiyu Zheng. I miss you and think of you often.

ACKNOWLEDGEMENTS

This thesis work could not have been finished without the help from a number of persons to whom I owe a great debt of gratitude, and I would like to thank them for their valuable contributions.

First, I want to express my deep and sincere appreciation to Professor Robert Krasny, who supervised my study and research work during the past five years. The high standard he always sets for himself has been inspiring to me. Besides the knowledge he has given me, I am particularly grateful for his enormous patience and encouragement during several difficult times.

I would also like to sincerely thank to Professor John P. Boyd as my advisor outside the math department, who also supervised my studies and research during the past five years. Professor Boyd's wide knowledge and logical way of thinking have been of great value to me. He is the person who taught me to do math in a different way.

I want to express my gratitude to Professor Christiane Jablonowski, Professor Divakar Viswanath and Professor Smadar Karni. Their help and support have been invaluable.

Last but not least, I want to thank my husband, Danqing Wu, and my other family members in China, for the love and support they have provided.

TABLE OF CONTENTS

DEDICATION . . . . . **ii**

ACKNOWLEDGEMENTS . . . . . **iii**

LIST OF FIGURES . . . . . **vii**

LIST OF TABLES . . . . . **xii**

CHAPTER

I. Introduction 1

1.1 Motivation for Thesis 1
1.2 Overview of Thesis Topics 5

II. Radial Basis Function Analysis 7

2.1 Introduction 7
2.2 RBFs Accuracy and Stability 9
2.3 Computational cost 11
2.4 Basic Formulas for Gaussian RBF 13
2.4.1 Computing the coefficients of an RBF expansion 14
2.4.2 Poisson summation 15
2.5 Cardinal Function 15
2.5.1 Derivation 16
2.6 Comparison to Finite Differences Using Fourier Analysis 19
2.6.1 Eigenvalues of the RBF difference operators for $\exp(iKX)$ 19
2.6.2 Numerical experiments 21

III. Fast Treecode for Evaluating RBFs 25

3.1 Introduction 25
3.2 Build Tree 26
3.3 Particle-Cluster Interaction 28
3.3.1 Well-separated particle-cluster interaction list 1 29
3.3.2 Well-separated particle-cluster interaction list 2 29
3.3.3 Far field expansion 31
3.4 Recurrence Relation for Taylor Series 32
3.4.1 Multiquadric 32
3.4.2 Gaussian 33
3.4.3 Inverse multiquadric 34
3.5 Treecode Algorithm 34
3.6 Cartesian Taylor Treecode for Multiquadric RBF 35
3.6.1 Far field expansion for multiquadric in 1D 36

3.6.2 The generalized multiquadric in muti-D . . . 38
3.6.3 Treecode performance . . . 40
3.6.4 Random nodes in a cube . . . 42
3.6.5 Random nodes on a sphere . . . 44

IV. Barotropic Vorticity Equation . . . 48

4.1 Introduction . . . 48
4.2 Solution of Poisson Equation on the Sphere . . . 50
4.2.1 Spherical harmonics . . . 50
4.2.2 Green's function . . . 52
4.2.3 Regularized Green's Function . . . 56
4.2.4 Gaussian forcing . . . 57
4.3 Rossby-Haurwitz Wave . . . 59
4.3.1 Stream function . . . 59
4.3.2 Example . . . 60

V. Solving the Barotropic Vorticity Equation by Gaussian RBF . . . 64

5.1 Soving PDEs by RBF . . . 64
5.2 Solving BVE by Gaussian RBF . . . 65
5.3 Rossby-Haurwitz Wave . . . 68
5.4 One Vortex Patch . . . 71
5.4.1 Choosing RBF parameter ϵ . . . 71
5.5 Two Vortex Patches on a Non-rotating Sphere . . . 74
5.5.1 Initial patches . . . 75
5.5.2 Numerical integration . . . 76

VI. Solving the Barotropic Vorticity Equation by Vortex Method . . . 78

6.1 Introduction . . . 78
6.2 Lagrangian Fomulation of BVE . . . 79
6.3 Meshes on the Surface of the Sphere . . . 81
6.3.1 Longitude-latitude (LL) . . . 81
6.3.2 Icosahedral triangles (IT) . . . 82
6.3.3 Icosahedral hexagon (IH) . . . 83
6.3.4 Cubed-sphere (CS) . . . 84
6.4 Area for Panels on Sphere . . . 87
6.5 Regularized Approximation . . . 88
6.6 Rossby-Haurwitz Wave Experiments . . . 89
6.6.1 Definition of error . . . 90
6.6.2 Mesh comparison . . . 90
6.6.3 Timestep experiments . . . 90
6.6.4 Test spatial error . . . 91
6.6.5 Stream function . . . 91
6.6.6 Nonuniform mesh test . . . 92
6.7 Adaptive Mesh Refinement (AMR) . . . 94
6.8 Gaussian Vortex Tests . . . 96
6.8.1 Non-rotating sphere . . . 97
6.8.2 Rotating sphere . . . 98
6.8.3 Convergence check . . . 98
6.9 Two Gaussian Vortices Test . . . 99
6.9.1 Non-rotating sphere . . . 102
6.9.2 Rotating sphere . . . 102

6.10 Remeshing 104
6.11 Comparison of Vortex Method with GARBF Method 104

VII. Summary and Future work 107

7.1 Thesis Summary 107
7.2 Improving the Vortex/RBF Method for solving BVE 108
7.3 Shallow Water Equation (SWE) 109
7.4 Application and Extension of Treecode 109
7.5 Jet Simulation 110

BIBLIOGRAPHY **111**

LIST OF FIGURES

Figure

1.1 Hierarchy of climate models (from John Thuburn, University of Exeter). 2

1.2 A sphere rotates around the z-axis with angular velocity Ω. 3

2.1 Gaussian RBF and MQ RBF for different shape parameter ϵ. The figures show that the basis function become flatter as $\epsilon \to 0$. 10

2.2 Gaussian RBF approximation of the Runge function (2.8) on a uniform grid with $N = 41$ for different ϵ. ((a)-(d)): f (red line) and interpolation function $s(x)$ (blue $\circ$) versus x; ((e)-(h)): error versus x. Errors are decreasing first then increasing when ϵ changes from 20 to 0.5. 12

2.3 Cardinal function in equation (2.25) for different α. It equals one when $X = 0$ and zero for the other integers. 17

2.4 The shaded regions show where the absolute error in the eigenvalue of the first derivative, $K - \kappa(K; M)$, is smaller than 0.05 for four different values of M as a function of α. 22

2.5 Finite difference error divided by Gaussian RBF error in the eigenvalue of the first derivative versus K for $M = 4, 8, 16$, which is a stencil of nine, seventeen and thirty three points with different α. The thick dashed horizontal line is where the ratio is one: the radial basis function method is better whenever the ratio is above this line, and the finite difference is better whenever the curve is below this dashed line. The thin dotted line marks the right one-third of the spectrum which would be removed by a dealiasing filter in a hydrodynamics computation. 23

3.1 Complete hierarchical tree structure in two dimension for 4 levels. Level = 0 is also called the root cluster. 27

3.2 Adaptive hierarchical tree structure in two dimensions for $N_0 = 1$, where N_0 is the maximum number of particles in a leaf. 28

3.3 Example of neighbors and well separated panels for a complete tree. The target points are in the red cell, the white cells are the neighbors and the blue cells are the well separated cells of the red cell. 30

3.4 Particle cluster interaction when $r/R < \theta$, where r is the radius of the cluster and θ is MAC number. (a): Barnes and Hut [7]; R is the distance between $\boldsymbol{x}$ and the center of the cluster $\boldsymbol{y}_C$; (b): Barnes [6], R is the distance between the centers of the two clusters. 31

3.5 Example of 2d stencils for recurrence relation. Taylor coefficients at blue point only depends on the coefficients at four red points. 34

3.6 Flowchart of Barnes and Hut [7] treecode algorithm. (a) main function and (b) subroutine *compute interaction*. Note that there is a recursive call in the subroutine. 35

3.7 Schematic showing branch points, branch cuts, and domain of convergence (shaded) for far-field expansions of $\phi(x)$ given in (3.16); (a) the Laurent series (3.17) converges outside a disk in the x-plane; (b) the Taylor series (3.19) converges inside a disk in the y-plane. As c increases, the shaded region in (a) becomes smaller and the shaded region in (b) becomes larger. 36

3.8 Example in 1D from [10] showing a cluster C of nodes satisfying $|y_j| \leq h$ and well-separated evaluation points satisfying $|x_i| \geq 3h$. 38

3.9 A well separated particle-cluster interaction example for testing the convergence of Laurent expansion (3.24) and Taylor expansion (3.26). 41

3.10 Error (3.36) in a particle-cluster approximation with one evaluation point at the origin in $\mathbb{R}^3$ and a cluster of $N = 10^3$ random nodes in the cube $[0.75, 1]^3$; $\nu = 1, d = 3$. The error is plotted as a function of the RBF parameter c and order p, for $10^{-3} \leq c \leq 10^3$ and $p = 0:2:10$. (a) Laurent series (3.24), (b) Taylor series (3.29). 41

3.11 N random points in a cube (left) and on the surface of a sphere. 42

3.12 Random nodes in a cube, scatter plot of CPU time and error (3.37), data from Table 3.1, system size $N = 216$K, order $p = 0:2:10$ (increasing from right to left), MAC parameter $\theta = 0.2$ ($\circ$,black), $\theta = 0.5$ ($*$,red), $\theta = 0.8$ ($\triangle$,blue), RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$. The lower envelope of the data gives the most efficient choice of parameters (p, θ) to attain a given error. 44

3.13 Random nodes in a cube, treecode error (3.37) plotted as a function of system size $N = 10^3, 20^3, \ldots, 100^3$, order $p = 0:2:10$, MAC parameter $\theta = 0.8$, RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$ except $N_0 = 400$ for $N = 90^3, 100^3$. 45

3.14 Random nodes in a cube, treecode CPU time in seconds, same parameters as in Figure 3.13 caption, ds denotes direct sum. 45

3.15 Random nodes in a cube, treecode memory usage in MB, same parameters as in Figure 3.13 caption, ds denotes direct sum. 46

4.1 Diagram for showing the central angle between two points (θ, λ) and (θ', λ') on sphere. 52

4.2 (a) and (b) are the plot of the vorticity ζ_δ (4.46) from smooth Green's function G_δ (4.42) for $\delta = 0.2, 0.1, 0.05$, where (b) is in log scale. Both figures share the same legend. Note that ζ_δ is an approximate delta function. 57

4.3 The vorticity function ζ_ϵ (4.51) with Gaussian forcing at north pole for different ϵ. 58

4.4 Initial stream function with amplitude $A = 0.05$ and $n = 1, m = 1$. 60

4.5 Stream function from equation (4.63) at $t = \pi, 2\pi, 3\pi, 4\pi$ with amplitude $A = 0.05$ and angular velocity $\Omega = 1/2$. 62

4.6 Rossby-Haurwitz wave (n = m = 1). (a) : particle trajectories obtained by solving equations (4.72) to (4.74) using fourth order Runge-Kutta for five revolutions, where $A = 0.05$ and $\Omega = 1/2$; (b): local zoom of the left figure. 63

5.1 Illustration of icosahedral points on the surface of the sphere with (a) $n = 20$ and (b) $n = 80$. 69

5.2 Absolute error of vorticity defined by (5.18) which is obtained by GARBF after one revolution ($t = 4\pi$) with $N = 20$, $\Delta t = 4\pi/200$, $\epsilon = 0.1$. 70

5.3 Relative error defined by (5.19) of vorticity obtained by GARBF after one revolution ($t = 4\pi$) for the number of timesteps is $25, 50, 100, 200, 400$. Total number of points is 20 and $\epsilon = 0.1$. 70

5.4 (a): The relative error defined by (5.19) of vorticity obtained by GARBF with $\Delta t = 4\pi/200$, $n = 80$ for different ϵ; (b): the condition number of RBF matrix A. . 71

5.5 Interpolation points for initial vorticity; refined spherical triangular mesh (which will be discussed in detail in Chapter Six); number of points is 170, minimum distance between two points are 0.0451. 73

5.6 Absolute error of relative vorticity computed in grid in Figure 5.5. (a): fixed ϵ; (b): ϵ adoptive from (5.25). Note that the maximum error occurs where the mesh changes from coarse to fine. 74

5.7 Absolute error of relative vorticity for different $\epsilon_{\min}$. Note that the maximum error occurs where the mesh changes from coarse to fine. 75

5.8 Points in a disk on a plane. The nNIHS = 2,4,8,16. Every point is at the center of a hexagon. 76

5.9 Vortex patches evolution. the ratio of the distance between two center of the patches the radius of the patch is 2.6. The total number of the particles is 2918. (a): two vortex patches at $t = 0$; (b): $t = 2$; (c): $t = 3$; (d): close up look at $t = 3$. 77

6.1 Active points (•) and passive points (○) of a triangle and quadrilateral panel. . . . 81

6.2 Longitude-Latitude Panel. (a): side view; (b): view from north pole. Points are clustered at the pole area. 81

6.3 Icosahedral triangle mesh refinement stencils. For example, new vertices v_{12} is the center of the edge v_1v_2. The coordinate of the new center c_1 is the average of three vertices v_1, v_{12}, v_{31} by equation (6.11) then project to the sphere by equation (6.12). 83

6.4 Icosahedral triangles. From left to right: $L = 0, 1, 2, 3$ and $n = 20, 80, 320, 1280$. Vertices are projections from 12 vertices of an icosahedron to a unit sphere. 83

6.5 Icosahedral triangles with local refinement. The red and yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed. . 83

6.6 The ratio of minimum area and maximum area in Icosahedral triangle mesh. This shows the mesh becomes more uniform as the number of points increases. 84

6.7 Icosahedral hexagon. From left to right: $L = 0, 1, 2, 3$ and $n = 12, 42, 162, 642$. The vertices of the hexagon/pendagon are the centers of the spherical triangle in figure (6.4). 84

6.8 a circumscribed cube in a sphere 85

6.9 Cubed-sphere mesh refinement stencils. For example, new vertices v_{12} is the center of the edge v_1v_2. The coordinate of the new center c_1 is the average of four vertices v_1, v_{12}, v_c, v_{41} by equation (6.11) then project to the sphere by equation (6.12), where v_c is at the same position as the center c. 86

6.10 Cubed-sphere mesh with $n = 6, 24, 96, 384$. The first eight vertices of quadrilateral panels are the vertices of a circumscribed cube. 86

6.11 Cubed-sphere local refinement. The red and yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed. 86

6.12 Angles of the spherical triangles. 88

6.13 Absolute error in particle positions after one revolution for different meshes: icosahedral triangle, icosahedral hexagon and cubed-sphere for RH wave. The error is comparable for the three meshes. All three lines are parallel to the line $O(1/n)$, which means the vortex method we use here is first order accurate in terms of the number of points. 91

6.14 Plot |AbsErr − SaturationError| versus dt, which shows the 4th order accuracy of RK4 with $N = 320, 1280$. star: $N = 320$; circle: $N = 1280$ 94

6.15 RH wave: Absolute error for particle position after one revolution in spherical triangular mesh for $\Delta t = 4\pi/1600$. n is the number of the panels on sphere. . . . 95

6.16 Trajectory of particles after five revolutions with $N = 80, dt = 4\pi/1600$, which is agree to the one we obtained from exact ODE equations (4.72) to (4.74) and figure (4.6). 96

6.17 Relative error (6.23) of the stream function (6.1) using midpoint numerical integral based on spherical triangle (ST). Area means the maximum area of the spherical triangle since they are not exactly the same.) 97

6.18 Exact and approximate (6.20) stream function for RH wave at $t = 0, t = 2\pi, t = 3\pi, t = 4\pi$ with $n = 80$ and $\delta = 0.02$ 98

6.19 Meshes for $N = 120, 180, 456, 1572$ for testing the effect of mesh non-uniformity. . . 99

6.20 (a): absolute error for particle positions for uniform cubed-sphere mesh as in figure (6.10) and nonuniform cubed-sphere meshes as in figure (6.19); Right : ratio of two absolute errors defined in (6.25). 99

6.21 Gaussian patch test case 1 at $t = 0, 2\pi, 4\pi$ on a nonrotating sphere using (a) cubed-sphere mesh and (b) icosahedral triangle mesh. The patch is self-rotating and preserves the symmetric shape. 100

6.22 Gaussian patch test case 2 at $t = 0, 2\pi, 4\pi$ on a rotating sphere. (a): cubed-sphere mesh; (b) icosahedral triangle mesh. Gaussian patch moves in the northwest direction which agrees with the physical expectation. 101

6.23 Gaussian patch test case 3. From left to right, $\epsilon_\Gamma = 0.001, 0.0005, 0.0002, 0.0001$ at $t = 4\pi$. We can observe qualitative convergence property. 101

6.24 Gaussian patch test case 4 at $t = 0, \pi, 3\pi$ on a nonrotating sphere. The initial centers of two patches are $(-0.3670, 0.8548, 0.3670)$ and $(0.3670, 0.8548, 0.3670)$. The two patches rotate around each other and merge. 103

6.25 Gaussian patch test case 5 at $t = 0, \pi, 3\pi$ on a rotating sphere. The initial centers of two patches are $(-0.3670, 0.8548, 0.3670)$ and $(0.3670, 0.8548, 0.3670)$. The two patches rotate around each other and merge and both are drift towards the northwest direction. 103

6.26 Preliminary results of remeshing. Retain the relative vorticity at $L = 4$ uniform mesh with $\delta = 0.02$ using equation (6.22) then refined with $\epsilon_\Gamma = 0.0004$ and $\epsilon_d = 0.02$. (a) and (c) are the Gaussian patch test at $t = 2\pi$ and $t = 4\pi$ with $\Omega = 0, L = 4, \epsilon_\Gamma = 0.0004$ and $\epsilon_d = 0.02, \Delta t = 4\pi/1000, \delta = 0.02$; (b) and (d) are the meshs after remeshing for (a) and (c) respectively. 105

7.1 Stencils for high order refinement. (a): one active point and eight passive points in quadrilateral panel; (b): one active point and 6 passive points in triangular panel. 109

7.2 Vortex sheet evolution $t = 0, 15, 20$ and the last one is a close-up at $t = 20$. 110

LIST OF TABLES

Table

2.1 Some common types of radial functions $\phi(r)$ for RBF. There are two types of radial functions, the piecewise smooth and the infinitely smooth radial functions. All the infinitely smooth radial functions have a shape parameter ϵ. 10

2.2 Condition number of the matrix A when interpolating Runge function using Gaussian RBF on a uniform grid with $N = 41$. 13

3.1 Random nodes in a cube, the treecode error (3.37) and CPU time (sec) are given for system size $N = 216$K, RBF parameter $c = 10^{-1}$, order $p = 0 : 2 : 10$, MAC parameter $\theta = 0.2, 0.5, 0.8$, maximum leaf size $N_0 = 200$. 43

3.2 Random nodes in a cube and on a sphere, tc denotes treecode and ds denotes direct sum, CPU time in seconds and memory in MB, RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$ except $N_0 = 400$ for $N = 1000$K, order $p = 6$, MAC parameter $\theta = 0.8$. 47

5.1 Relative Error for RBF approximation on a sphere with fixed number of interpolation points and different ϵ (uniform). α is defined in equation (5.24). 73

5.2 Condition number for $\epsilon_{\min} = 1, 0.8, 0.6, 0.4$. 74

6.1 Absolute Error for particle positions after one revolution, N = 320. The error saturates around 0.004339 due to the spatial discretization. 92

6.2 Absolute error for particle positions after one revolution, N = 1280. The error saturate around 0.001244 due to the spacial discretization. 93

6.3 Absolute error for particle position after one revolution with fixed $\Delta t = 0.00785$ in icosahedral triangular mesh for different spacial resolution $N = 20, 80, 320, 1280$. . . 93

6.4 Parameters for Gaussian patch tests. CS = cubed-sphere mesh; IT = icosahedral triangle mesh; Ω = angular velocity; L = levels of the mesh; ϵ_Γ = circulation refinement criterion; ϵ_d = side length refinement criterion; $\Delta t = 4\pi/1000$ and the regulation parameter $\delta = 0.02$. 102

CHAPTER I

Introduction

This dissertation describes the three main research projects on which the author has worked in the past four years. All four projects are related to Lagrangian mesh free methods. Section 1.1 explains the motivation for writing this thesis. Section 1.2 gives an outline of the thesis.

1.1 Motivation for Thesis

The climate and weather are closely related to human life, which is the reason why researchers are trying to understand them theoretically and practically. Partial differential equations (PDE) are used to model the complex flow motions of the atmosphere and ocean. However, these equations are too complex to have an analytical solution. With the help of modern computers, researchers started to do numerical weather prediction in the 1950s [20] and long-time climate modeling later. After half a century, this area is still under development. There are a lot of open questions which are related to the PDE models, numerical algorithms and the ability to compute the fluid flow. We are interested in the numerical simulation part of this challenging field.

Basically, there exist two major forms of computational methods for fluid mechanics problems, each named after the form of the advection equations that they

use: Eulerian and Lagrangian [73]. For the former one, the grid points are fixed in time, that is, the equations track the dynamics of the fluid at fixed spatial locations. On the other hand, Lagrangian means that the dynamics is tracked at points that move with the flow. Both methods have advantages and disadvantages. Eulerian models are much more advanced in terms of modeling atmospheric and oceanic motion. For example, even though a semi-Lagrangian strategy is applied in some general circulation models (GCM), most GCMs (e.g. [4], [66] and NCAR's Community Atmospheric Model(CAM) [28]) are based on an Eulerian framework. The main motivation for this thesis is to introduce a Lagrangian method to simulate flow on a rotating sphere.

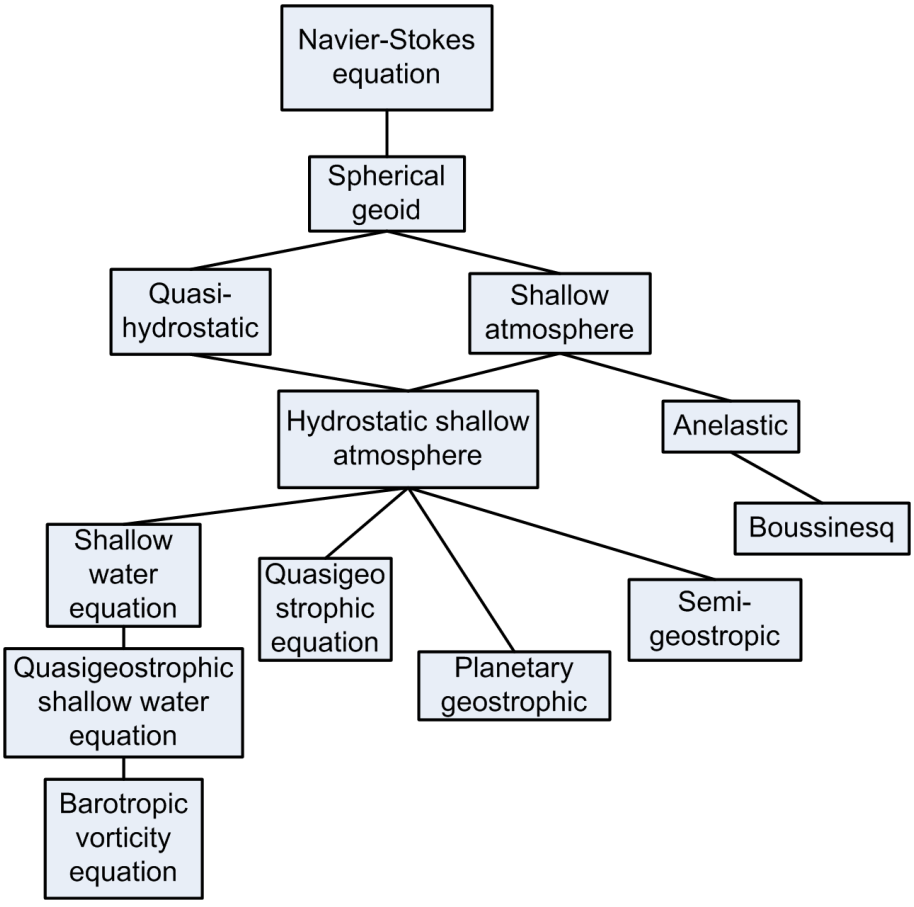


Figure 1.1: Hierarchy of climate models (from John Thuburn, University of Exeter).

Figure 1.1 shows the hierarchy of atmospheric models. It starts with the complex Navier-Stokes equation, which does not have an analytic solution. A variety of assumptions are used to simplify the Navier-Stokes equations, and different assumptions lead to different atmospheric models. The incompressible Barotropic Vorticity Equation (BVE) at the bottom of figure (1.1) is a simple mathematical model for the description of large-scale horizontal motions of the atmosphere. For theoretical investigations of the evolution of vortices, atmospheric researchers are still using the barotropic assumption [26], [71]. Therefore the BVE is a reasonable first step for investigating complex atmospheric motion. We suppose the earth rotates around the z-axis with angular velocity Ω as shown in Figure 1.2.

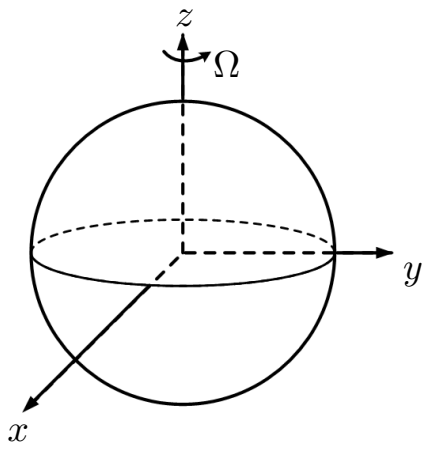


Figure 1.2: A sphere rotates around the z-axis with angular velocity Ω.

The Coriolis parameter is defined

$$f = 2\Omega \cos\theta, \tag{1.1}$$

where $\theta \in [0, \pi]$ is the colatitude, which is zero at the north pole and π at the south pole and $\lambda \in [0, 2\pi]$ is the longitude. By definition,

$$\cos\theta = z, \tag{1.2}$$

we have $f = 2\Omega z$. For incompressible flow, the divergence-free condition is

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1.3}$$

where $\boldsymbol{u}(\boldsymbol{x}, t) = (u(\boldsymbol{x}, t), v(\boldsymbol{x}, t))$ is the fluid velocity that is relative to the rotating sphere. The velocity $u(\boldsymbol{x}, t)$ and $v(\boldsymbol{x}, t)$ are the components at θ and λ direction respectively. We have

$$\nabla \times \boldsymbol{u} = \zeta \boldsymbol{e_r}, \tag{1.4}$$

where ζ is the relative vorticity and $\boldsymbol{e_r}$ is the unit vector in the radial direction. The BVE system on a rotating sphere is given by

$$\boldsymbol{u} = \nabla\psi \times \boldsymbol{x}, \tag{1.5}$$

$$\Delta_s \psi = -\zeta, \tag{1.6}$$

$$\frac{\partial \eta}{\partial t} + \boldsymbol{u} \cdot \nabla \eta = 0, \tag{1.7}$$

where $\psi(\boldsymbol{x}, t)$ is the stream function and Δ_s is the surface spherical Laplacian,

$$\Delta_s \equiv \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\lambda^2}. \tag{1.8}$$

and $\eta = \zeta + f$ is the absolute vorticity. Equation (1.5) gives a relation between velocity and stream function. Equation (1.6) is a Poisson equation on sphere, which gives a relation between stream function and the relative vorticity. Equation (1.7) shows the conservation of the absolute vorticity.

In this thesis, before we solve the BVE system, we first introduce the radial basis function method (RBF). It is one of the primary tools for interpolating multidimensional scattered data. The method can handle arbitrarily scattered data and it is easy to generalize to high dimensions. It also has a great potential as a numerical method for meshfree solution of PDEs [49], [50], [54], [32], [33]. Before we start using

RBF to solve PDEs in a Lagrangian way in Chapter Five, Chapter Two investigates several basic properties of RBF.

Computational cost is an issue related to both RBF interpolation and discretization of RBF and vortex methods. In Chapter Three, a Cartesian treecode is developed to reduce the computational cost.

The vortex method is characterized by considering the PDE in stream function-velocity form and Lagrangian discretization of the vorticity [78], [77], [63], [64], [70]. Chapter Six provides a solution for the BVE by the vortex method.

BVE is our first step to dea with flows on the sphere by a Lagrangian method. Future investigations will be discussed in Chapter Seven.

1.2 Overview of Thesis Topics

Chapter Two gives an introduction to the RBF method, including its background, advantages and related problems. The most impressive property is the trade-off between exponential convergence and ill-conditioning. We also present an RBF Cardinal function for the case of a one dimensional unbounded evenly spaced grid. At the end of this chapter, we compare the accuracy of RBF and finite difference methods using Fourier analysis. One major concern of RBF applications is the computational cost. To address this, Chapter Three presents a fast treecode for evaluating RBFs. Instead of using direct summation between each point, it applies the divide-and-conquer strategy to use particle-cluster interactions. Taylor approximation is applied as far-field expansion and shows better performance than the Laurent expansion. The treecode reduces the computational cost from $O(N^2)$ to $O(N \log N)$, where N is the number of the particles in the system. The following three chapters are applications related to fluid flow on the surface of the rotating sphere. Chap-

ter Four presents an overview of the Barotropic Vorticity Equation (BVE). Chapters Five and Six solve BVE by Gaussian RBF and Vortex method in a Lagrangian sense. The Rossby-Haurwitz wave and Gaussian patch are tested in both chapters. The last chapter contains a summary and outlook for future work.

CHAPTER II

Radial Basis Function Analysis

2.1 Introduction

Interpolation of data is a common problem in engineering and science. Suppose we have N data values f_j at points $\boldsymbol{x}_j, j = 1, ..., N$, which are obtained by sampling or experimentation, and the task is to find a function $s(\boldsymbol{x})$ which fits those data points as closely as possible. Interpolation is one way to find such functions with conditions that the functions go exactly through the data points, that is,

$$s(\boldsymbol{x}_j) = f_j, \quad j = 1, ..., N. \tag{2.1}$$

The general idea is to expand function $s(\boldsymbol{x})$ in a set of basis functions $\psi_j(\boldsymbol{x})$,

$$s(\boldsymbol{x}) = \sum_{j=1}^{N} d_j \psi_j(\boldsymbol{x}) \tag{2.2}$$

such that $s(\boldsymbol{x}_j) = f_j, j = 1, ..., N$. The expansion coefficients d_j can be obtained by solving a linear system

$$\left[A\right]\left[d\right] = \left[f\right], \tag{2.3}$$

where $A_{ij} = \psi_j(\boldsymbol{x}_i)$. This idea works well for one dimensional problems. However, for data in higher dimensions, this is not always the case. Haar's theorem [60] shows that for any set of basis functions $\psi_j(\boldsymbol{x})$, there exist sets of distinct data points

$\boldsymbol{x}_j, j = 1, ..., N$ in $\mathbb{R}^D, D \geq 2$ such that the linear system of equations (2.3) for coefficients d_j becomes singular. This difficulty can be bypassed by taking the basis function radially symmetric about its center [62],

$$s(\boldsymbol{x}) = \sum_{j=1}^{N} d_j \phi(||\boldsymbol{x} - \boldsymbol{x}_j||), \tag{2.4}$$

that is, using a single basis function that depends on the data set $\boldsymbol{x}_j$. This is referred to as the radial basis function (RBF) method.

The RBF method was first introduced by Hardy(1971) [45] for the multiquadric (MQ) radial function

$$\phi(r) = \sqrt{1 + \epsilon^2 r^2}, \tag{2.5}$$

where ϵ is a shape parameter. The method was used to solve a cartography problem, where approximate topography and contour lines were needed by interpolation of sparse and scattered data. There was no other interpolation method which offered a satisfactory result and furthermore, as mentioned before, the non-singularity of the interpolation matrix was not guaranteed [60]. Franke (1982) studied various methods and found the MQ RBF method overall to be the best one [39]. He also conjectured the unconditional non-singularity of the interpolation matrix associated with the multiquadric radial function, but it was not until a few years later that Micchelli [62] was able to prove it as mentioned above.

The main feature of the MQ method is that the interpolant is a linear combination of translations of a basis function which only depends on the Euclidean distance from its center. This basis function is therefore radially symmetric with respect to its center. That is how its name radial basis function comes about. The MQ method was generalized to other radial functions, such as the thin plate spline [27], the Gaussian, the cubic, etc. In the 1990s researchers became to pay attention to

the RBF method again when Kansa (1990) introduced a way to use it for solving parabolic, elliptic and (viscously damped) hyperbolic PDEs [49, 50]. His method consisted of approximating spatial partial derivatives by differentiating smooth RBF interpolants to solve parabolic, elliptic, and viscously damped hyperbolic PDEs to spectral accuracy, in a completely mesh-free manner [49, 50]. Flyer and Wright [32] solved advection equations on a sphere by RBF. They found that RBF offers larger integration time step than traditional methods with a simpler implementation. We are also interested in solving PDEs by RBF method. Chaper Five will discuss solving Barotropic Vorticity Equation (BVE) on the surface of a sphere using Gaussian RBF in detail. Some work of this chapter is published in [16, 17].

2.2 RBFs Accuracy and Stability

Here we discuss the accuracy and stability issues that related to RBF method. A radial basis function interpolant takes the form in equation (2.4), let's repeat it here,

$$s(\boldsymbol{x}) = \sum_{j=1}^{N} d_j \phi(||\boldsymbol{x} - \boldsymbol{x}_j||), \tag{2.6}$$

when we interpolate data values f_i at the scattered node locations $\boldsymbol{x}_i, i = 1, 2, ...N$ in d dimensions, and where $||\cdot||$ denotes the Euclidean 2-norm.

We obtain the expansion coefficients d_j by solving a linear system

$$\underbrace{\begin{pmatrix} \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_1||) & \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_2||) & \cdots & \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_N||) \\ \phi(||\boldsymbol{x}_2 - \boldsymbol{x}_1||) & \phi(||\boldsymbol{x}_2 - \boldsymbol{x}_2||) & \cdots & \phi(||\boldsymbol{x}_2 - \boldsymbol{x}_N||) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(||\boldsymbol{x}_1 - \boldsymbol{x}_N||) & \phi(||\boldsymbol{x}_N - \boldsymbol{x}_2||) & \cdots & \phi(||\boldsymbol{x}_N - \boldsymbol{x}_N||) \end{pmatrix}}_{A} \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}, \tag{2.7}$$

which is based on the interpolation conditions $s(\boldsymbol{x}_i) = f_i$. There are two types of

radial functions, the piecewise smooth and the infinitely smooth radial functions, as shown in Table 2.1. All the infinitely smooth radial functions have a shape parameter ϵ. The closer this parameter is to zero, the flatter the radial function becomes, as in Figure 2.1 which shows the Gaussian and Multiquadric as examples.

Radial function $\phi(r;\epsilon)$	Name	smoothness
$r^{2n-1}, n = 1, 2, 3, ...$	Powers (linear, cubic,...)	piecewise smooth
$r^{2n} \ln r, n = 1, 2, 3, ...$	Thin Plate Splines (TPS)	
$\frac{1}{1+(\epsilon r)^2}$	Inverse Quadratic (IQ)	infinitely smooth
$\sqrt{1+(\epsilon r)^2}$	Multiquadric (MQ)	
$\exp\left(-(\epsilon r)^2\right)$	Gaussian (GA)	

Table 2.1: Some common types of radial functions $\phi(r)$ for RBF. There are two types of radial functions, the piecewise smooth and the infinitely smooth radial functions. All the infinitely smooth radial functions have a shape parameter ϵ.

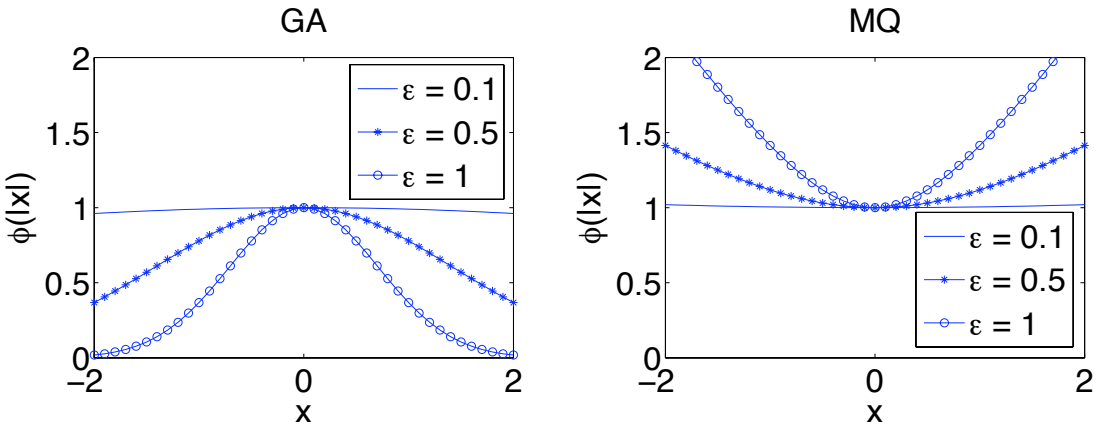


Figure 2.1: Gaussian RBF and MQ RBF for different shape parameter ϵ. The figures show that the basis function become flatter as $\epsilon \to 0$.

For a numerical method, we are interested in several properties such as accuracy, stability and cost. Here, we discuss the first two properties and then afterwards, we'll consider the third one. Now, we show the RBF accuracy and stability properties by

using interpolation of Runge function,

$$f(x) = \frac{1}{1+x^2}, \tag{2.8}$$

as an example on a one dimensional uniform grid. The basic features of this example can be extended to high dimensional scattered data. The infinitely smooth radial basis functions exhibit better approximation properties than the piecewise smooth ones [34]. The accuracy of the infinitely smooth radial functions also depends on the shape parameter ϵ and can be improved by increasing the flatness of the radial function. Figure 2.2 shows Gaussian RBF interpolation for the Runge function (2.8) when ϵ changes from 20 to 0.5. The left column shows both the function $f(x)$ in red and the interpolation function $s(x)$ in blue circles. The right column shows the absolute error $|f(x) - s(x)|$. Note that the error is decreasing from $\epsilon = 20$ to $\epsilon = 10$ and $\epsilon = 1$. The reason for this improvement is that when ϵ is small, the basis functions are flat and have a lot of overlaps. However, the error increases as ϵ decrease from $\epsilon = 1$ to $\epsilon = 0.5$. The reason is that the difference between each basis function is small due to the flatness of the basis. The condition number shown in Table 2.2 is large for small ϵ. This introduces a large computational error when the RBF coefficients d_j are computed by solving the linear system (2.7). Hence, there is a trade-off between accuracy and conditioning for small ϵ and researchers are showing great interest in this regime. Fornberg and his collaborators developed "Contour Padé" method and "RBF-QR" method to overcome the ill-conditioning and find RBF offers exponential convergence when ϵ is close to zero [36, 37].

2.3 Computational cost

Computational cost is another issue linked to using radial basis functions. Smooth radial functions are global, thus the interpolation matrices A are dense. The system

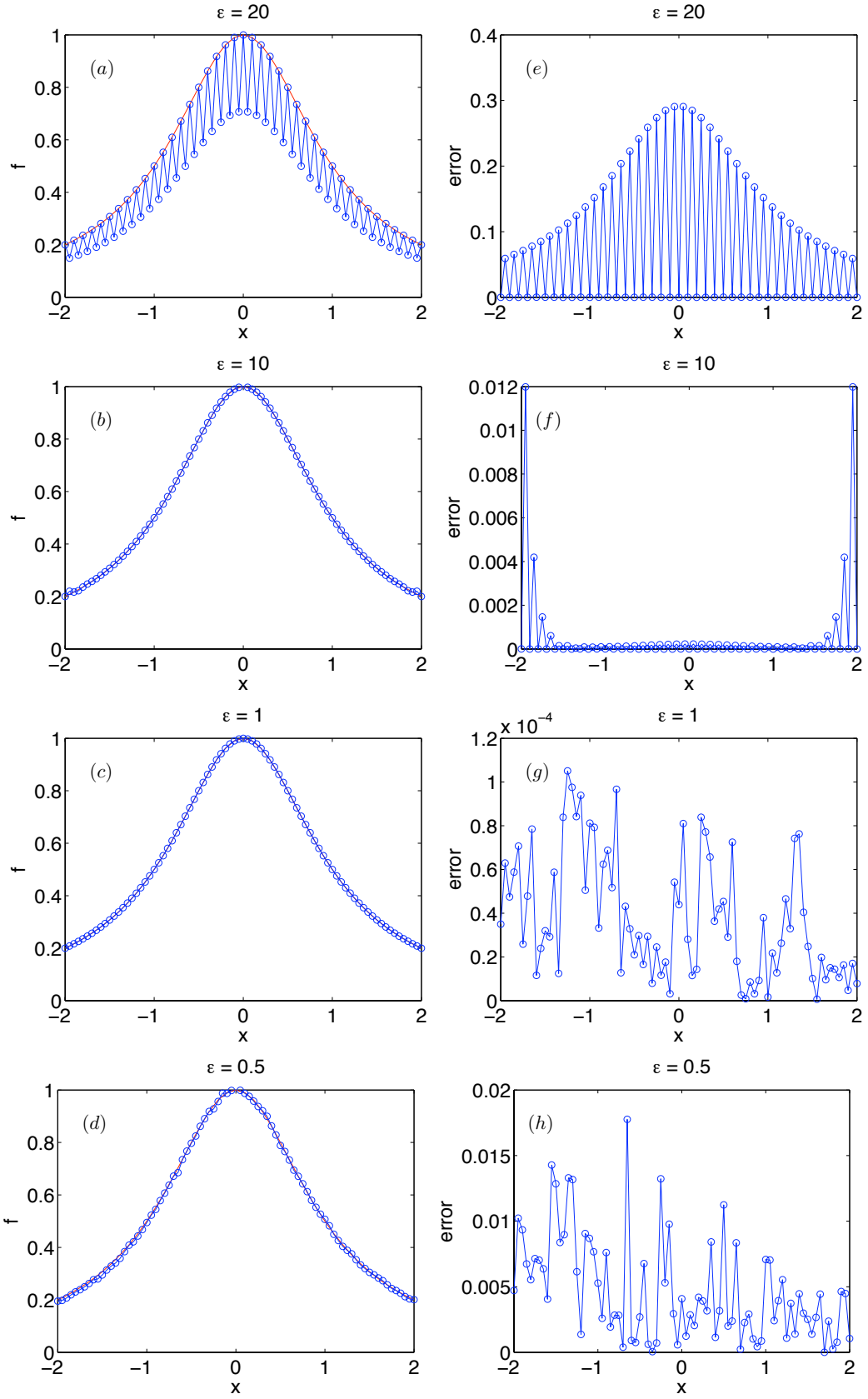


Figure 2.2: Gaussian RBF approximation of the Runge function (2.8) on a uniform grid with $N = 41$ for different ϵ. ((a)-(d)): f (red line) and interpolation function $s(x)$ (blue $\circ$) versus x; ((e)-(h)): error versus x. Errors are decreasing first then increasing when ϵ changes from 20 to 0.5.

ϵ	cond(A)
20	1.0758
10	5.8559
1	2.1038e+018
0.5	4.4236e+018

Table 2.2: Condition number of the matrix A when interpolating Runge function using Gaussian RBF on a uniform grid with $N = 41$.

requires $O(N^3)$ operations to solve for the RBF coefficients d_j by a direct method such as LU factorization. Even though for iterative methods such as GMRES, the solution involves matrix-vector multiplication with operation count $O(N^2)$ which is still prohibitively expensive when N is large. However, there exist several fast methods for performing the dense matrix-vector multiplication, which reduce the operation count, such as the fast multiple method (FMM) [42, 43], treecode algorithm [6, 7] and other choices such as multilevel summation [44, 59]. We will discuss this in detail in Chapter Three. Next, we will be focus on the basic formulas for Gaussian RBF.

2.4 Basic Formulas for Gaussian RBF

RBF is impressive for its ability to handle problems in scattered data and high dimensions. However, it is not easy to do theoretical analysis of RBF in these cases. Here and in the next sections, we will be focused on the Gaussian RBF for a one dimensional evenly spaced grid, which can at least help us understand this methodology. Here we rewrite the Gaussian RBF in a slightly different way,

$$\phi(x) \equiv \exp(-[\alpha^2/h^2]x^2), \tag{2.9}$$

where $\alpha = \epsilon h$ and h is the grid spacing. More formally, we can always make the change of variable

$$X = x/h, \tag{2.10}$$

and the RBF approximation with grid spacing h in x is converted into one with the same coefficients but unit grid spacing in X. One Reason for choosing the Gaussian is that it is infinitely differentiable and such RBFs are more accurate than RBFs of finite smoothness. In addition, Gaussian functions allow many theoretical simplifications as illustrated below.

2.4.1 Computing the coefficients of an RBF expansion

RBF coefficients d_j are usually obtained by solving a linear system. However, there is another way to solve for the coefficients [34], [35]. Suppose we have a basis function $\phi(X)$ and its Fourier transform is

$$\Phi(K) \equiv \int_{-\infty}^{\infty} \phi(X) \exp(iKX) dX. \tag{2.11}$$

Now consider a function $f(X)$ approximated by RBFs on a uniform grid,

$$f(X) = \sum_{j=-\infty}^{\infty} d_j \phi(X - j). \tag{2.12}$$

Then the RBF coefficients can be computed by,

$$d_j = \frac{1}{2\pi} \int_{-\pi}^{\pi} \psi(Y) \exp(ijY) dY, \tag{2.13}$$

where

$$\psi(Y) = \frac{\sum_{n=-\infty}^{n=\infty} f(n) \exp(-inY)}{\sum_{n=-\infty}^{n=\infty} \phi(n) \exp(-inY)} = \frac{\sum_{m=-\infty}^{\infty} F(-Y + 2\pi m)}{\sum_{m=-\infty}^{\infty} \Phi(-Y + 2\pi m)}, \tag{2.14}$$

where F is the Fourier transform of the function $f(X)$.

2.4.2 Poisson summation

There is an important relation between a function and its Fourier transform on a evenly spaced grid.

Theorem II.1. (Poisson Summation) *If $g(X)$ and $G(K)$ are a function and its Fourier transform*

$$G(k) \equiv \int_{\infty}^{\infty} g(X) \exp(iKX) dX, \tag{2.15}$$

$$g(X) = (1/2\pi) \int_{-\infty}^{\infty} G(K) \exp(-iKX) dK, \tag{2.16}$$

then for any positive constant q, it follows that

$$q \sum_{n=-\infty}^{\infty} g(qn) \exp(inqx) = \sum_{m=-\infty}^{\infty} G(x - m\frac{2\pi}{q}). \tag{2.17}$$

Furthermore, we have

$$\sum_{n=-\infty}^{\infty} (-1)^n g(n) = \sum_{m=-\infty}^{\infty} G([2m+1]\pi). \tag{2.18}$$

Equation (2.17) shows a periodic function can be represented as a series of identical but translated copies of the Fourier transform of the function $g(x)$ [16].

2.5 Cardinal Function

One way to look into the intrinsic properties of RBFs and bypass both the ill-conditioning and efficiency issues is to rearrange the RBF basis into the Lagrange Cardinal basis,

$$C_j(x_k) = \begin{cases} 1, & j = k, \\ 0, & j \neq k. \end{cases}$$

This gives an explicit, matrix-free solution to the interpolation problem,

$$f(x) = \sum_{k=1}^{n} f(x_j) C_j(x). \tag{2.19}$$

The cardinal function basis is much cheaper than the equivalent Gaussian RBF interpolation, which forms a dense matrix. We developed an explicit expression for the cardinal basis function for Gaussian RBF interpolation on a uniform grid [16].

2.5.1 Derivation

On a uniform grid, all $C_j(x)$ are translations of a master basis function $C_j(x) = C(x - jh)$. All dependence on the (uniform) grid spacing h can be removed by the change of variable $X = x/h$. The RBF approximation with grid spacing h in x is converted into one with the same coefficients but unit grid spacing in X. Buhmann [19] showed that, although there is no simple form for the cardinal function itself, there is a remarkable formula for the Fourier Transform of an RBF cardinal function. Define the Fourier Transform of the RBF function $\phi(x)$ by equation (2.11). Buhmann proved that, not just for Gaussians but for RBF in general, the Fourier Transform of $C(X)$ is

$$\hat{C}(K) = \Phi(K) / \sum_{m=-\infty}^{\infty} \Phi(K - 2\pi m). \tag{2.20}$$

For Gaussian RBFs, this yields

$$\hat{C}(K) = \frac{\exp(-K^2/(4\alpha^2))}{\sum_{m=-\infty}^{\infty} \exp\left(-\frac{(K-2\pi m)^2}{4\pi^2}\right)} = \frac{1}{\sum_{m=-\infty}^{\infty} \exp\left(-\frac{\pi^2}{\alpha^2} m^2\right) \exp\left(\frac{\pi m}{\alpha^2} K\right)}. \tag{2.21}$$

We are more interested in the RBF properties for small α since the RBF approximation error is unacceptably big when $\alpha > 1$ [16]. Hence, for small α, we approximate the infinite series in (2.21) by three terms, the $m = 0, -1, 1$ terms, and obtain a function whose inverse Fourier Transform can be explicitly computed from [41],

$$\hat{C}_a(K) = \frac{1}{1 + 2\exp\left(-\frac{\pi^2}{\alpha^2}\right) \cosh\left(\frac{\pi}{\alpha^2} K\right)}. \tag{2.22}$$

We have

(2.23)
$$C(X) \approx \frac{\alpha^2}{\pi\sqrt{1-4\exp\left(-2\frac{\pi^2}{\alpha^2}\right)}} \sin\left(\frac{\alpha^2 X}{\pi}\text{arccosh}\left(\frac{1}{2}\exp\left(\frac{\pi^2}{\alpha^2}\right)\right)\right)\frac{1}{\sinh(\alpha^2 X)}.$$

Neglecting terms of $O(\exp(-2\pi^2/\alpha^2))$ and using the lowest term in the arccosh series

$$\text{arccosh}\left(\frac{1}{2}\exp(\frac{\pi^2}{\alpha^2})\right) \approx \frac{\pi^2}{\alpha^2} - \exp(-2\frac{\pi}{\alpha^2}) + \cdots, \tag{2.24}$$

we have

$$C(X) \approx \frac{\alpha^2}{\pi}\frac{\sin(\pi X)}{\sinh(\alpha^2 X)}. \tag{2.25}$$

It is obvious that

$$C(0) \approx \frac{\alpha^2}{\pi}\frac{\pi\cos(\pi X)}{\alpha^2\cosh(\pi X)}|_{X=0} = 1, \tag{2.26}$$

by L'Hospital's rule. Figure 2.3 shows $C(X)$ on the interval $[-10, 10]$ with several small values of α.

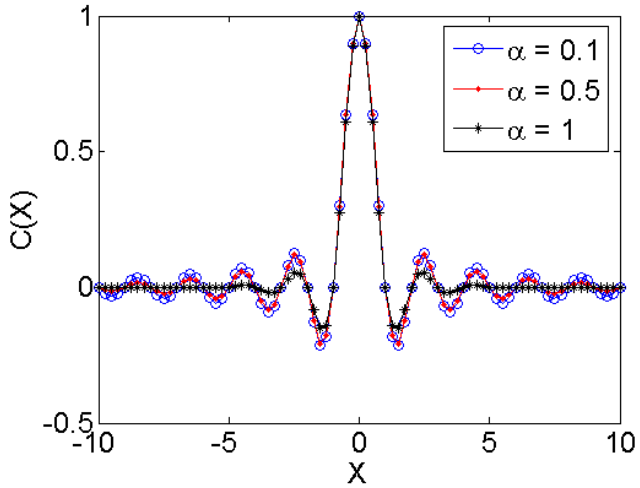


Figure 2.3: Cardinal function in equation (2.25) for different α. It equals one when $X = 0$ and zero for the other integers.

The approximation can be written alternatively as

$$C(X) \approx \frac{\alpha^2}{\pi}\frac{\sin(\pi X)}{\sinh(\alpha^2 X)} = \frac{\alpha^2 X}{\sinh(\alpha^2 X)}\text{sinc(X)}, \tag{2.27}$$

since $\mathrm{sinc(X)} = \frac{\sin(\pi \mathrm{X})}{\pi \mathrm{X}}$. Note that $\lim_{X \to 0} X/\sinh(X) = 1$, we have

$$\lim_{\alpha \to 0} \frac{\alpha^2}{\pi} \frac{\sin(\pi X)}{\sinh(\alpha^2 X)} = \mathrm{sinc(X)}, \tag{2.28}$$

which confirms that the RBF cardinal function reduces to the sinc function as $\alpha \to 0$. This cardinal function formula can be generalized to high dimensions.

Theorem II.2. (Uniform grid multidimensional cardinal functions)
On a uniform grid in any number of dimensions d, the Gaussian RBF cardinal function is the direct product of one-dimensional cardinal functions,

$$C^d(x_1, x_2, , ..., x_d) = \prod_{m=1}^{d} C(x_m). \tag{2.29}$$

The general cardinal function is just the translate of the master cardinal function, i.e., in two dimensions

$$C^2_{jk}(x, y) = C(x - jh)C(y - kh). \tag{2.30}$$

Proof. The direct product trivially satisfies the multidimensional cardinal function condition. To show that this product is in the space spanned by d-dimensional Gaussian RBFs, observe that by its very definition, the cardinal function is an exact sum of basis functions. Thus, setting $h = 1$ for simplicity, $C(X) = \sum_{j=-\infty}^{\infty} p_j \exp(\alpha^2(X - j)^2)$ for some coefficients p_j whose exact values are irrelevant. Therefore,

$$C^{(d)}(X_1, X_2, ..., X_d) = \prod_{m=1}^{d} \sum_{j_m=-\infty}^{\infty} p_{j_m} \exp(-\alpha^2(X_m - j_m)^2). \tag{2.31}$$

The identity $\exp(a)\exp(b) = \exp(a + b)$ implies that

$$\prod_{m=1}^{d} \exp(-\alpha^2(X_m - j_m)^2) = \exp(-\alpha^2 ||X - X_j||_2). \tag{2.32}$$

Thus, each term is a d-dimensional RBF. □

This concludes our discussion of RBF cardinal functions. Next, we will compare the RBF with finite difference method using Fourier Analysis.

2.6 Comparison to Finite Differences Using Fourier Analysis

In addition to interpolation, we are interested in RBF approximation for derivatives and solving PDEs. A finite difference (FD) formula can be obtained by differentiating a polynomial interpolation and a standard way for assessing the accuracy of differentiation formula is Fourier analysis. For RBF, we can treat it in the similar way, that is, obtain an approximation by differentiating the RBF interpolant $s(\boldsymbol{x})$. Here, we will focus on Gaussian RBF and a basic Fourier mode $\exp(iKX)$ in a one-dimensional unbounded domain with uniform grid distribution. We compare the accuracy of this RBF derivative and FD derivative using Fourier analysis. We start with the investigation of the Fourier mode $\exp(iKX)$.

2.6.1 Eigenvalues of the RBF difference operators for $\exp(iKX)$

Differentiating $\exp(iKX)$ analytically yield

$$\frac{d}{dX}\exp(iKX) = iK\exp(iKX). \tag{2.33}$$

Ignoring the imaginary factor "i", denote the eigenvalue of the differential operator by

$$\kappa_{\text{exact}}(K) = K. \tag{2.34}$$

For a second-order centered FD approximation,

$$\frac{d}{dX}\exp(iKX) \approx \frac{1}{2}\left(\exp(iK(X+1)) - \exp(iK(X-1))\right) = i\sin K\exp(iKX), \tag{2.35}$$

that is,

$$\kappa_{\text{FD}_2}(K) = \sin K. \tag{2.36}$$

Suppose we are looking for the difference formula that approximates the derivative at $X = 0$ by

$$\frac{df}{dX}(X=0) \approx \sum_{m=-M}^{M} \omega_m f(m). \tag{2.37}$$

Then for $f(m) = \exp(imX)$, assume $\omega_m = -\omega_m$ which is true for all RBF and centered FD approximation,

$$\kappa_{\mathrm{FD}} = \sum_{m=1}^{M} 2\omega_m \sin(mK). \tag{2.38}$$

Fornberg and Flyer [34], using the formula for d_j given by (2.13) and (2.14), show that for a radial basis function on a uniform grid, the eigenvalue is

$$\kappa_{\mathrm{RBF}}(K) = \frac{\sum_{m=-\infty}^{\infty}(K-2\pi m)\Phi(-K+2\pi m)}{\sum_{m=-\infty}^{\infty}\Phi(-K+2\pi m)}, \tag{2.39}$$

where Φ is the Fourier transform of the basis function ϕ, as in (2.11). So for Gaussian RBFs in particular,

$$\kappa_{\mathrm{GARBF}}(K) = \frac{\sum_{m=-\infty}^{\infty}(K-2\pi m)\exp(-(K-2\pi m)^2/(4\alpha^2))}{\sum_{m=-\infty}^{\infty}\exp(-(K-2\pi m)^2/(4\alpha^2))}. \tag{2.40}$$

Note that, $\kappa_{\mathrm{RBF}}(K)$ is the ratio of two infinite series. It is hard to use this formula directly but we can rewrite it using Jacobian theta function [2, 12],

$$\theta_3(y = K/2; q = \exp(-\alpha^2)) = \frac{\sqrt{\pi}}{\alpha} \sum_{m=-\infty}^{\infty} \exp\left(-\frac{(K-2\pi m)^2}{4\alpha^2}\right), \tag{2.41}$$

where θ_3 is the usual Jacobian theta function and q is the "elliptic nome" and

$$\frac{d}{dy}\log(\theta_3)(y;q) = 4\sum_{n=1}^{\infty}(-1)^n \frac{q^n}{1-q^{2n}}\sin(2ny). \tag{2.42}$$

We have

$$\kappa_{\mathrm{GARBF}}(K) = -2\alpha^2 \frac{\frac{d}{dK}\theta_3(y = K/2; q = \exp(-\alpha^2))}{\theta_3(y = K/2; q = \exp(-\alpha^2))} \tag{2.43}$$

$$= -2\alpha^2 \frac{d}{dK}\left(\log(\theta(y = K/2; q = \exp(-\alpha^2)))\right) \tag{2.44}$$

$$= 4\alpha^2 \sum_{n=1}^{\infty} (-1)^{n+1} \frac{\exp(-\alpha^2 n)}{1 - \exp(-2\alpha^2 n)} \sin(nK) \tag{2.45}$$

$$= \sum_{n=1}^{\infty} (-1)^{n+1} \frac{2\alpha^2}{\sinh(\alpha^2 n)} \sin(nK). \tag{2.46}$$

Because the coefficients of the Fourier series for κ in equation (2.38) are also twice the weights of the difference formula (2.37), we can truncate the above series (2.46) to M terms to obtain the approximate eigenvalue of the Gaussian RBF when the sum over all points on the grid is truncated to $(2M + 1)$ terms:

$$\kappa_{\mathrm{GARBF}}(K, M) = \sum_{m=1}^{M} (-1)^{m+1} \frac{2\alpha^2}{\sinh(\alpha^2 m)} \sinh(mK). \tag{2.47}$$

This means that the differentiation weights for RBF are given

$$\omega_m = (-1)^{m+1} \frac{\alpha^2}{\sinh(\alpha^2 m)}. \tag{2.48}$$

Hence from (2.46), the truncated RBF approximation for $f = \exp(iKX)$ is

$$\frac{df}{dX}(X) \approx \sum_{m=1}^{M} (-1)^{m+1} \frac{\alpha^2}{\sinh(\alpha^2 m)} \left(f(X + m) - f(X - m)\right). \tag{2.49}$$

2.6.2 Numerical experiments

A differentiation formula is quite useless if it gives a poor approximation to the differentiation eigenvalue. Somewhat arbitrarily, we have chosen an absolute error of 0.05 as a threshold of minimum acceptability, and graphed the performance of truncated Gaussian RBF differentiation formulas in the $K-\alpha$ plane for four different stencil widths in Figure 2.4. It shows that the area in which the error is less than 0.05 is increasing as M increases. However, the results are unsatisfactory for $\alpha < 1$,

which is the regime we are interested in for RBF interpolation. For example, the area that the error is less than 0.05 is quite small even for a 9 point-stencil.

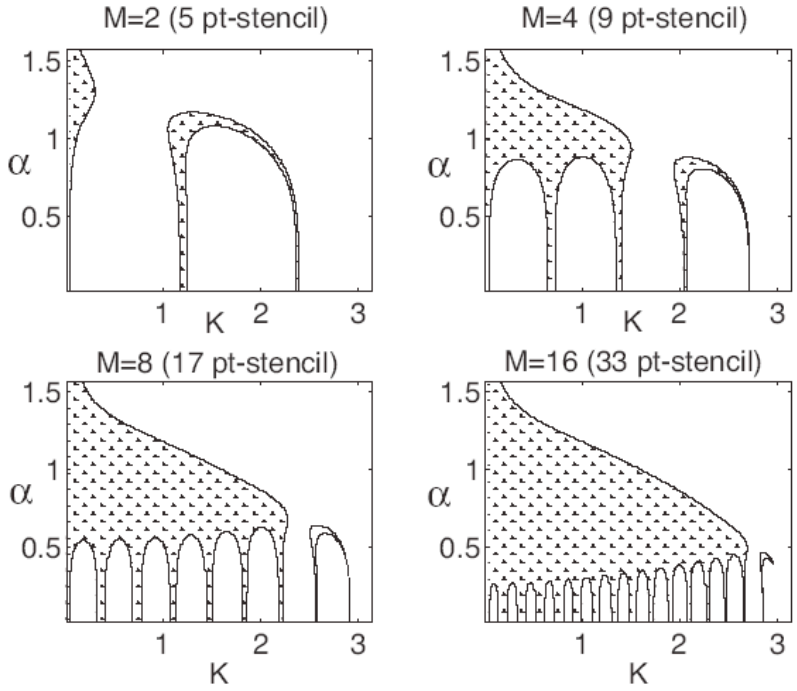


Figure 2.4: The shaded regions show where the absolute error in the eigenvalue of the first derivative, $K - \kappa(K; M)$, is smaller than 0.05 for four different values of M as a function of α.

Figure (2.5 plots the ratio of the finite difference errors to the RBF errors in the eigenvalue of the first derivative $\kappa(K)$ for nine-point, seventeen-point and thirty-three point stencils. The pattern is quite consistent between different orders M: the RBF method is better for K near the aliasing limit, but much worse by a huge factor for small K. For the thirty-three point stencil, the region of RBF superiority lies wholly in the right one-third of the wavenumber range. In realistic calculations, these Fourier components would likely be corrupted by aliasing error and in fact are completely eliminated by a dealiasing filter of the sort common in fluid mechanics [15].

These numerical tests show that the truncated RBF approximation (2.49) which

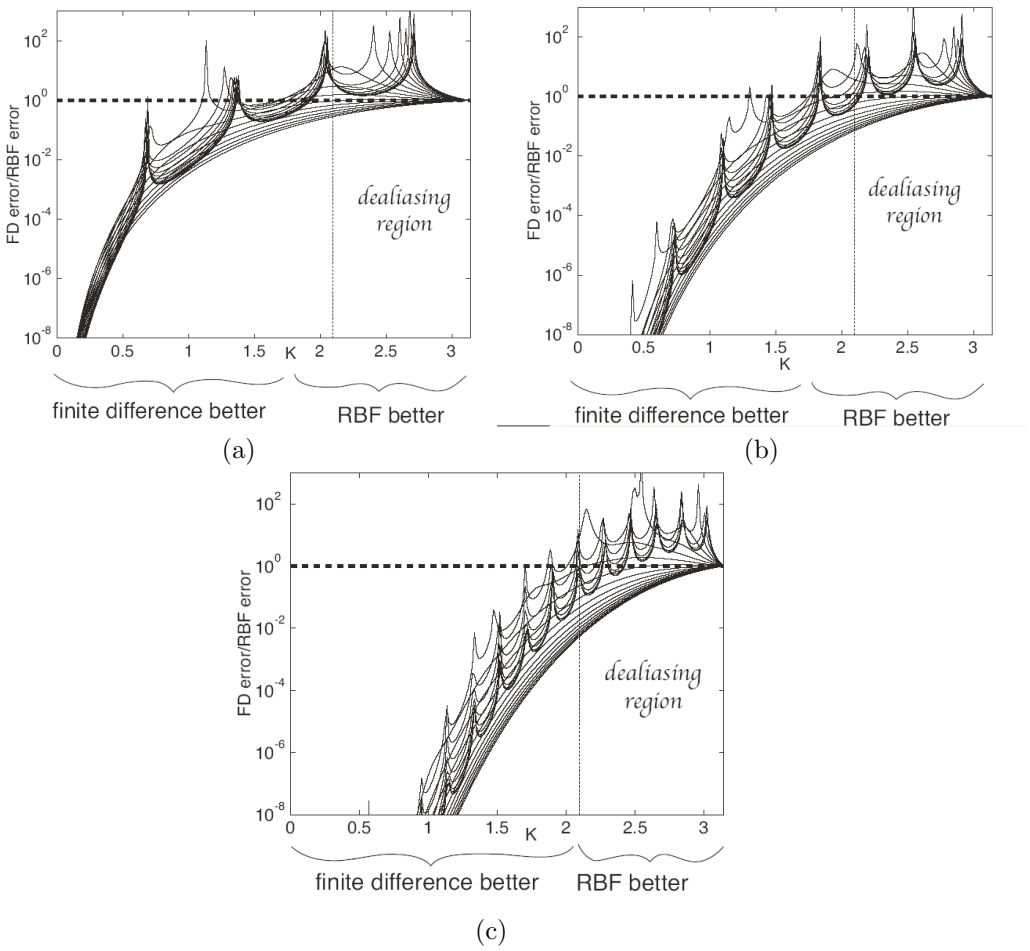


Figure 2.5: Finite difference error divided by Gaussian RBF error in the eigenvalue of the first derivative versus K for $M = 4, 8, 16$, which is a stencil of nine, seventeen and thirty three points with different α. The thick dashed horizontal line is where the ratio is one: the radial basis function method is better whenever the ratio is above this line, and the finite difference is better whenever the curve is below this dashed line. The thin dotted line marks the right one-third of the spectrum which would be removed by a dealiasing filter in a hydrodynamics computation.

is obtained using the similar idea as FD is inferior than FD in terms of approximating the Fourier mode $\exp(iKX)$.

In this chapter, we give an introduction to the RBF method, including its background, advantage and related problems. The most significant property of RBF approximation is the trade-off between exponential convergence and ill-conditioning. We also present an RBF cardinal function for Gaussian in one dimensional unbounded evenly spaced grid. At the end, we compare the accuracy of RBF method

and Finite Differences method in terms of Fourier analysis. We find that the truncated Gaussian RBF method is inferior to the FD for differentiating the function $f(X) = \exp(iKX)$, where K is the wavenumber.

CHAPTER III

Fast Treecode for Evaluating RBFs

3.1 Introduction

As we mentioned in Chapter Two, one of the major concerns in the application of RBFs is the computational cost. There are two expensive steps in RBF approximation. The first step is to calculate the RBF coefficients d_j by solving a linear system (2.7). The operation count for solving a dense linear system by LU decomposition is $O(N^3)$. Other choices like iterative methods such as GMRES, involve matrix-vector multiplication with operation count $O(N^2)$. The second step is to evaluate the RBF summations,

$$s(\boldsymbol{x}_i) = \sum_{j=1}^{N} d_j \phi(\boldsymbol{x}_i - \boldsymbol{y}_j),\ i = 1 : M, \tag{3.1}$$

where $\boldsymbol{y}_j$ are interpolation points and $\boldsymbol{x}_i$ are evaluation points. Note that compared to (2.6), equation (3.1) is rewritten with a slight abuse of notation. Equation (3.1) can be viewed as a matrix-vector multiplication with operation count $O(NM)$. Both steps are prohibitively expensive when N, M are large. Hence, the bottleneck of the computational cost is the matrix-vector multiplication. There are several existing fast evaluation algorithms including the treecode algorithm of Barnes and Hut [7] and the Fast Multipole Method (FMM) of Greengard and Rokhlin [43] and others for different RBF kernels [44, 59, 11]. Some of the work in this chapter is published

in [53]. For a given accuracy, the treecode requires $O(N \log N)$ operations and the FMM requires $O(N)$ operations. Both Treecode and FMM divide the particles into a hierarchy of clusters having a tree structure and they approximate particle-cluster or cluster-cluster interactions instead of particle-particle interactions. The Treecode algorithm is more advantageous in terms of algorithm complexity and memory usage. For simplicity, we consider speeding up the N-body system

$$s(\boldsymbol{x}_i) = \sum_{j=1}^{N} d_j \phi(\boldsymbol{x}_i - \boldsymbol{x}_j),\ i = 1 : N, \tag{3.2}$$

by treecode algorithm. That is, target points are the same as source points. Note that it is not a restriction for the treecode.

3.2 Build Tree

There are two ways to construct a hierarchical tree. The first choice is using the maximum number of levels (L). A complete tree is constructed as follows:

1. The collection of particles is enclosed with a rectangular box, which becomes the root cell of the tree. Set the level of current cell p_1 to be $l = 0$;

2. Subdivide current cell p_i into subcells (children) by bisecting p_i in the two/three coordinate directions. The result is four children in two dimensions and eight children in three dimensions. Label the level number of each child $l = l + 1$;

3. If l is less then L, apply step 2 to each child of p_i.

Figure 3.1 shows a tree for $L = 3$ in 2 dimensions. Note that every leaf is in the same level. a leaf is a cell which doesn't have children.

The other way to construct a tree ensures that every leaf contains fewer particles than a parameter N_0. The tree is constructed as follows:

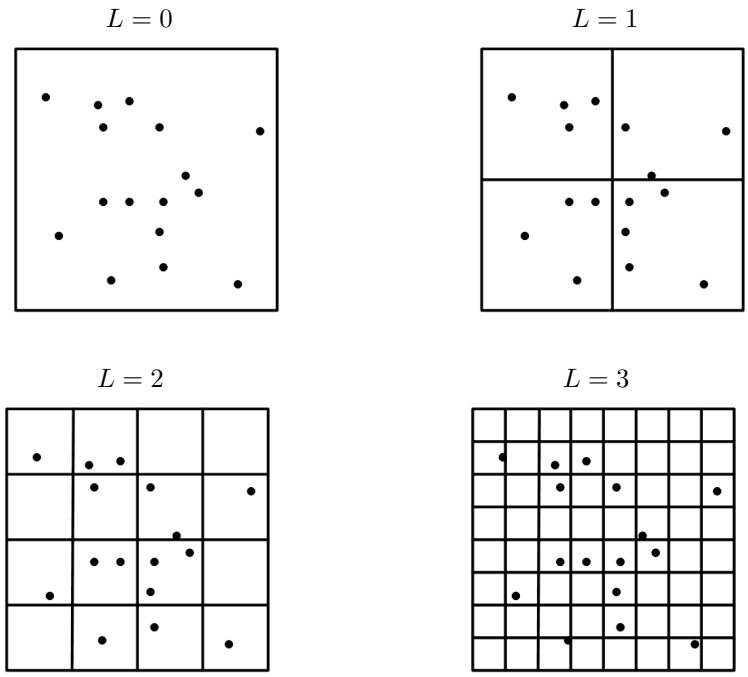


Figure 3.1: Complete hierarchical tree structure in two dimension for 4 levels. Level = 0 is also called the root cluster.

1. The collection of particles is enclosed with a rectangular box, which becomes the root panel p_1 of the tree;

2. If the current panel p_i contains fewer than N_0 particles then exit. The panel is a leaf of the tree.

3. Otherwise, subdivide current cell p_i into subpanels (children) by bisecting p_i in the coordinate directions and apply step 2 to each children.

Figure 3.2 shows the structure of tree for $N_0 = 1$. Compare to the complete tree, this version is more adapted to the distribution of particles and each leaf can be at a different level of the tree.

We notice that both L and N_0 controls the depth of the tree. They affect the performance of the algorithm. If N_0 is too small, then the tree will have many levels, leading to a large memory requirement. However, if N_0 is too large, then the tree

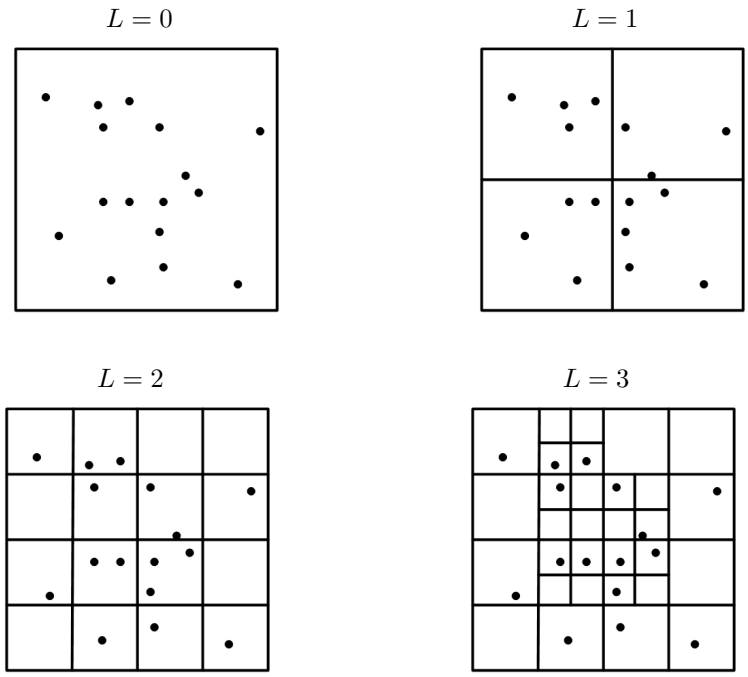


Figure 3.2: Adaptive hierarchical tree structure in two dimensions for $N_0 = 1$, where N_0 is the maximum number of particles in a leaf.

consists of cells having large spatial dimensions that evaluate many particle-particle interaction by direct summation and the efficiency may be affected.

3.3 Particle-Cluster Interaction

In this section, we will introduce the particle-cluster interaction. Barnes and Hut (1986) [7] presented a hierarchical treecode method for calculating the force on N bodies with operation count $O(N \log N)$, where particle-cluster interactions were performed by approximating the cluster as if all particles in the cluster are located at the cluster's center of mass. A drawback of this approximation is its low accuracy. The Fast Multipole Method of Greengard and Rokhlin (1987) [43] overcame this obstacle by using a series expansion to approximate particle-cluster interactions to any specified tolerance. They also introduced cluster-cluster interactions by expanding the far-field approximation into a local near-field expansion for rapid evaluation

at multiple target points. The operation account is $O(N)$. However the algorithm is very complicated. Lindsay and Krasny (2001) [58] developed a Barnes-Hut type treecode with Taylor series as far-field expansion, which improved the accuracy of the algorithm and requires $O(N \log N)$ operations. Here we will focus on a treecode with Cartesian Taylor series as far field expansion. One of the major steps of the treecode algorithm is to find a well-separated particle-cluster interaction list. But how is a well-separated particle-cluster interaction defined?

3.3.1 Well-separated particle-cluster interaction list 1

Points $\boldsymbol{x}$ are said to be well-separated from cluster p if $\boldsymbol{x}$ are separated from the panel p by at least the diameter of p [10]. For the complete tree structure, Figure 3.3 shows a way to find a well separated particle-cluster interaction list. Suppose the target points are in the red cell, then the white cells are the neighbors of the red cell, they are in the same level and share a edge. The direct summation will be performed between the particles in the target panel and the particles in the neighbor panels. The blue cells are the well separated ones. Notice that some of them are in the same level of the target panel and some are in the parents' level of target cell. Since the level number of parents is always smaller than the level number of the children, it is better to choose smaller level clusters as long as they are well-separated.

3.3.2 Well-separated particle-cluster interaction list 2

For the adaptive tree structure shown in Figure 3.2, Barnes and Hut present the following approach (Figure 3.4(a)). For a given target point, start from the root panel in the tree.

1. evaluate the distance between the target point and the center of the current panel, denote as R, the radius of the current panel , denote as r;

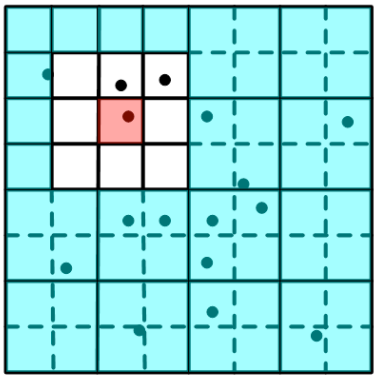


Figure 3.3: Example of neighbors and well separated panels for a complete tree. The target points are in the red cell, the white cells are the neighbors and the blue cells are the well separated cells of the red cell.

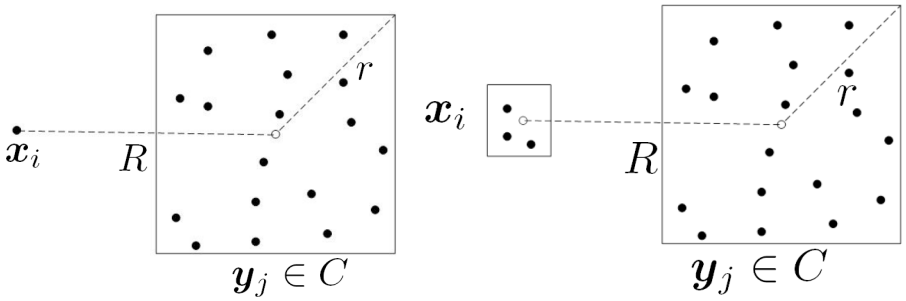
2. if the ratio r/R is less than a user specified number (MAC) θ, the current panel is a well-separated panel for the target point;

3. otherwise, if current panel is a leaf, then it is a panel to which direct summation will be applied, otherwise apply step 2 for each children of current panel.

Barnes [6] presented a modified treecode. Instead of defining an interaction list for each target point, he built an interaction list for each leaf. All the particles in the same leaf share the same interaction list, Figure 3.4(b). For Barnes [6], the target cells are the leafs of the tree and R is the distance between the center of the target cell and current cell.

Experiments show that the performance of Barnes [6] is better than Barnes and Hut [7] and the neighbor idea in the complete tree shown in Figure 3.3. We use Barnes [6] in our implementation of the treecode algorithm.

For the leaf panels that don't satisfy the MAC condition, direct summation will be applied.



(a) (b)

Figure 3.4: Particle cluster interaction when $r/R < \theta$, where r is the radius of the cluster and θ is MAC number. (a): Barnes and Hut [7]; R is the distance between $\boldsymbol{x}$ and the center of the cluster $\boldsymbol{y}_C$; (b): Barnes [6], R is the distance between the centers of the two clusters.

3.3.3 Far field expansion

This section explains the far-field expansion for a well-separated particle-cluster interaction. The flow chat for the entire treecode will be presented later. Using Cartesian coordinates and standard multi-index notation, equation (3.2) yields:

$$s(\boldsymbol{x}_i) = \sum_{j=1}^{N} d_j \phi(\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{3.3}$$

$$= \sum_{C} \sum_{\boldsymbol{y}_j \in C} d_j \phi(\boldsymbol{x}_i - \boldsymbol{y}_j) \tag{3.4}$$

$$= \sum_{C} \sum_{\boldsymbol{y}_j \in C} d_j \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i - \boldsymbol{y}_C)(\boldsymbol{y}_C - \boldsymbol{y}_j)^{\boldsymbol{k}} \tag{3.5}$$

$$= \sum_{C} \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i - \boldsymbol{y}_C) \sum_{\boldsymbol{y}_j \in C} d_j (\boldsymbol{y}_C - \boldsymbol{y}_j)^{\boldsymbol{k}} \tag{3.6}$$

$$\approx \sum_{C} \sum_{||\boldsymbol{k}||=0}^{p} a_{\boldsymbol{k}}(\boldsymbol{x}_i - \boldsymbol{y}_C)\, m_{\boldsymbol{k}}(C), \tag{3.7}$$

where C are far interaction cells for target point $\boldsymbol{x}_i$. The equation (3.5) shows that a far-field expansion of the basis ϕ is taken at $\boldsymbol{y}_C$, which is the center of the cell C. The $\boldsymbol{k} = (k_1; k_2; k_3)$ is an integer multi-index with all $k_i > 0$ in three dimensions,

$||\boldsymbol{k}|| = k_1 + k_2 + k_3$, $\boldsymbol{k}! = k_1!k_2!k_3!$. We call

$$a_{\boldsymbol{k}}(\boldsymbol{x}_i, \boldsymbol{y}_C) = \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i - \boldsymbol{y}_C) \tag{3.8}$$

the kth order Taylor coefficient of $\phi(\boldsymbol{x} - \boldsymbol{y})$ at $\boldsymbol{y} = \boldsymbol{y}_C$ and

$$m_{\boldsymbol{k}}(C) = \sum_{\boldsymbol{y}_j \in C} d_j (\boldsymbol{y}_C - \boldsymbol{y}_j)^{\boldsymbol{k}} \tag{3.9}$$

is the kth moment of cluster C about its center $\boldsymbol{y}_C$. Note that in (3.6), we change the order of the summation. The cluster moment (3.9) doesn't depend on the target points. (it depends only on the cluster), so it can be calculated once after the tree is constructed.

3.4 Recurrence Relation for Taylor Series

For the algorithm to be computationally efficient, it is necessary to rapidly compute the Taylor coefficients $a_{\mathbf{k}}(\boldsymbol{x}_i, \boldsymbol{y}_C) = \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i, \boldsymbol{y}_C)$ in (3.8). We derive the recurrence relations for several radial basis functions in Table 2.1. Note that here we use a slightly different notation for the basis function, that is, instead of using ϵ as shape parameter as shown in Table 2.1, We use $c \sim \epsilon^{-1}$. For example, the Multiquadric is $\phi(\boldsymbol{x} - \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \boldsymbol{y}) + c^2}$.

3.4.1 Multiquadric

Theorem III.1. *The Taylor coefficients $a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}) = a_{(k_1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y})$ of the Multiquadric basis function $\phi(\boldsymbol{x} - \boldsymbol{y}) = \sqrt{|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2}$ satisfy the recurrence relation*

$$||\boldsymbol{k}||(|\boldsymbol{x}-\boldsymbol{y}|^2+c^2)a_{\boldsymbol{k}} - (2||\boldsymbol{k}||-3)\sum_{i=1}^{3}(x_i-y_i)a_{\boldsymbol{k}-\mathbf{e}_i} + (||\boldsymbol{k}||-3)\sum_{i=1}^{3} a_{\boldsymbol{k}-2\boldsymbol{e}_i} = 0 \tag{3.10}$$

for $||\boldsymbol{k}|| \geq 1$, *where* $a_{\mathbf{0}} = \phi(\boldsymbol{x}, \boldsymbol{y})$, $a_{\boldsymbol{k}} = 0$ *if any* $k_i < 0$, $\boldsymbol{e}_i$ *is the Cartesian unit vectors and*

$$\boldsymbol{k} - \boldsymbol{e_i} = \begin{cases} (k_1 - 1, k_2, k_3), & i = 1, \\ (k_1, k_2 - 1, k_3), & i = 2, \\ (k_1, k_2, k_3 - 1), & i = 3. \end{cases}$$

Proof. The Multiquadric basis function $\phi(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2}$ satisfies the differential equation

$$(x_1 - y_1) + \sqrt{|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2} D_{y_1} \phi = 0 \tag{3.11}$$

Applying the operator $D_{y_1}^{k_1 - 1}$ and using Leibniz's rule for differentiating a product we obtain

$$(|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2) D_{y_1}^{k_1} \phi + (3 - 2k_1)(x_1 - y_1) D_{y_1}^{k_1 - 1} \phi + (k_1 - 3)(k_1 - 1) D_{y_1}^{k_1 - 2} \phi = 0. \tag{3.12}$$

Next we apply $D_{y_2}^{k_2} D_{y_3}^{k_3}$ and substitute the definitions of $a_{\boldsymbol{k}}$ to obtain

$$k_1(|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2) a_{\boldsymbol{k}} - 2k_1 \sum_{i=1}^{3} a_{\boldsymbol{k} - \mathbf{e}_i} + 3(x_i - y_i) a_{\boldsymbol{k} - \boldsymbol{e}_1} + k_1 \sum_{i=1}^{3} a_{\boldsymbol{k} - 2\boldsymbol{e}_i} - 3a_{\boldsymbol{k} - 2\boldsymbol{e}_1} = 0. \tag{3.13}$$

Equation (3.13) is a recurrence relation for $a_{\boldsymbol{k}}$ in which the index 1 plays a special role. Similar equations can be obtained for indices 2 and 3. Summing these equations, we obtain (3.10). □

3.4.2 Gaussian

Theorem III.2. *The Taylor coefficients* $a_{\boldsymbol{k}}$ *of Gaussian basis function* $\phi(\boldsymbol{x} - \boldsymbol{y}) = \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2 / c^2)$ *satisfy the recurrence relation*

$$c||\boldsymbol{k}|| a_{\boldsymbol{k}} - 2 \sum_{i=1}^{3} (x_i - y_i) a_{\boldsymbol{k} - \boldsymbol{e}_i} + 2 \sum_{i=1}^{3} a_{\boldsymbol{k} - 2\boldsymbol{e}_i} = 0. \tag{3.14}$$

3.4.3 Inverse multiquadric

Theorem III.3. *The Taylor coefficients* $a_{\boldsymbol{k}}$ *of Inverse Multiquadric basis function* $\phi(\boldsymbol{x}-\boldsymbol{y}) = \frac{1}{\sqrt{|\boldsymbol{x}-\boldsymbol{y}|^2+c^2}}$ *satisfy the recurrence relation*

$$(3.15)\quad ||\boldsymbol{k}||(|\boldsymbol{x}-\boldsymbol{y}|^2+c^2)a_{\boldsymbol{k}}+(1-2||\boldsymbol{k}||)\sum_{i=1}^{3}(x_i-y_i)a_{\boldsymbol{k}-\boldsymbol{e}_i}+(||\boldsymbol{k}||-1)\sum_{i=1}^{3}a_{\boldsymbol{k}-2\boldsymbol{e}_i}=0.$$

We omit the proofs of (3.14) and (3.15) here since they are almost the same as the proof of Multiquadric recurrence relation (3.10). With the help of these recurrence relations, the Taylor coefficients $a_{\boldsymbol{k}}$ can be calculated very efficiently. Figure 3.5 shows how recurrence relations work in two dimensions, i.e. $\boldsymbol{k} = (k_1, k_2)$. Suppose we need to calculate the $a_{(k_1,k_2)}$ at the blue dot position, we only need the value of a at four red dot positions which have already been calculated and stored.

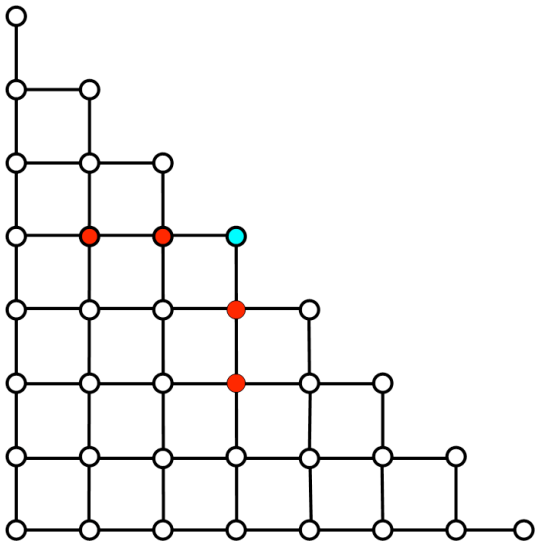


Figure 3.5: Example of 2d stencils for recurrence relation. Taylor coefficients at blue point only depends on the coefficients at four red points.

3.5 Treecode Algorithm

Figure 3.6 is the flowchart for Barnes and Hut [7] treecode algorithm. The main function (3.6(a)) starts with inputting RBF nodes $\boldsymbol{x}_i$, coefficients d_j, treecode parameters, which include the MAC number θ, Taylor approximation order p and maximum

leaf size N_0. θ is the number which controls where we will use particle-cluster interaction. Then we construct the tree according to the position of RBF nodes $\boldsymbol{x}_i$. Next, for every particle, starting from the root cluster, compute the particle-cluster interaction. The role of the subroutine *compute interaction* (3.6 (b)) is to compute the particle-cluster interaction. If the current target particle and current cluster are well separated, then calculate the interaction using Taylor approximation, otherwise, check the children of the current cluster to see whether they are well separated from target point. There is a recursive call in the subroutine which makes the program easy to implement.

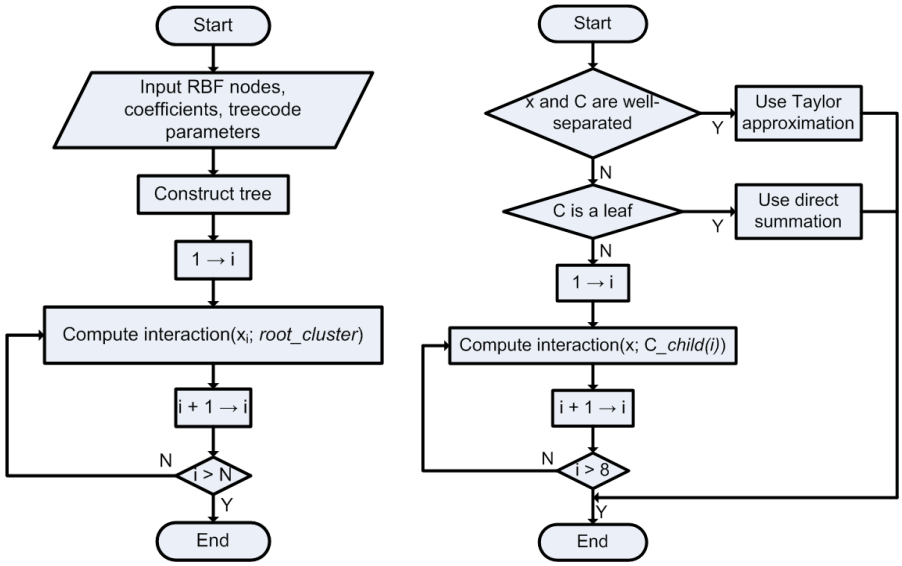


Figure 3.6: Flowchart of Barnes and Hut [7] treecode algorithm. (a) main function and (b) subroutine *compute interaction*. Note that there is a recursive call in the subroutine.

3.6 Cartesian Taylor Treecode for Multiquadric RBF

As we mentioned in Chapter Two, the shape parameter ϵ plays an important role in RBF approximation. Since $c \sim 1/\epsilon$, c controls the shape of the basis functions also. When c is large, the basis functions overlap, the RBF approximation shows exponential convergence but on the other hand, the difference between each row of

the RBF matrix $A_{i,j} = \phi(\boldsymbol{x}_i - \boldsymbol{x}_j)$ is small, the condition number of A is large. Researchers in this field are very interested in large c regime. Here, we present a treecode for evaluating Multiquadric RBF using Taylor series. We show that for the MQRBF case, the Taylor series converges for all RBF parameter values $c > 0$, in contrast to the Laurent series which converges for a limited range of c as presented in literature [10, 22].

3.6.1 Far field expansion for multiquadric in 1D

The multiquadric function in 1D is:

$$\phi(x-y) = \sqrt{(x-y)^2 + c^2}. \tag{3.16}$$

Beatson and Greengard [10] proposed the Laurent series in x for (3.16)

$$\phi(x-y) = \sqrt{|x-y|^2 + c^2} = \text{sign}(x) \cdot \left(x - y + \frac{c^2}{2x} + \frac{c^2 y}{2x^2} + \cdots \right) \tag{3.17}$$

Note that in the complex x plane for $\phi(x-y)$, the two branch points are $y + ic$ and $y - ic$. The shaded area outside of the unit circle in Figure 3.7 is the convergence area for the Laurent series. The convergence criterion for (3.17) is

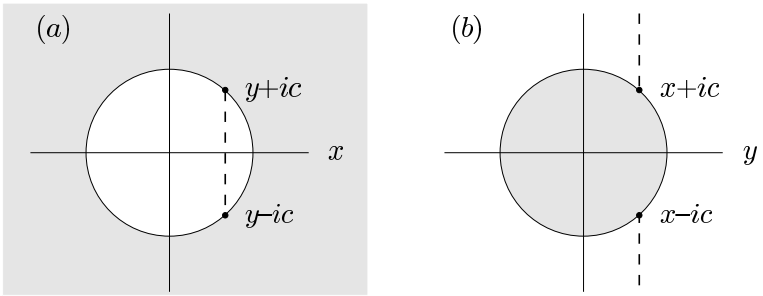


Figure 3.7: Schematic showing branch points, branch cuts, and domain of convergence (shaded) for far-field expansions of $\phi(x)$ given in (3.16); (a) the Laurent series (3.17) converges outside a disk in the x-plane; (b) the Taylor series (3.19) converges inside a disk in the y-plane. As c increases, the shaded region in (a) becomes smaller and the shaded region in (b) becomes larger.

$$\rho_1 \equiv \frac{\sqrt{y^2 + c^2}}{|x|} < 1. \tag{3.18}$$

Note that in (3.18), the RBF parameter c is in the numerator. Assuming $|x| > |y|$, the convergence criterion (3.18) restricts the value of c that can be used. Now, consider the Multiquadric (3.16) as a function of y and x is a parameter. Taking Taylor series with respect to y at $y = 0$, we have

$$\phi(x-y) = x_c - \frac{xy}{x_c} + \frac{c^2y^2}{2x_c^3} + \frac{c^2xy^3}{2x_c^5} + \frac{c^2(4x^2-c^2)y^4}{8x_c^7} + \cdots, \tag{3.19}$$

where $x_c = \sqrt{x^2+c^2}$. Then the Multiquadric (3.16) has two branch points $y = x+ic$ and $y = x-ic$ in the complex y plane. Figure 3.7 (b) shows the convergence area for Taylor series (3.19) is the shaded area inside the circle. The convergence criterion for (3.19) is

$$\rho_2 \equiv \frac{|y|}{\sqrt{x^2+c^2}} < 1. \tag{3.20}$$

Note that the RBF parameter c in (3.20) is in the denominator. So for $|x| > |y|$, the series (3.19) is uniformly converging for $c > 0$. Furthermore, the convergence rate improves as c increases.

Here we consider an example from in one space dimension [10]. Figure 3.8 shows that a cluster C is a line interval with radius h, all particles $y_i \in C$ satisfy $|y_i| < h$. All evaluation points are $|x_i| > 3h$. Evaluation points and the cluster C are separated by at least one diameter of C. Let $c = h$ as in [10], the convergence rates are

$$\rho_1 = \max_{i,j} \frac{\sqrt{y_j^2+c^2}}{|x_i|} \le \frac{\sqrt{h^2+h^2}}{3h} = \frac{\sqrt{2}}{3} = 0.47, \tag{3.21}$$

$$\rho_2 = \max_{i,j} \frac{|y_j|}{\sqrt{x_i^2+c^2}} \le \frac{h}{\sqrt{9h^2+h^2}} = \frac{1}{\sqrt{10}} = 0.32. \tag{3.22}$$

We can see that $\rho_2 < \rho_1$, which means that the Taylor series (3.19) converges faster than the Laurent series (3.17). Again, increasing c accelerates the convergence rate of the Taylor series (3.19).

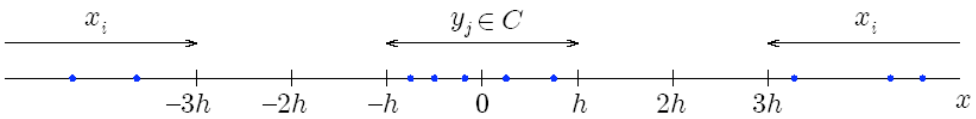


Figure 3.8: Example in 1D from [10] showing a cluster C of nodes satisfying $|y_j| \leq h$ and well-separated evaluation points satisfying $|x_i| \geq 3h$.

3.6.2 The generalized multiquadric in muti-D

Generalized multiquadric is defined:

$$\phi(\boldsymbol{x}-\boldsymbol{y}) = (|\boldsymbol{x}-\boldsymbol{y}|^2 + c^2)^{\nu/2}, \tag{3.23}$$

where $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and $\nu \in \mathbb{R}$ is an odd number. As we did for the one dimension case, there is a Laurent series in $\boldsymbol{x}$,

$$\phi(\boldsymbol{x}-\boldsymbol{y}) = (|\boldsymbol{x}-\boldsymbol{y}|^2 + c^2)^{\nu/2} = \sum_{l=0}^{\infty} \frac{P_l^{(\nu)}(|\boldsymbol{y}|^2 + c^2, -2\langle \boldsymbol{x}, \boldsymbol{y}\rangle, |\boldsymbol{x}|^2)}{|\boldsymbol{x}|^{2l-\nu}}, \tag{3.24}$$

where $P_l^{(\nu)}$ is a multivariate polynomial

$$P_l^{(\nu)}(b_1, b_2, b_3) = \sum_{j=\lfloor \frac{l+1}{2} \rfloor}^{l} \binom{\nu/2}{j} \binom{j}{l-j} b_2^{2j-l} (b_1 b_3)^{l-j} \tag{3.25}$$

for $l \geq 0$ and $P_l^{(\nu)}(b_1, b_2, b_3) = 0$ for $l < 0$ [22]. The Laurent series (3.24) converges for $|\boldsymbol{x}| > \sqrt{|\boldsymbol{y}|^2 + c^2}$. There is also a Taylor series in $\boldsymbol{x}$

$$\phi(\boldsymbol{x}-\boldsymbol{y}) = (|\boldsymbol{x}-\boldsymbol{y}|^2 + c^2)^{\nu/2} = \sum_{l=0}^{\infty} \frac{P_l^{(\nu)}(|\boldsymbol{x}|^2, -2\langle \boldsymbol{x}, \boldsymbol{y}\rangle, |\boldsymbol{y}|^2 + c^2)}{(|\boldsymbol{y}|^2 + c^2)^{(2l-\nu)/2}}, \tag{3.26}$$

which converges for $|\boldsymbol{x}| < \sqrt{|\boldsymbol{y}|^2 + c^2}$. Cherrie, Beatson and Newsam (2002) [22] developed a FMM using the Laurent series (3.24) for the far-field expansion and the Taylor series (3.26) for the near-field expansion.

We are interested in a Taylor series in $\boldsymbol{y}$. Let us interchange $\boldsymbol{x}$ and $\boldsymbol{y}$ in Taylor series (3.26) and have

$$\phi(\boldsymbol{x}-\boldsymbol{y}) = (|\boldsymbol{x}-\boldsymbol{y}|^2 + c^2)^{\nu/2} = \sum_{l=0}^{\infty} \frac{P_l^{(\nu)}(|\boldsymbol{y}|^2, -2\langle \boldsymbol{x}, \boldsymbol{y}\rangle, |\boldsymbol{x}|^2 + c^2)}{(|\boldsymbol{x}|^2 + c^2)^{(2l-\nu)/2}}, \tag{3.27}$$

which converges for $|\boldsymbol{y}| < \sqrt{|\boldsymbol{x}|^2 + c^2}$. This is a Taylor series in $\boldsymbol{y}$. We rewrite it by collecting like powers of $\boldsymbol{y}$ to have a standard Taylor expansion with respect to $\boldsymbol{y}$ at $\boldsymbol{y} = 0$

$$\phi(\boldsymbol{x} - \boldsymbol{y}) = (|\boldsymbol{x} - \boldsymbol{y}|^2 + c^2)^{\nu/2} = \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D^{\boldsymbol{k}} \phi(\boldsymbol{x}) \, (-\boldsymbol{y})^{\boldsymbol{k}}, \tag{3.28}$$

where $\boldsymbol{k} = (k_1, \ldots, k_d)$, $k_i \geq 0$ are integers, $||\boldsymbol{k}|| = k_1 + \cdots + k_d$, $\boldsymbol{k}! = k_1! \cdots k_d!$, $\boldsymbol{y}^{\boldsymbol{k}} = y_1^{k_1} \cdots y_d^{k_d}$, and $D^{\boldsymbol{k}} = D_1^{k_1} \cdots D_d^{k_d}$.

The far-field expansion will be applied for well separated particle cluster interactions. We take the Taylor expansion at the center of cluster $\boldsymbol{y}_C$,

$$\phi(\boldsymbol{x} - \boldsymbol{y}) = (|\boldsymbol{x} - \boldsymbol{y}|)^{\nu/2} = \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D^{\boldsymbol{k}} \phi(\boldsymbol{x} - \boldsymbol{y}_C) \, (-(\boldsymbol{y} - \boldsymbol{y}_C))^{\boldsymbol{k}}, \tag{3.29}$$

which converges for $|\boldsymbol{y} - \boldsymbol{y}_C| < \sqrt{|\boldsymbol{x} - \boldsymbol{y}_C|^2 + c^2}$. Then as we did before

$$s(\boldsymbol{x}_i) = \sum_{j=1}^{N} d_j \phi(\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{3.30}$$

$$= \sum_{C} \sum_{\boldsymbol{y}_j \in C} d_j \phi(\boldsymbol{x}_i - \boldsymbol{y}_j) \tag{3.31}$$

$$= \sum_{C} \sum_{\boldsymbol{y}_j \in C} d_j \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i - \boldsymbol{y}_C)(\boldsymbol{y}_C - \boldsymbol{y}_j)^{\boldsymbol{k}} \tag{3.32}$$

$$= \sum_{C} \sum_{||\boldsymbol{k}||=0}^{\infty} \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \phi(\boldsymbol{x}_i - \boldsymbol{y}_C) \sum_{\boldsymbol{y}_j \in C} d_j (\boldsymbol{y}_C - \boldsymbol{y}_j)^{\boldsymbol{k}} \tag{3.33}$$

$$\approx \sum_{C} \sum_{||\boldsymbol{k}||=0}^{p} a_{\boldsymbol{k}}(\boldsymbol{x}_i - \boldsymbol{y}_C) \, m_{\boldsymbol{k}}(C) \tag{3.34}$$

where $a_{\boldsymbol{k}}$ are kth order Taylor coefficient and $m_{\boldsymbol{k}}$ are the cluster moments as mentioned in section 3.3.3. In section 3.4, we derived the recurrence relation for Taylor coefficients for multiquadric for $\nu = 1$ and $d = 3$. Here is the recurrence relation for all odd $\nu \in \mathbb{R}$,

$$a_{\boldsymbol{k}} + \frac{2(||\boldsymbol{k}|| - 1) - \nu}{||\boldsymbol{k}||(|\boldsymbol{x}_i - \boldsymbol{y}_C|^2 + c^2)} \sum_{j=1}^{3} ((\boldsymbol{x}_i)_j - (\boldsymbol{y}_C)_j) a_{\boldsymbol{k} - \boldsymbol{e}_j} + \frac{||\boldsymbol{k}|| - 2 - \nu}{||\boldsymbol{k}||(|\boldsymbol{x}_i - \boldsymbol{y}_C|^2 + c^2)} \sum_{j=1}^{3} a_{\boldsymbol{k} - 2\boldsymbol{e}_j} = 0. \tag{3.35}$$

The proof is similar to (3.10). For $||\boldsymbol{k}|| = 0, 1$ the coefficients $a_{\boldsymbol{k}}$ are computed explicitly and then (3.10) is used to compute the coefficients for $||\boldsymbol{k}| \geq 2$.

Next we compare the errors in the Laurent series and the Taylor series for a particle-cluster interaction in the case $\nu = 1, d = 3$, in Figure 3.9. The particle (red) is located at the origin in $\mathbb{R}^3$, $\boldsymbol{x} = 0$, and the cluster C consists of $N = 10^3$ nodes distributed randomly in the cube $[0.75, 1]^3$. They are well separated since the distance between the cluster and the particle is larger than the diameter of the cluster. The RBF coefficients are set to $d_i = 1$ as in [22]. The error is defined by

$$E_1 = \frac{|s(\boldsymbol{x}, C) - \tilde{s}(\boldsymbol{x}, C)|}{|s(\boldsymbol{x}, C)|}, \tag{3.36}$$

where $s(\mathbf{x}, C)$ is the exact particle-cluster interaction(3.2) and $\tilde{s}(\boldsymbol{x}, C)$ is the approximation obtained by truncating the expansion for $\phi(\boldsymbol{x}_i - \boldsymbol{y}_j)$ using the Laurent series (3.24) or the Taylor series (3.29). For each approximation, the necessary series coefficients were computed using the recurrence relations discussed above.

Figure 3.10 shows the error as a function of the RBF parameter c and order p, for $10^{-3} \leq c \leq 10^3$ and $p = 0:2:10$. As noted above, the Laurent series converges on an interval $0 \leq c \leq \bar{c}$ and diverges for $c > \bar{c}$, while the Taylor series converges for all $c \geq 0$. For some special values of p and c, the error drops to zero; this occurs near $c = 10^{-1}$ for the Laurent series and near $c = 1$ for the Taylor series. For small values of c, the two series have comparable errors. For a given order p, the rate of convergence of the Taylor series improves as $c \to \infty$. These results confirm the favorable convergence properties of the far-field Taylor series (3.29).

3.6.3 Treecode performance

The treecode was implemented in the C programming language and the computations were performed on an Intel Pentium M processor (1.5 GHz, 768 MB RAM,1024

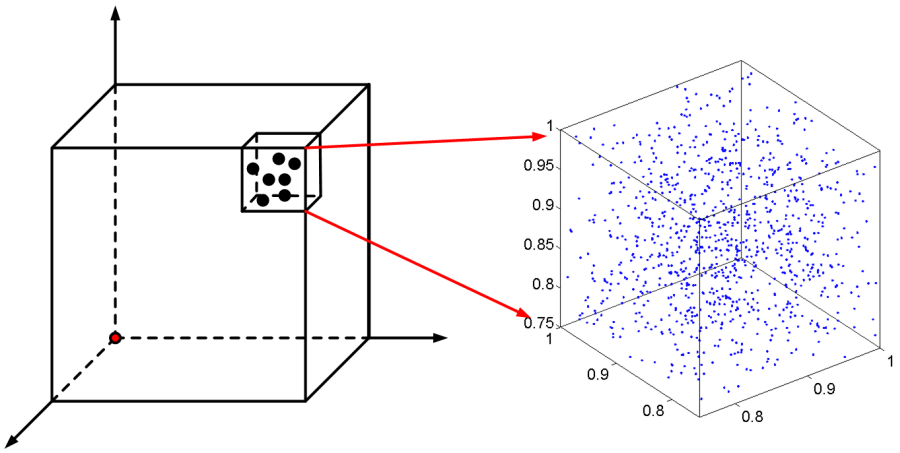


Figure 3.9: A well separated particle-cluster interaction example for testing the convergence of Laurent expansion (3.24) and Taylor expansion (3.26).

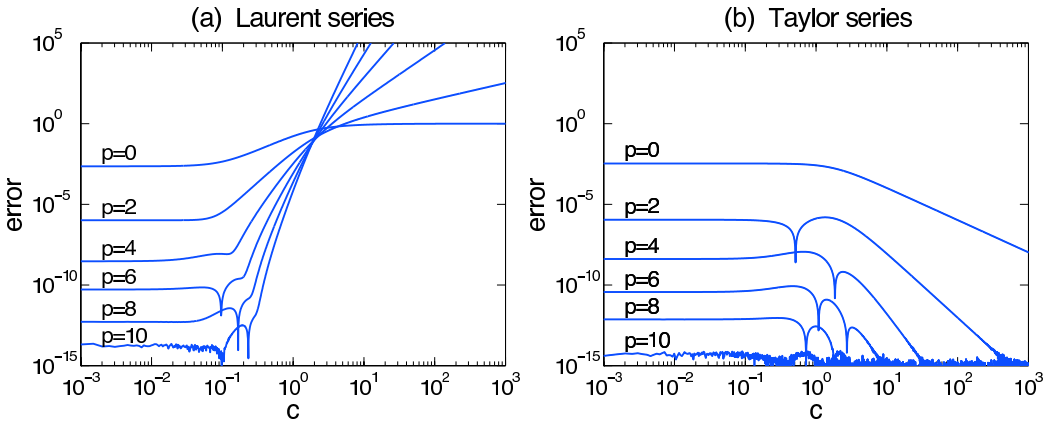


Figure 3.10: Error (3.36) in a particle-cluster approximation with one evaluation point at the origin in $\mathbb{R}^3$ and a cluster of $N = 10^3$ random nodes in the cube $[0.75, 1]^3$; $\nu = 1, d = 3$. The error is plotted as a function of the RBF parameter c and order p, for $10^{-3} \leq c \leq 10^3$ and $p = 0:2:10$. (a) Laurent series (3.24), (b) Taylor series (3.29).

KB level 2 cache). Memory usage was obtained using the performance counter tool in the Windows operating system.

We present results using the treecode for two test cases in 3D. In the first case we consider random nodes in a unit cube and in the second case the nodes were projected radially from the cube to the surface of a unit sphere Figure 3.11. One motivation for considering the sphere is the growing interest in RBF methods for geophysical fluid flow [32] and our Lagrangian method application on sphere (Chapter Five and

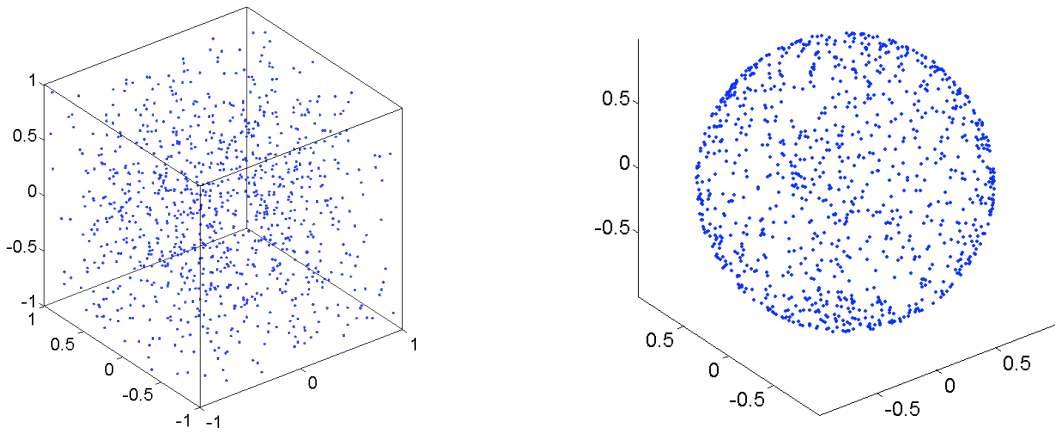


Figure 3.11: N random points in a cube (left) and on the surface of a sphere.

Chapter Six). In both test cases, the RBF coefficients were set to $d_i = 1$, as in [22], and the RBF parameter was set to $c = 10^{-1}$. The relative error is defined by

$$E_2 = \left(\sum_{i=1}^{N} |s(\boldsymbol{x}_i) - \tilde{s}(\boldsymbol{x}_i)|^2 \Big/ \sum_{i=1}^{N} |s(\boldsymbol{x}_i)|^2 \right)^{1/2}, \tag{3.37}$$

where $s(\boldsymbol{x}_i)$ is the exact value (3.2) obtained by direct summation and $\tilde{s}(\boldsymbol{x}_i)$ is the treecode approximation.

3.6.4 Random nodes in a cube

First we present results using $N = 216$K nodes for order $p = 0:2:10$ and MAC parameter $\theta = 0.2, 0.5, 0.8$, with maximum leaf size $N_0 = 200$. Table 3.1 presents the data. As expected, for a given value of θ, increasing order p yields smaller error and larger CPU time. Similarly, for a given value of p, decreasing MAC parameter θ also yields smaller error and larger CPU time. Figure 3.12 displays the data as a scatter plot. The data points for each θ are connected by a line and p increases moving from right to left on each line. The lower envelope of the data gives the most efficient choice of parameters (p, θ) to attain a given error. For example, errors in the interval $[10^{-6}, 10^{-1}]$ are computed with least CPU time using $\theta = 0.8$. We set $\theta = 0.8$ in the remainder of this work.

Table 3.1: Random nodes in a cube, the treecode error (3.37) and CPU time (sec) are given for system size $N = 216$K, RBF parameter $c = 10^{-1}$, order $p = 0 : 2 : 10$, MAC parameter $\theta = 0.2, 0.5, 0.8$, maximum leaf size $N_0 = 200$.

	treecode error (3.37)			CPU time (sec)		
p	$\theta = 0.8$	$\theta = 0.5$	$\theta = 0.2$	$\theta = 0.8$	$\theta = 0.5$	$\theta = 0.2$
0	3.379e-2	1.376e-2	2.213e-3	0.9	3.2	259.0
2	5.835e-5	1.292e-5	3.081e-7	7.8	30.2	470.1
4	2.014e-5	1.336e-6	2.711e-9	15.7	55.8	798.1
6	2.149e-6	8.761e-8	3.065e-11	33.5	119.3	1444.0
8	5.805e-7	5.461e-9	4.890e-13	63.0	308.9	2522.5
10	1.850e-7	6.107e-10	2.220e-14	115.7	684.4	4304.4

Next we vary the problem size. Figure 3.13 presents the error (3.37) as a function of order p for problems of size $N = 10^3, 20^3, \ldots, 100^3$. The maximum leaf size was $N_0 = 200$, except for the two largest systems ($N = 90^3, 100^3$) for which $N_0 = 400$. Figure 3.13 shows that the error is almost independent of system size N, and for a given value of N, the error decreases with increasing order p. Figure 3.14 presents the corresponding CPU time, showing that the treecode CPU time is consistent with $O(N \log N)$ scaling and that the treecode is faster than direct summation except for small system size N and large order p.

Figure 3.15 plots the amount of memory used by the treecode as a function of system size N for order $p = 0 : 2 : 10$. The memory used by direct summation is also shown. The treecode uses arrays of size $O(p^3)$ to store cluster moments and Taylor coefficients, and hence it requires more memory as the order p increases. For low order p, the treecode memory usage is a small multiple of the memory required by direct summation. For high order p, the treecode memory usage becomes irregular and this is why the maximum leaf size was changed from $N_0 = 200$ to $N_0 = 400$ for the two largest systems. It is likely that the treecode memory usage can be reduced

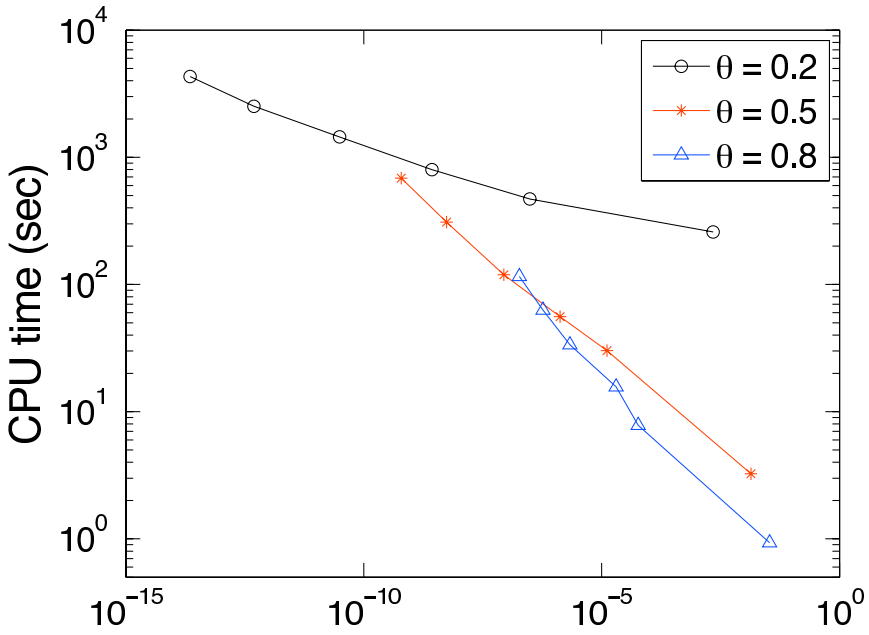


Figure 3.12: Random nodes in a cube, scatter plot of CPU time and error (3.37), data from Table 3.1, system size $N = 216$K, order $p = 0:2:10$ (increasing from right to left), MAC parameter $\theta = 0.2$ ($\circ$,black), $\theta = 0.5$ ($*$,red), $\theta = 0.8$ ($\triangle$,blue), RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$. The lower envelope of the data gives the most efficient choice of parameters (p, θ) to attain a given error.

by further tuning, but even so, for the largest system considered here ($N = 10^6$) and the highest order ($p = 10$), the treecode used approximately only three times as much memory as direct summation.

3.6.5 Random nodes on a sphere

Table 3.2 compares two test cases, random nodes in a cube and on a sphere, for problems of varying size, using a representative set of parameter values, $\theta = 0.8, p = 6$. We record results for the treecode and direct summation. Note that the direct sum CPU time and memory usage depend on the system size N, but not on the node distribution, so these values are the same for the two test cases. Next note that the treecode error is essentially the same for the two node distributions, while the treecode CPU time and memory usage are slightly higher for the sphere than for the cube, with approximately a 10% penalty in CPU time and a 33% penalty in

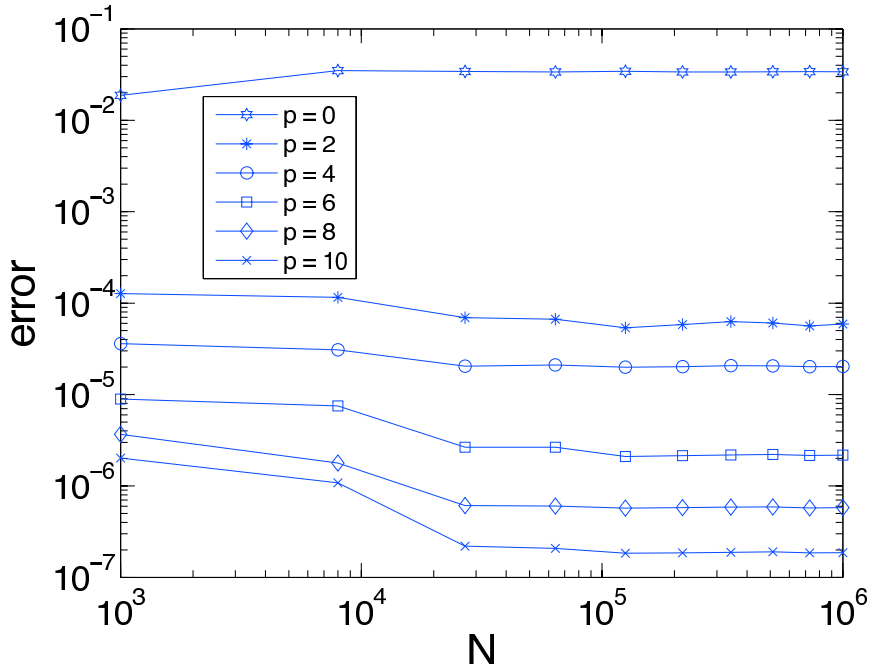


Figure 3.13: Random nodes in a cube, treecode error (3.37) plotted as a function of system size $N = 10^3, 20^3, \ldots, 100^3$, order $p = 0:2:10$, MAC parameter $\theta = 0.8$, RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$ except $N_0 = 400$ for $N = 90^3, 100^3$.

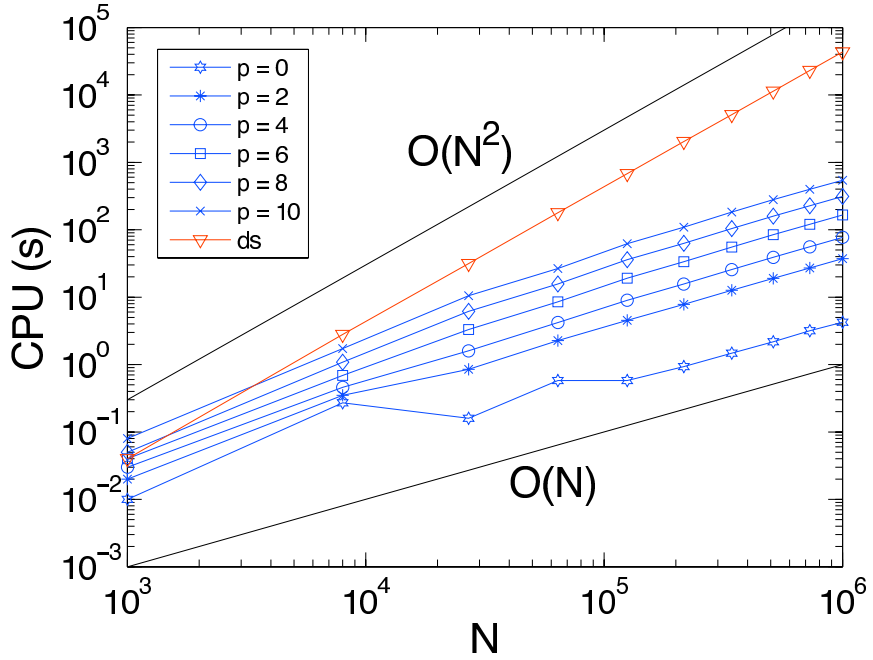


Figure 3.14: Random nodes in a cube, treecode CPU time in seconds, same parameters as in Figure 3.13 caption, ds denotes direct sum.

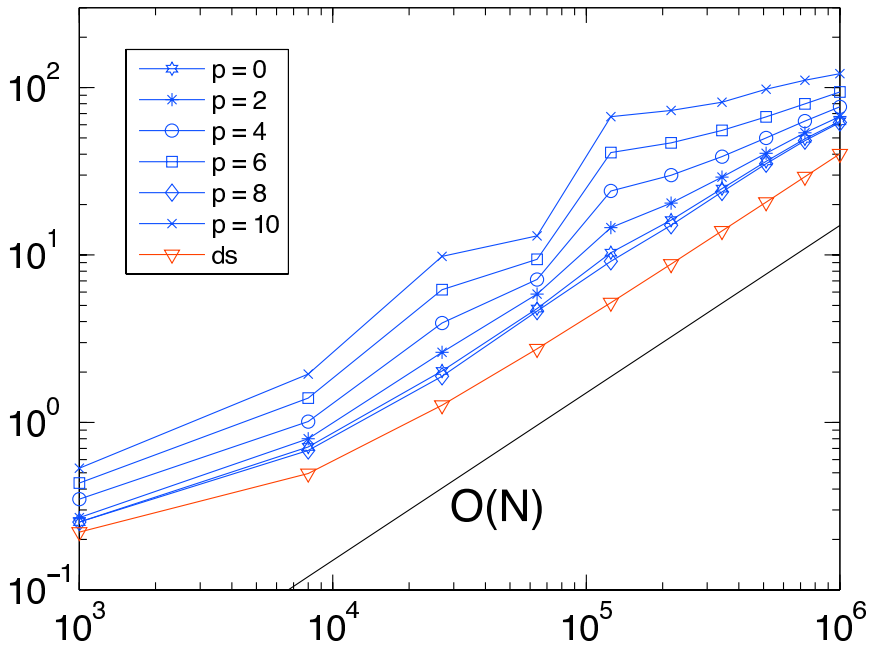


Figure 3.15: Random nodes in a cube, treecode memory usage in MB, same parameters as in Figure 3.13 caption, ds denotes direct sum.

memory usage for the largest system $N = 1000$K. It is likely that greater efficiency can be gained in the sphere test case, for example by shrinking the clusters [57] or by spherical panel clustering. The overall favorable performance of the treecode in the sphere test case arises from the adaptive nature of the tree structure, which has been noted before [7].

In this chapter, we present a fast Cartesian treecode for evaluating RBFs efficiently. The method applies a divide and conquer strategy and uses particle-cluster interactions in place of particle-particle interactions. Taylor approximation is applied for the far-field expansion. For multiquadric RBFs, $\phi(x) = \sqrt{x^2 + c^2}$, the Laurent series presented in the literature converges only for a limited range of c, but the Taylor series converges for all $c \geq 0$. The treecode algorithm reduces the computational cost from $O(N^2)$ to $O(N \log N)$ operations, where N is the size of the system.

Table 3.2: Random nodes in a cube and on a sphere, tc denotes treecode and ds denotes direct sum, CPU time in seconds and memory in MB, RBF parameter $c = 10^{-1}$, maximum leaf size $N_0 = 200$ except $N_0 = 400$ for $N = 1000$K, order $p = 6$, MAC parameter $\theta = 0.8$.

$N \rightarrow$	8K	64K	216K	512K	1000K
cube					
error (tc)	7.5e-6	2.7e-6	2.1e-6	2.2e-6	2.2e-6
CPU (tc)	0.7	8.5	33.5	84.4	165.9
CPU (ds)	1.5	95.7	978.8	5479.0	21888.2
memory (tc)	1.0	7.1	29.9	50.0	76.7
memory (ds)	0.5	2.8	8.8	20.7	40.2
sphere					
error (tc)	4.0e-6	2.6e-6	2.6e-6	2.4e-6	2.1e-6
CPU (tc)	0.7	8.7	34.1	88.5	180.7
CPU (ds)	1.5	95.7	978.8	5479.0	21888.2
memory (tc)	1.0	9.1	29.6	72.0	101.2
memory (ds)	0.5	2.8	8.8	20.7	40.2

CHAPTER IV

Barotropic Vorticity Equation

4.1 Introduction

We are interested in fluid flow on the surface of the sphere $S^2 \subset \mathbb{R}^3$. General atmospheric motion is described by partial differential equations with different assumptions. Figure 1.1 shows hierarchy of the atmospheric models. The incompressible Barotropic Vorticity Equation (BVE) is a simple mathematical model for the description of large-scale horizontal motions of the atmosphere. For theoretical investigations of the evolution of vortices, atmospheric researchers are still using the barotropic assumption, that is, pressure depends only on the density. For example, the BVE is useful for modeling the movement of tropical cyclones [26] and interaction of two vortices in close proximity to one another [71]. Therefore, the (BVE) is a reasonable first step for investigating complex atmospheric motions. In this chapter, we'll give an overview of the solutions of the BVE. The detailed discussion of numerical methods and computational experiments will be done in the following two chapters.

For convenience, we will use both Cartesian coordinates $\boldsymbol{x} = (x, y, z)$ and spherical coordinates (θ, λ), where $\theta \in [0, \pi]$ is the colatitude, which is 0 at the north pole and

π at the south pole, and $\lambda \in [0, 2\pi]$ is the longitude. They are related by

$$x = \sin\theta \cos\lambda, \tag{4.1}$$

$$y = \sin\theta \sin\lambda, \tag{4.2}$$

$$z = \cos\theta. \tag{4.3}$$

We suppose the sphere is rotating around the z axis with angular velocity Ω, then the Coriolis force,

$$f = 2\Omega \cos\theta, \tag{4.4}$$

can be rewritten by

$$f = 2\Omega z. \tag{4.5}$$

For incompressible flow, we have

$$\nabla \cdot \boldsymbol{u} = 0, \tag{4.6}$$

where $\boldsymbol{u}(\boldsymbol{x}, t)$ is the fluid velocity on the sphere and we have

$$\nabla \times \boldsymbol{u} = \zeta \boldsymbol{e_r}, \tag{4.7}$$

where ζ is the relative vorticity and $\boldsymbol{e_r}$ is the unit vector in the radial direction. The BVE system on a rotating sphere is given by

$$\boldsymbol{u} = \nabla\psi \times \boldsymbol{x}, \tag{4.8}$$

$$\Delta_s \psi = -\zeta, \tag{4.9}$$

$$\frac{\partial \eta}{\partial t} + \boldsymbol{u} \cdot \nabla \eta = 0, \tag{4.10}$$

where $\psi(\boldsymbol{x}, t)$ is the stream function, Δ_s is the surface spherical Laplacian

$$\Delta_s \equiv \frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial}{\partial\theta} \right) + \frac{1}{\sin^2\theta} \frac{\partial^2}{\partial\lambda^2} \tag{4.11}$$

and $\eta = \zeta + f$ is the absolute vorticity. Equation (4.8) gives the relation of velocity $\boldsymbol{u}$ and the stream function ψ. Equation (4.9) is the stream function and vorticity Poisson equation. Equation (4.10) is the conservation of the absolute vorticity η. The vorticity equation (4.10) can also be written as [46],

$$\frac{\partial \nabla^2 \psi}{\partial t} = \frac{\partial \psi}{\partial \mu}\frac{\partial \nabla^2 \psi}{\partial \lambda} - \frac{\partial \psi}{\partial \lambda}\frac{\partial \nabla^2 \psi}{\partial \mu} - 2\Omega \frac{\partial \psi}{\partial \lambda}, \tag{4.12}$$

where $\mu = \cos\theta$ and

$$\nabla^2 \psi = \frac{\partial}{\partial \mu}\left[\left(1-\mu^2\right)\frac{\partial \psi}{\partial \mu}\right] + \frac{1}{1-\mu^2}\frac{\partial^2 \psi}{\partial \lambda^2}, \tag{4.13}$$

which we will use later.

The BVE has been studied for more than half a century. Charney, Fjörtoft and von Neumann (1950) presented a numerical solution of BVE over a limited area of the earth's surface. It was the first generation of numerical weather prediction [20]. Adem (1956) introduced a method to obtain a series solution for BVE [3]. Sadourny, Arakawa and Mintz (1968) solved the BVE by a finite-difference scheme using icosahedral-hexagonal grid [69]. Levy, Nair and Tufo (2009) [56] presented a combination of a discontinuous Galerkin and spectral method for solving BVE numerically.

The solution process for solving the BVE involves solving a conservative transport equation (4.10) for the vorticity and a Poisson equation (4.9) for the stream function. Let's focus on the Poisson equation now.

4.2 Solution of Poisson Equation on the Sphere

4.2.1 Spherical harmonics

The spherical harmonics are defined as,

$$Y_m^n(\mu, \lambda) \equiv P_m^n(\mu)\exp(im\lambda), \tag{4.14}$$

where $\mu = \cos\theta$ and n, m are integer indices given by $m = 0, \pm1, \pm2, \pm3, ..., n = 1, 2, 3, ...$, with $|m| \leq n$. The formula are orthogonal basis functions for (4.11)[46], which are defined as Here, $P_m^n(\mu)$ designates an associated Legendre function of the first kind of degree n and m designates the zonal wave number.

An important property of the spherical harmonics is that they are eigenfunctions of (4.11) and satisfy the relation

$$\nabla^2 Y_m^n = -n(n+1)Y_m^n, \tag{4.15}$$

where we assume the sphere has unit radius. The Laplacian of a spherical harmonic is proportional to the function itself. That means, if we let the stream function be a particular spherical harmonic,

$$\psi(\mu, \lambda) = Y_m^n, \tag{4.16}$$

then the corresponding vorticity is

$$\zeta = -\nabla^2\psi = -\nabla^2 Y_m^n = n(n+1)Y_m^n, \tag{4.17}$$

which implies that the vorticity is simply proportional to the stream function for this case. We can expand a general stream function in a series of spherical harmonics

$$\psi(\lambda, \mu, t) = \sum_{n=1}^{\infty} \sum_{m=-n}^{n} \psi_{m,n}(t) Y_m^n(\mu, \lambda), \tag{4.18}$$

where $\psi_{m,n}$ is the complex amplitude for the Y_m^n spherical harmonic and the summation is over m and n. The spherical harmonic coefficients $\psi_{m,n}$ are related to the stream function $\psi(\lambda, \mu)$ through the inverse transform

$$\psi_{m,n}(t) = \frac{1}{4\pi} \int_S \bar{Y_m^n} \psi(\lambda, \mu, t) dS, \tag{4.19}$$

where $dS = d\mu d\lambda$ and $\bar{Y_m^n}$ designates the complex conjugate of Y_m^n.

4.2.2 Green's function

The other way to solve the Poisson equation on the sphere is using Green's function. Kimura and Okamoto (1987) presented the Green's function for the surface spherical Laplacian (4.11)

$$G(\theta, \lambda, \theta', \lambda') = -\frac{1}{4\pi} \log(1 - \cos\gamma), \tag{4.20}$$

where

$$\cos\gamma = \cos\theta \cos\theta' + \sin\theta \sin\theta' \cos(\lambda - \lambda'), \tag{4.21}$$

and γ is the central angle between two points θ, λ and θ', λ', as in Figure 4.1 [51].

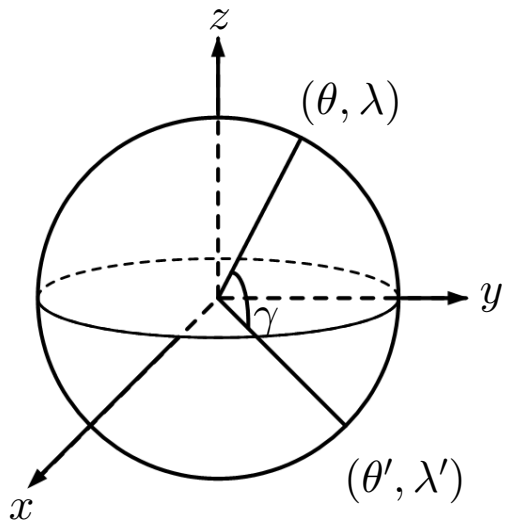


Figure 4.1: Diagram for showing the central angle between two points (θ, λ) and (θ', λ') on sphere.

The Green's function (4.20) satisfies

$$\frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial G}{\partial\theta} \right) + \frac{1}{\sin^2\theta} \frac{\partial^2 G}{\partial\lambda^2} = -\left(\delta(\theta, \lambda, \theta', \lambda') - \frac{1}{4\pi} \right). \tag{4.22}$$

where $\delta(\theta, \lambda, \theta', \lambda')$ is the delta function. The right hand side of equation (4.22) satisfies

$$\int_{S^2} \left(\delta - \frac{1}{4\pi} \right) dS = \int_{S^2} \delta dS - \int_{S^2} \frac{1}{4\pi} dS = 0, \tag{4.23}$$

which is a necessary condition for (4.9) to have a bounded solution on the sphere [25], [18]. The formula (4.22) can be derived easily as we now show. Without loss of generality, let's suppose the source point θ', λ' is located at the north pole, that is, $\theta' = 0$ and λ' can be any real number, then by (4.21) we have $\cos\gamma = \cos\theta$. Hence, the Green's function (4.20) depends on the colatitude θ only,

$$G(\theta) = -\frac{1}{4\pi}\log(1-\cos\theta), \tag{4.24}$$

and the spherical Laplacian becomes,

$$\Delta_s = \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right). \tag{4.25}$$

For $\theta > 0$, that is, for all source points not at the north pole, we have

$$\Delta_s G(\theta) = -\frac{1}{4\pi}\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial G}{\partial\theta}\right) \tag{4.26}$$

$$= -\frac{1}{4\pi}\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\frac{\sin^2\theta}{1-\cos\theta}\right) \tag{4.27}$$

$$= -\frac{1}{4\pi}\frac{-(1-\cos\theta)^2}{(1-\cos\theta)^2} \tag{4.28}$$

$$= \frac{1}{4\pi}. \tag{4.29}$$

Next, we show that (4.22) holds in the sense of distribution.

Theorem IV.1. *The Green's function defined in (4.24) is a fundamental solution of the Laplace equation on the sphere (4.11), that is, $\Delta_s G = -\delta + \frac{1}{4\pi}$ in the sense of distributions*

$$< \Delta_s G, f > = < -\delta + \frac{1}{4\pi}, f >, \tag{4.30}$$

for any test function $f(\theta, \lambda)$.

Proof. The right hand side of (4.30) is

$$< -\delta + \frac{1}{4\pi}, f > = \int_0^{2\pi} \int_0^{\pi} (-\delta + \frac{1}{4\pi}) f(\theta, \lambda) \sin\theta d\theta d\lambda \tag{4.31}$$

$$= -f(0,0) + \frac{1}{4\pi} \int_0^{2\pi} \int_0^{\pi} f(\theta, \lambda) \sin\theta d\theta d\lambda. \tag{4.32}$$

Since the Laplacian Δ_s is a self-adjoint operator, we have

$$< \Delta_s G, f > \equiv < G, \Delta_s f >, \tag{4.33}$$

and

$$\begin{aligned} < G, \nabla^2 f > = & \int_0^{2\pi} \int_0^{\pi} G \frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial f}{\partial\theta} \right) \sin\theta d\theta d\lambda \\ & + \int_0^{2\pi} \int_0^{\pi} G \frac{1}{\sin^2\theta} \frac{\partial^2 f}{\partial\lambda^2} \sin\theta d\theta d\lambda. \end{aligned} \tag{4.34}$$

The second term in (4.34) is zero due to f is a periodic function on sphere, then we have

$$< G, \nabla^2 f > = \int_0^{2\pi} \int_0^{\pi} G \frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial f}{\partial\theta} \right) \sin\theta d\theta d\lambda. \tag{4.35}$$

Plug in the Green's function (4.24) to equation (4.35),

$$< G, \nabla^2 f > = -\frac{1}{4\pi} \left\{ \int_0^{2\pi} \int_0^{\pi} \log(1 - \cos\theta) \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial f}{\partial\theta} \right) d\theta d\lambda \right\}. \tag{4.36}$$

Then employ the integration by parts twice, we have

$$
\begin{aligned}
< G, \nabla^2 f > &= -\frac{1}{4\pi}\left\{\int_0^{2\pi} \log(1-\cos\theta)\sin\theta \frac{\partial f}{\partial\theta}|_{\theta=0}^{\pi} d\lambda - \int_0^{2\pi}\int_0^{\pi} \sin\theta \frac{\partial f}{\partial\theta}\frac{\sin\theta}{1-\cos\theta} d\theta d\lambda\right\} \\
&= \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} \frac{\partial f}{\partial\theta}\frac{\sin^2\theta}{1-\cos\theta} d\theta d\lambda \\
&= \frac{1}{4\pi}\left\{\int_0^{2\pi} f(\theta,\lambda)\frac{\sin^2\theta}{1-\cos\theta}|_{\theta=0}^{\pi} d\lambda - \int_0^{\pi}\int_0^{2\pi} f(\theta)\left(\frac{\sin^2\theta}{1-\cos\theta}\right)' d\theta d\lambda\right\} \\
&= \frac{1}{4\pi}\left\{\int_0^{2\pi} f(\theta,\lambda)\frac{1-\cos^2\theta}{1-\cos\theta}|_{\theta=0}^{\pi} d\lambda + \int_0^{2\pi}\int_0^{\pi} f(\theta,\lambda)\sin\theta d\theta d\lambda\right\} \\
&= \frac{1}{4\pi}\left\{\int_0^{2\pi} f(\theta,\lambda)(1+\cos\theta)|_{\theta=0}^{\pi} d\lambda + \int_0^{2\pi}\int_0^{\pi} f(\theta,\lambda)\sin\theta d\theta d\lambda\right\} \\
&= \frac{1}{4\pi}\left\{\int_0^{2\pi} -2f(0,\lambda) d\lambda + \int_0^{2\pi}\int_0^{\pi} f(\theta,\lambda)\sin\theta d\theta d\lambda\right\} \\
&= \frac{1}{4\pi}\left\{-4\pi f(0,0) + \int_0^{2\pi}\int_0^{\pi} f(\theta,\lambda)\sin\theta d\theta d\lambda\right\} \\
&= -f(0,0) + \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} f(\theta,\lambda)\sin\theta d\theta d\lambda.
\end{aligned}
$$

□

For later reference, the Green's function (4.20) can be written as

$$G(\boldsymbol{x}, \boldsymbol{x}') = -\frac{1}{4\pi}\log(1 - \boldsymbol{x}\cdot\boldsymbol{x}'). \tag{4.37}$$

For points on the surface of a unit sphere, it is obvious that

$$\cos\gamma = \boldsymbol{x}\cdot\boldsymbol{x}', \tag{4.38}$$

which proves that (4.20) and (4.37) are equivalent. Using the Green's function (4.20), we can obtain the solution of Poisson equation (4.9),

$$\psi(\theta,\phi) = \frac{1}{4\pi}\int\int \zeta(\theta',\lambda')\log(1-\cos\gamma)\sin\theta' d\theta' d\lambda'. \tag{4.39}$$

The relative vorticity ζ defined as a sum of δ functions,

$$\zeta(\theta,\lambda) = \sum_{j=1}^{N} \Gamma_j \delta_j(\theta,\lambda,\theta_j,\lambda_j), \tag{4.40}$$

where Γ_j is the strength of the jth point vortex at (θ_j, λ_j). This is the basic idea for the point vortex method [51]. Next, we'll introduce regularized point vortex method.

4.2.3 Regularized Green's Function

The regularized point vortex model has been applied in vortex sheet simulation and vortex dynamics on the sphere [52], [70]. It shows better properties than point vortex model. The Green's function given by (4.20) may be regularized as follows

$$G_\delta(\theta, \lambda, \theta', \lambda') = -\frac{1}{4\pi} \log(1 - \cos\gamma + \delta^2), \tag{4.41}$$

where here $\delta > 0$ is a small number. Now the question is what is the corresponding vorticity field? Again, without loss of generality, we suppose the source point θ', λ' is located at the north pole, then the Green's function only depends on the colatitude θ, that is,

$$G_\delta(\theta) = -\frac{1}{4\pi} \log(1 - \cos\theta + \delta^2). \tag{4.42}$$

Taking the surface spherical Laplacian (4.11),

$$\Delta_s G_\delta(\theta) = -\frac{1}{4\pi} \left(\frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\sin\theta \frac{\partial G_\delta}{\partial\theta} \right) \right) \tag{4.43}$$

$$= -\frac{1}{4\pi} \left(\frac{1}{\sin\theta} \frac{\partial}{\partial\theta} \left(\frac{\sin^2\theta}{1 - \cos\theta + \delta^2} \right) \right) \tag{4.44}$$

$$= -\frac{1}{4\pi} \left(\frac{-(1 - \cos\theta)^2 + 2\delta^2 \cos\theta}{(1 - \cos\theta + \delta^2)^2} \right) \tag{4.45}$$

$$\approx -\zeta_\delta(\theta). \tag{4.46}$$

Figure 4.2 shows the regularized vorticity associated with the regularized Green's function $G_\delta(\theta)$ for different δ. As we see from this Figure, as δ decreases, the vorticity function tends to the spike delta function. This implies the regularized Green's function (4.42) is a good approximation of the Green's function (4.20). For a source

point at any position (θ_s, λ_s), we have by rotation

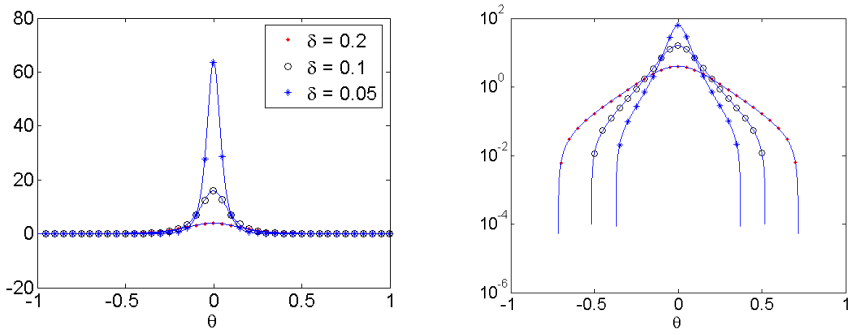
$$\Delta_s G_\delta(\theta,\lambda) = -\frac{1}{4\pi}\left(\frac{-(1-\cos(\theta,\lambda,\theta_s,\lambda_s))^2 + 2\delta^2\cos(\theta,\lambda,\theta_s,\lambda_s)}{(1-\cos(\theta,\lambda,\theta_s,\lambda_s)+\delta^2)^2}\right) \tag{4.47}$$

$$\approx -\zeta_\delta(\theta,\lambda), \tag{4.48}$$

where $\cos(\theta,\lambda,\theta_s,\lambda_s)$ is the cosine of the central angle between (θ,λ) and (θ_s,λ_s). The regularized stream function is

$$\psi(\theta,\phi) = -\frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi}\zeta(\theta',\lambda')\log(1-\cos\gamma+\delta^2)\sin\theta' d\theta' d\lambda'. \tag{4.49}$$

We will employ this regularized point vortex model in detail in Chapter Six.



(a) (b)

Figure 4.2: (a) and (b) are the plot of the vorticity ζ_δ (4.46) from smooth Green's function G_δ (4.42) for $\delta = 0.2, 0.1, 0.05$, where (b) is in log scale. Both figures share the same legend. Note that ζ_δ is an approximate delta function.

4.2.4 Gaussian forcing

Besides the regularized Green's function mentioned above, there are other ways to approximate the smooth vorticity field, such as RBF approximations. Here we will focus on the Gaussian RBF. Boyd and Zhou (2009) [18] presented several ways to solve the Poisson equation for a Gaussian centered at the north pole with the stream function dependent on colatitude θ only:

$$\Delta_S G_\epsilon(\theta) = \zeta_\epsilon(\theta), \tag{4.50}$$

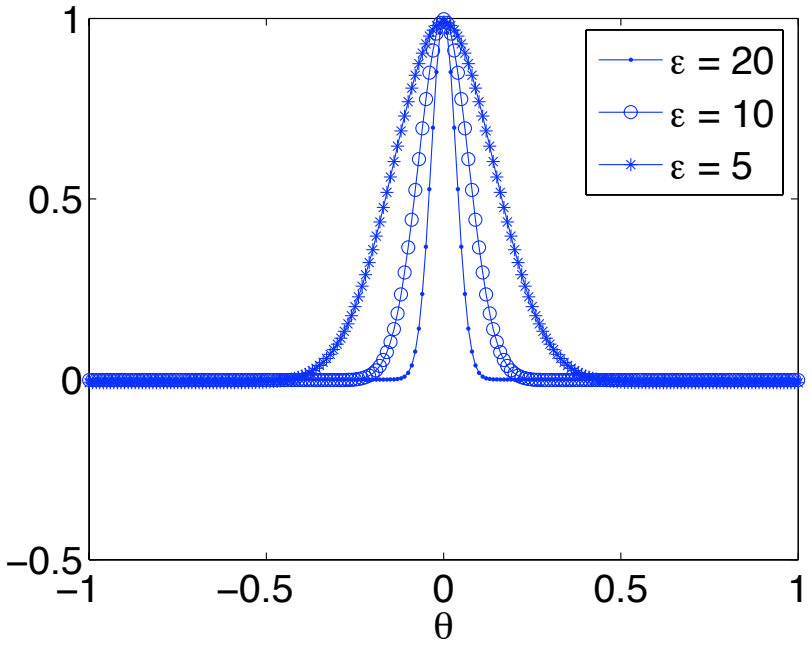


Figure 4.3: The vorticity function ζ_ϵ (4.51) with Gaussian forcing at north pole for different ϵ.

where ζ_ϵ is

$$\zeta_\epsilon(\theta) = \exp(-2\epsilon^2(1-\cos\theta)) - C^{\text{Gauss}}(\epsilon), \tag{4.51}$$

θ is colatitude, ϵ is a positive constant, and

$$C^{\text{Gauss}}(\epsilon) = \frac{1}{4\epsilon}\left(1-\exp(-4\epsilon^2)\right) \tag{4.52}$$

is the "Gauss constraint constant". The reason for the Gauss constraint constant is that the vorticity averaged over the sphere must be zero. This is a necessary condition for the ψ of the Poisson equation to be bounded [25, 24].

As we mentioned in section (4.2.1), the spherical harmonics are eigenfunctions of the Laplace operator on the surface of a sphere. Hence the solution of (4.51) in Legendre polynomials is

$$\psi = \sum_{n=1}^{\infty}\left(-\frac{1}{n(n+1)}\right)\frac{2n+1}{2}\frac{\sqrt{\pi}}{\epsilon}\exp(-2\epsilon^2)I_{n+1/2}(2\epsilon^2)P_n(\cos\theta), \tag{4.53}$$

where the $P_n(x)$ are the usual unnormalized Legendre polynomials and $I_{n+1/2}$ are the usual modified spherical Bessel functions. The disadvantage of this solution formula is that the rate of convergence of the series is slow for large ϵ, as shown in [18].

In [18] the exact solution formula for (4.51) is derived,

$$\begin{aligned} G_\epsilon(\theta) &= \frac{1}{4\epsilon^2}\left\{1-\exp(-4\epsilon^2)\right\}\log(1-\mu) - \frac{1}{4\epsilon^2}\exp(-4\epsilon^2)\log\left(\frac{1+\mu}{1-\mu}\right) \\ &+ \frac{1}{4\epsilon^2}\mathrm{E_1}(2\epsilon^2[1-\mu]) + \frac{1}{4\epsilon^2}\exp(-4\epsilon^2)\mathrm{E_i}(2\epsilon^2[1+\mu]), \end{aligned} \tag{4.54}$$

where $\mu = \cos\theta$,

$$\mathrm{E_1(z)} \equiv \int_1^\infty \frac{\exp(-\mathrm{zt})}{\mathrm{t}}\mathrm{dt}, \tag{4.55}$$

and

$$\mathrm{E_i(z)} \equiv \gamma + \log(\mathrm{z}) + \int_0^\mathrm{z} \frac{\exp(\mathrm{t})-1}{\mathrm{t}}\mathrm{dt}. \tag{4.56}$$

Note that the above solution (4.54) is a function of $\cos\theta$ and ϵ only, and it can be extended to the solution of Poisson equation with Gaussian at any position λ_s, θ_s by rotation as we did in (4.47).

4.3 Rossby-Haurwitz Wave

4.3.1 Stream function

Note that an exact analytic solution of equation (4.9) can be obtained in the special case where the stream function is equal to a single spherical harmonic. Thus we let

$$\psi(\lambda,\mu,t) = \psi_{m,n}(t)\exp(im\lambda)P_m^n(\mu). \tag{4.57}$$

Substituting (4.57) into (4.12) and applying (4.15), we find that the non-linear advection terms are identically zero so that the amplitude coefficient satisfies the ordinary

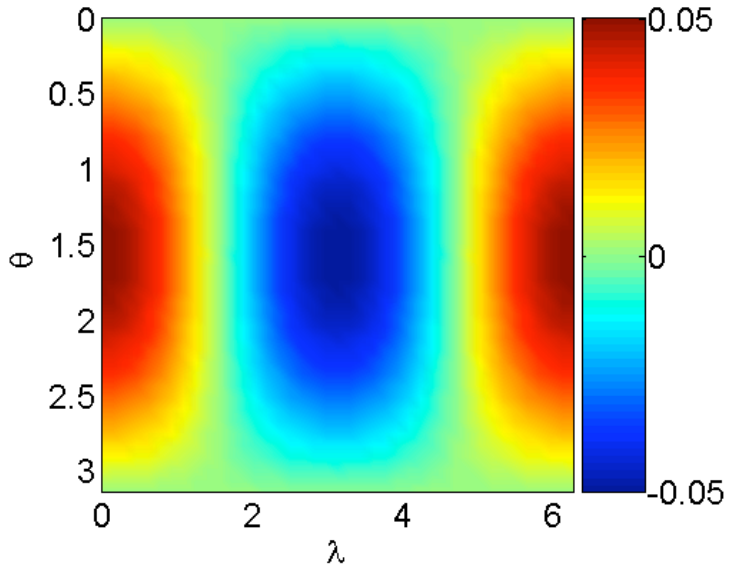


Figure 4.4: Initial stream function with amplitude $A = 0.05$ and $n = 1, m = 1$.

linear differential equation

$$-n(n+1)\frac{d\psi_{m,n}}{dt} = -2\Omega im\psi_{m,n}, \tag{4.58}$$

which has the solution $\psi_{m,n}(t) = \psi_{m,n}(0)\exp(i\nu_{m,n}t)$, where

$$\nu_{m,n} = 2\Omega m/\left[n(n+1)\right], \tag{4.59}$$

is the dispersion relation for the Rossby-Haurwitz wave [46].

4.3.2 Example

Here we take the simplest case as an example: $n = 1, m = 1$, then $\nu_{(m,n)} = \Omega$. The initial Rossby-Hauwritz wave is

$$\psi(\theta, \lambda) = A\sin\theta\cos\lambda, \tag{4.60}$$

as in Figure 4.4 in spherical coordinates, where A is the amplitude of the wave. Its

Cartesian coordinate counterpart is

$$\psi = Ax, \tag{4.61}$$

$$\zeta = 2Ax. \tag{4.62}$$

The stream function at time t is

$$\psi(\theta, \lambda, t) = A \sin\theta \cos(\lambda + \Omega t) \tag{4.63}$$

$$= A \sin\theta \left(\cos\lambda \cos\Omega t - \sin\lambda \sin\Omega t\right) \tag{4.64}$$

$$= A \left(x \cos\Omega t - y \sin\Omega t\right). \tag{4.65}$$

Figure 4.5 shows the stream lines at $t = \pi, 2\pi, 3\pi, 4\pi$ with amplitude $A = 0.05$ and angular velocity $\Omega = 1/2$. We can see the wave propagates towards west with a constant velocity Ω.

From (4.8) we have

$$\boldsymbol{u} = (u, v, w) \tag{4.66}$$

$$= \nabla\psi \times \boldsymbol{x} \tag{4.67}$$

$$= (\frac{\partial\psi}{\partial y} z - \frac{\partial\psi}{\partial z} y)\boldsymbol{i} + (\frac{\partial\psi}{\partial z} x - \frac{\partial\psi}{\partial x} z)\boldsymbol{j} + (\frac{\partial\psi}{\partial x} y - \frac{\partial\psi}{\partial y} x)\boldsymbol{k} \tag{4.68}$$

that is,

$$u = \frac{\partial\psi}{\partial y} z - \frac{\partial\psi}{\partial z} y, \tag{4.69}$$

$$v = \frac{\partial\psi}{\partial z} x - \frac{\partial\psi}{\partial x} z, \tag{4.70}$$

$$w = \frac{\partial\psi}{\partial x} y - \frac{\partial\psi}{\partial y} x. \tag{4.71}$$

The trajectories of the fluid particles are obtained by solving the following set of

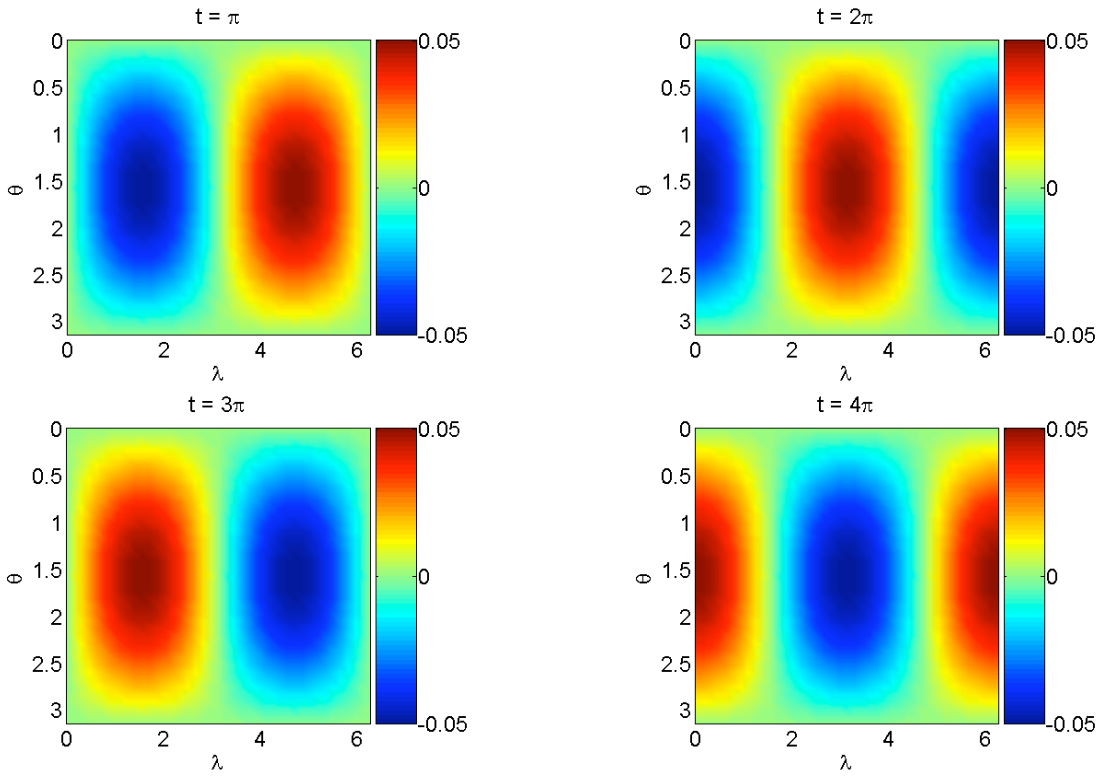


Figure 4.5: Stream function from equation (4.63) at $t = \pi, 2\pi, 3\pi, 4\pi$ with amplitude $A = 0.05$ and angular velocity $\Omega = 1/2$.

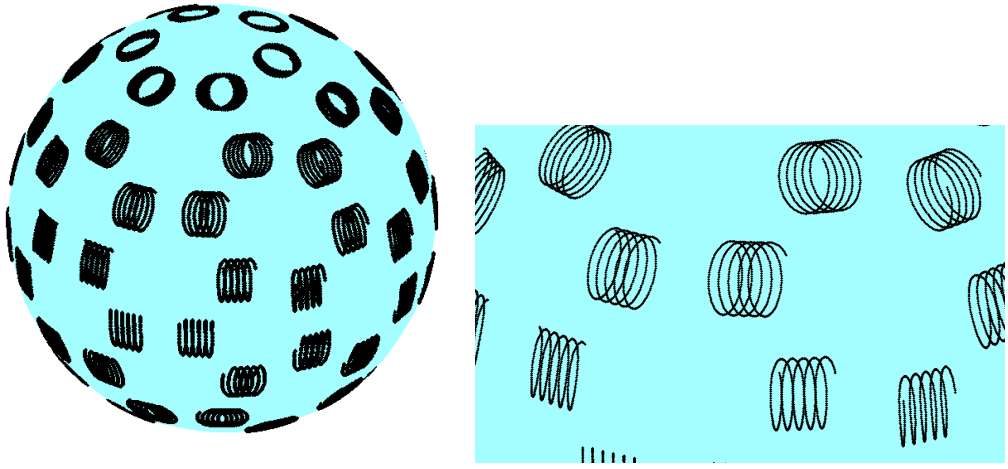
ODEs,

$$\frac{dx}{dt} = u = -Az\sin\Omega t, \tag{4.72}$$

$$\frac{dy}{dt} = v = -Az\cos\Omega t, \tag{4.73}$$

$$\frac{dz}{dt} = z = A(y\cos\Omega t + x\sin\Omega t), \tag{4.74}$$

Figure 4.6 shows the particle trajectories obtained by solving equations (4.72) to (4.74) using fourth order Runge-Kutta for five revolutions, where $A = 0.05$ and $\Omega = 1/2$. We will compare these trajectories with numerical results obtained by the RBF and point vortex methods described in the next two chapters.



(a) (b)

Figure 4.6: Rossby-Haurwitz wave (n = m = 1). (a) : particle trajectories obtained by solving equations (4.72) to (4.74) using fourth order Runge-Kutta for five revolutions, where $A = 0.05$ and $\Omega = 1/2$; (b): local zoom of the left figure.

In this chapter, we give an overview of the solutions of the barotropic vorticity equation. It includes the spherical harmonic expansions, Green's function on sphere and Poisson equation with Gaussian forcing. The Rossby-Haurwitz wave example will be tested by the numerical methods that we will employ in Chapters Five and Six.

CHAPTER V

Solving the Barotropic Vorticity Equation by Gaussian RBF

5.1 Soving PDEs by RBF

The Radial Basis Function method has great potential not only for its ability for scattered data interpolation in multi-dimensions but also for meshfree solving PDEs. Kansa (1990) [49, 50] was the first person to solve PDEs by Multiquadric RBF. His method consisted of approximating spatial partial derivatives by differentiating smooth RBF interpolants to solve parabolic, elliptic, and viscously damped hyperbolic PDEs to spectral accuracy in a completely mesh-free manner. Larsson and Fornberg (2003) [54] proposed a method for solving elliptic PDEs by means of collocation with both piecewise and infinitely smooth RBFs. They concluded that for PDE applications with smooth solutions, the infinitely smooth RBFs are preferable for higher accuracy. They also compared RBF-based methods against a second-order finite difference method and a pseudospectral method and showed the former gave a much superior accuracy. Flyer and Wright (2007) [32] solved hyperbolic partial differential equations on a sphere. They showed that RBFs allow for a much lower spatial resolution and are able to take unusually large time steps to achieve the same accuracy as compared to other commonly used spectral methods on a sphere such as spherical harmonics, double Fourier series, and spectral element methods. Flyer and

Wright (2009) [33] presented a radial basis function method for the shallow water equations on a sphere. However all the existing approaches for solving PDE by RBFs use an Eulerian formulation, that is, all the points that are the centers of the RBFs are fixed in time. Here we present a way to solve the BVE by RBF in a Lagrangian manner, which means, all the centers of RBFs are Lagrangian points which move with the fluid flow.

5.2 Solving BVE by Gaussian RBF

The first step to solve the BVE by Gaussian RBF is to approximate the smooth relative vorticity by Gaussians [8]. Let us expand the relative vorticity ζ by Gaussian RBF (GARBF)

$$\zeta(\theta,\lambda) = \sum_{j=1}^{N} d_j \left(\exp(-\epsilon^2 r_j^2) - C^{\text{Gauss}}(\epsilon)\right), \tag{5.1}$$

where ϵ is an RBF parameter and r_j is the Euclidean distance between the evaluation points (θ,λ) and the centers (θ_j,λ_j),

$$r_j = r(\theta,\lambda,\theta_j,\lambda_j) = \sqrt{2(1-\cos\theta\cos\theta_j\cos(\lambda-\lambda_j)-\sin\theta\sin\theta_j)}, \tag{5.2}$$

$C^{\text{Gauss}}(\epsilon)$ is the "Gauss constraint constant" defined in (4.52). The RBF coefficients d_j are obtained by interpolation as we mentioned in Chapter Two,

$$s(\theta_m,\lambda_m) = \sum_{j=1}^{N} d_j \left(\exp(-\epsilon^2 r_{mj}^2) - C^{\text{Gauss}}(\epsilon)\right) = \zeta_m, \tag{5.3}$$

where $\zeta_m = \zeta(\theta_m,\lambda_m)$ and $r_{mj} = r(\theta_m,\lambda_m,\theta_j,\lambda_j)$. Note that the basis functions are Gaussians minus a constant since the total integral of the relative vorticity on sphere should be zero, which is a necessary condition for Poisson equation (4.9) has a bounded solution as we mentioned in Chapter Four. It is a slight difference compared to the Gaussian RBF we defined in Chapter Two. Substituting (5.1) into the Poisson

equation (4.9), we have

$$\Delta_s \psi(\theta,\lambda) = -\zeta = -\sum_{j=1}^{N} d_j \left(\exp(-\epsilon^2 r_j^2) - C^{\text{Gauss}}(\epsilon)\right). \tag{5.4}$$

Define ψ_j by

$$\Delta_s \psi_j(\theta,\lambda) = \exp(-\epsilon^2 r_j^2) - C^{\text{Gauss}}(\epsilon). \tag{5.5}$$

Since surface spherical Laplacian is a linear operator, then the stream function is

$$\psi(\theta,\lambda) = -\sum_{j} d_j \psi_j(\theta,\lambda). \tag{5.6}$$

Boyd and Zhou (2009) presented the solution of the Poisson equation with Gaussian forcing ψ_j in equation (4.54) [18] as introduced in Chapter Four, which is the solution of equation (5.5). Then the stream function $\psi(\theta,\lambda)$ can be obtained by equation (5.6). In order to get the velocity field, we need to know the gradient of the stream function ψ_j, [18],

$$\frac{\partial \psi_j}{\partial \lambda} = \frac{\partial \psi_j}{\partial \gamma}\frac{\partial \gamma}{\partial \lambda} = \frac{\partial \psi_j}{\partial \gamma}\left(\sin\theta_j \sin\theta \sin(\lambda_j - \lambda)\right), \tag{5.7}$$

$$\frac{\partial \psi_j}{\partial \theta} = \frac{\partial \psi_j}{\partial \gamma}\frac{\partial \gamma}{\partial \theta} = \frac{\partial \psi_j}{\partial \gamma}\left(-\cos\theta_j \sin\theta + \sin\theta_j \cos\theta \cos(\lambda_j - \lambda)\right), \tag{5.8}$$

where

$$\frac{\partial \psi_j}{\partial \gamma} = \frac{1}{4\epsilon^2}\left(\frac{\exp(-2\epsilon^2(1-\gamma)) - 1}{1-\gamma} + \frac{\exp(-2\epsilon^2(1-\gamma)) - \exp(1-4\epsilon^2)}{1+\gamma}\right), \tag{5.9}$$

and γ is the central angle between (θ,λ) and (θ_j,λ_j) as defined in Figure 4.1. Now, we have ODEs for the trajectory of the centers of the Gaussian RBFs,

$$\frac{d\lambda_m}{dt} = \frac{1}{\sqrt{1-\mu_m^2}} \sum_{j=1}^{N} d_j \sin\theta_j \sin\theta_m \sin(\lambda_j - \lambda_m) \frac{\partial \psi_j}{\partial \gamma}|_{\gamma=\gamma_{mj}}, \tag{5.10}$$

$$\frac{d\mu_m}{dt} = \sum_{j=1}^{N} d_j(-\cos\theta_j \sin\theta_m + \sin\theta_j \cos\theta_m \cos(\lambda_j - \lambda_m)) \frac{\partial \psi_j}{\partial \gamma}|_{\gamma=\gamma_{mj}}, \tag{5.11}$$

where $\mu_m = \cos\theta_m$ and $\gamma_{mj} = \cos\theta_m \cos\theta_j + \sin\theta_m \sin\theta_j \cos(\lambda_m - \lambda_j)$. We use fourth order Runge-Kutta for time integration.

The description of the algorithm is as follows

- 1. Choose RBF initial centers, i.e., the initial distribution of points (θ_j, λ_j). These points will move with the fluid flow and they are carrying vorticity. The relative vorticity ζ is assigned to each point according to the initial vorticity distribution. The absolute vorticity η can be calculated by $\eta_j = \zeta_j + 2\Omega\cos\theta_j$ and it is conserved due to equation (4.10).

- 2. Compute the RBF coefficients by interpolation as in (5.3). In this step, the centers also play the role of interpolation points.

- 3. The points are evoluted by equations (5.10) and (5.11). The relative vorticity is updated by $\zeta_j = \eta_j - 2\Omega\cos\theta_j$. If the current time is not the final time, go to step 2.

Next we will show that GARBF method we used here is related to the point vortex method. Note that Kimura and Okamoto (1987) [51] write the vorticity in terms of the summation of Dirac delta function,

$$\zeta(\theta, \lambda) = \sum_{j=1}^{N} \Gamma_j \delta(\theta, \lambda, \theta_j, \lambda_j), \tag{5.12}$$

where Γ_j is the strength of the jth point vortex at (θ_j, λ_j). Substituting (5.12) into (4.39), we have the velocity in the θ direction v_θ and the λ direction v_λ

$$v_\theta = \frac{1}{\sin\theta}\frac{\partial\psi}{\partial\lambda} = -\frac{1}{4\pi}\sum_{j=1}^{N}\Gamma_j \frac{\sin\theta_j \sin(\lambda - \lambda_j)}{1 - \cos\gamma}, \tag{5.13}$$

$$v_\lambda = -\frac{\partial\psi}{\partial\theta} = -\frac{1}{4\pi}\sum_{j=1}^{N}\Gamma_j \frac{\cos\theta \sin\theta_j \cos(\lambda - \lambda_j) - \sin\theta\cos\theta_j}{1 - \cos\gamma}, \tag{5.14}$$

where γ is the central angle between point (θ, λ) and (θ_j, λ_j) as in Figure 4.1. Furthermore, since a point vortex is advected by a velocity induced by other vortices, the motion of the mth vortex is given by

$$\dot{\theta}_m = \frac{1}{\sin\theta}\frac{\partial \psi}{\partial \lambda} = -\frac{1}{4\pi}\sum_{j=1, j\neq m}^{N} \Gamma_j \frac{\sin\theta_j \sin(\lambda_m - \lambda_j)}{1-\cos\gamma_{m,j}}, \tag{5.15}$$

$$\sin\theta_m \dot{\lambda}_m = -\frac{\partial \psi}{\partial \theta} = -\frac{1}{4\pi}\sum_{j=1, j\neq m}^{N} \Gamma_j \frac{\cos\theta_m \sin\theta_j \cos(\lambda_m - \lambda_j) - \sin\theta_m \cos\theta_j}{1-\cos\gamma_{m,j}}. \tag{5.16}$$

Note that the equation system (5.10) and (5.11) from GARBF are very similar to the one (5.15) and (5.16) from point vortex method. Note that the former one converges to the later one as the Gaussians become infinitely narrow, that is, ϵ goes to infinity. This idea of interpolation the vorticity is from Beale. Next we will present several numerical experiments using the ODE system (5.10) and (5.11).

5.3 Rossby-Haurwitz Wave

As mentioned in Chapter Four, a Rossby-Haurwitz wave is an exact solution of the BVE. Here we use the same example as in section (4.3.2): $n = 1, m = 1, \nu_{mn} = \Omega$. The initial stream function is the same as in (4.65), the relative vorticity at time t is

$$\zeta(\theta, \lambda, t) = 2A\sin\theta\cos(\lambda + \Omega t), \tag{5.17}$$

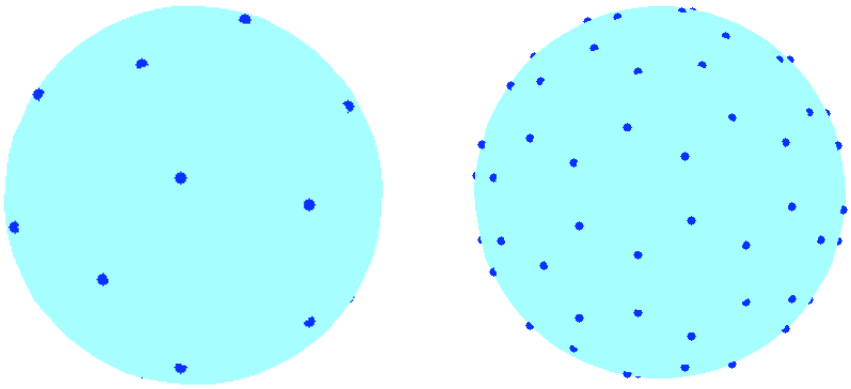
by which we can test our numerical results. Next we choose Sadourney's icosahedral points [69] as RBF centers where we expand the relative vorticity. We leave the detailed discussion of the point generation to Chapter Six, where we will discuss other choices of the point distribution on the sphere in detail. Figure 5.1 is the illustration of $N = 20$ and $N = 80$ icosahedral points on a sphere. The absolute error is defined,

$$E_{\text{Abs}}(\theta, \lambda, t) = |\zeta_{ex}(\theta, \lambda, t) - \zeta_{RBF}(\theta, \lambda, t)|, \tag{5.18}$$

where ζ_{ex} is the exact relative vorticity given by (5.17) and ζ_{RBF} is the numerical result calculated by (5.1). The error is evaluated at longitude-latitudinal grid. The relative error is defined,

$$E_{\text{rela}}(\theta, \lambda, t) = \frac{|\zeta_{ex}(\theta, \lambda, t) - \zeta_{RBF}(\theta, \lambda, t)|_\infty}{|\zeta_{ex}(\theta, \lambda, t)|_\infty}. \tag{5.19}$$

We analyze the temporal and spatial convergence of the numerical result in the following sections.



(a) (b)

Figure 5.1: Illustration of icosahedral points on the surface of the sphere with (a) $n = 20$ and (b) $n = 80$.

We can also do a rigorous error analysis since we have the exact solution of the PDE. The parameters that control the accuracy of the method include spatial discretization N, time step Δt and RBF parameter ϵ. Figure 5.2 is the absolute error of GARBF defined in equation (5.18) after one revolution with $N = 20$, $\Delta t = 4\pi/200$ and $\epsilon = 0.1$. It is impressive since there are only 20 points on sphere. Figure 5.3 shows the relative error defined by (5.19) of the vorticity obtained by GARBF after one revolution ($t = 4\pi$) when the number of timesteps is $25, 50, 100, 200, 400$. The total number of points is 20 and $\epsilon = 0.1$. We see that the relative error decreases at the rate $O((\Delta t)^4)$, which agrees with our expectation of fourth-order Runge-Kutta. On the other hand, we can see that the relative error is around 10^{-8} even though

we only use $n = 20$ points on the sphere. Figure 5.4 shows the relative error and the condition number of the RBF matrix A (see equation (2.7)) versus the RBF parameter ϵ. We can see that as ϵ decreases, the relative error first decreases, then increases. The reason is that for small ϵ, the condition number of the interpolation matrix A is large and the interpolation system is ill-conditioned. This introduces error even though theoretically the RBF method has high accuracy when ϵ is small [37].

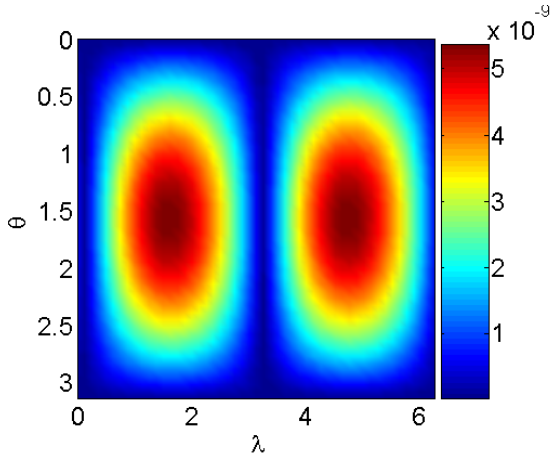


Figure 5.2: Absolute error of vorticity defined by (5.18) which is obtained by GARBF after one revolution ($t = 4\pi$) with $N = 20$, $\Delta t = 4\pi/200$, $\epsilon = 0.1$.

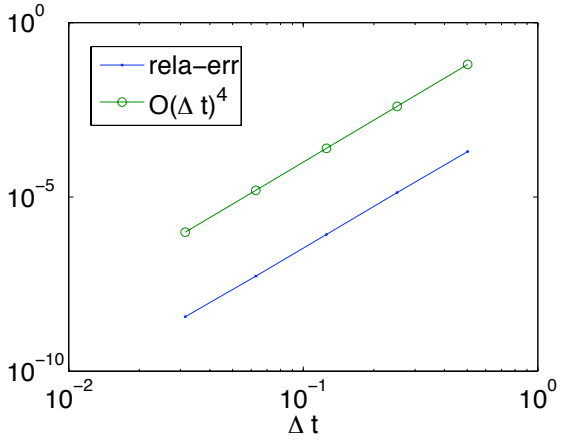
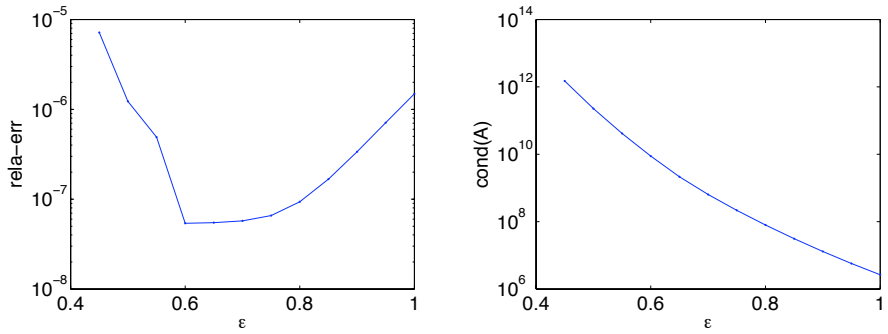


Figure 5.3: Relative error defined by (5.19) of vorticity obtained by GARBF after one revolution ($t = 4\pi$) for the number of timesteps is $25, 50, 100, 200, 400$. Total number of points is 20 and $\epsilon = 0.1$.



(a) (b)

Figure 5.4: (a): The relative error defined by (5.19) of vorticity obtained by GARBF with $\Delta t = 4\pi/200$, $n = 80$ for different ϵ; (b): the condition number of RBF matrix A.

5.4 One Vortex Patch

Our next numreical experiment is the evolution of a Gaussian vortex. For a fixed Gaussian shape parameter β, let the initial vorticity be

$$\zeta(\lambda, \theta) = \exp(-2\beta^2(1 - \cos(\lambda, \theta, \lambda_c, \theta_c))) - C_{Gauss}, \tag{5.20}$$

where (λ_c, θ_c) is the center of the vortex and the Gaussian, β controls the shape of the Gaussian, here and in the following sections, we fix $\lambda_c = \pi/3, \theta = \pi/3$ and $\beta = 4$. C_{Gauss} is the Gauss constraint as we mentioned in chapter 4,

$$C_{Gaussian} = \frac{1 - \exp(-4\beta^2)}{4\beta^2}, \tag{5.21}$$

and $\cos(\lambda, \theta, \lambda_c, \theta_c)$ is the cosine of the angle between point (λ, θ) and (λ_c, θ_c) as in (4.21)

5.4.1 Choosing RBF parameter ϵ

For RBF approximation, the numerical error is not only related to the timestep and spatial discretization, but also highly depends on the parameter ϵ. As we discussed in Chapter Two, there is a tradeoff in choosing ϵ. Large ϵ will give an inaccurate approximation. However, a small ϵ leads to an ill-conditioned RBF interpolation matrix, which will damage the accuracy of the approximation too. We choose our ϵ

by checking the accuracy of the initial vorticity interpolation. Suppose we have an RBF approximation for the above initial vorticity

$$\zeta_{\mathrm{RBF}}(\lambda,\theta)=\sum_{j=1}^{N} d_j \exp(-\epsilon^2 r(\theta,\lambda,\theta_j,\lambda_j)^2), \tag{5.22}$$

where $\theta_j, \lambda_j,\ j = 1 : N$ are N interpolation points (we call them nodes later) and $r(\theta,\lambda,\theta_j,\lambda_j)$ is the Euclidean distance between (θ,λ) and (θ_j,λ_j). We choose ϵ by experiments. Since the initial vorticity is highly concentrated around the region that is close to the center of the Gaussian, we put more nodes around this area, as in Figure 5.5. We define the relative error at $t = 0$ for RBF approximation:

$$E_2=\frac{|\zeta_{\mathrm{EX}}-\zeta_{\mathrm{RBF}}|_\infty}{|\zeta_{\mathrm{EX}}|_\infty}. \tag{5.23}$$

Table 5.1 shows the RBF approximation for fixed points $n = 170$ as in Figure 5.5 with different ϵ, which is uniform for all points. Again, it shows the tradeoff of the choice of ϵ. Small ϵ means better accuracy but worsens the condition number of interpolation matrix. Note that

$$\alpha=\epsilon h_{\min}, \tag{5.24}$$

is the relative width parameter, where $h_{\min}$ is the minimum distance between any two points.

For the nonuniform grid distribution, Fornberg and Zuev (2009) [38] show that a spatially variable shape parameter ϵ has a better performance than uniform ϵ. They took

$$\epsilon_j=\frac{1}{d_{j,\min}}, \tag{5.25}$$

where $d_{j,\min}$ is the Euclidean distance between the node (λ_j,θ_j) and its closest neighbor node. Figure 5.6 shows the absolute error for the relative vorticity with fixed $\epsilon = 6$ and variable ϵ as defined in (5.25) for points in Figure 5.5.

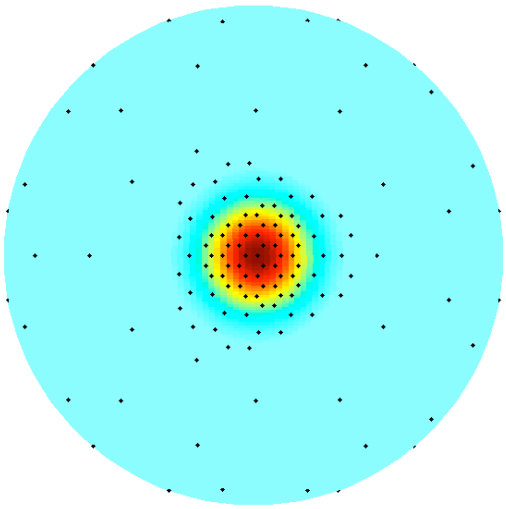


Figure 5.5: Interpolation points for initial vorticity; refined spherical triangular mesh (which will be discussed in detail in Chapter Six); number of points is 170, minimum distance between two points are 0.0451.

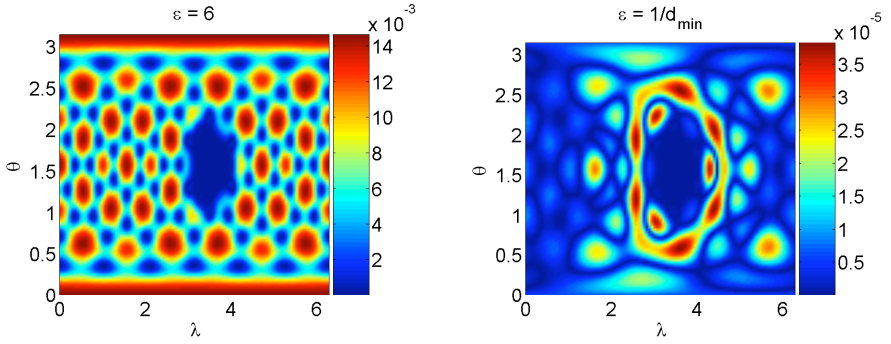
ϵ	α	E_2	cond(A)
4	0.1804	0.0071	2.8158e+015
5	0.2255	0.0123	7.3093e+012
6	0.2706	0.0149	6.0551e+010
7	0.3158	0.0157	1.2360e+009
8	0.3609	0.0159	5.1877e+007

Table 5.1: Relative Error for RBF approximation on a sphere with fixed number of interpolation points and different ϵ (uniform). α is defined in equation (5.24).

In a more recent paper, Flyer and Lehto [31] present the other choice of ϵ for nonuniform points on sphere,

$$\epsilon_j = \epsilon_{\min} \left(\frac{\max_j d_{j,\min}}{d_{j,\min}} \right), \tag{5.26}$$

and $\epsilon_{\min}$ $O(1)$ is a scaling parameter. We experiment the above ϵ choice by choosing different $\epsilon_{\min}$ and find that for the $\epsilon_{\min}$ which makes the absolute error of the initial relative vorticity acceptable, the condition number is large. Figure 5.7 shows the absolute error of the relative vorticity and Table 5.2 is the condition number. We



(a) (b)

Figure 5.6: Absolute error of relative vorticity computed in grid in Figure 5.5. (a): fixed ϵ; (b): ϵ adoptive from (5.25). Note that the maximum error occurs where the mesh changes from coarse to fine.

see that varying the ϵ value using the formula given in (5.25) still has the trade-off between accuracy and ill-conditioning.

$\epsilon_{\min}$	cond(A)
1	6.1871e+012
0.8	1.9754e+016
0.6	2.6365e+019
0.4	3.4059e+019

Table 5.2: Condition number for $\epsilon_{\min} = 1, 0.8, 0.6, 0.4$.

We conducted similar experiments for ϵ in (5.25) and found the same problem. We did the tests for time integration and had even worse results. Since the choice of the ϵ is so important for RBF methods, this will be the subject of our future work.

5.5 Two Vortex Patches on a Non-rotating Sphere

Vortex patch interaction is an interesting topic in fluid dynamics. Levy, Nair and Tufo (2009) [56] and Shin [71] experimented that the vortex patch interaction on a $\beta-$plane. Here we suppose there are two disk-shaped vortex patches on the surface of the sphere. For a non-rotating sphere, when the diameter of the disk is much less

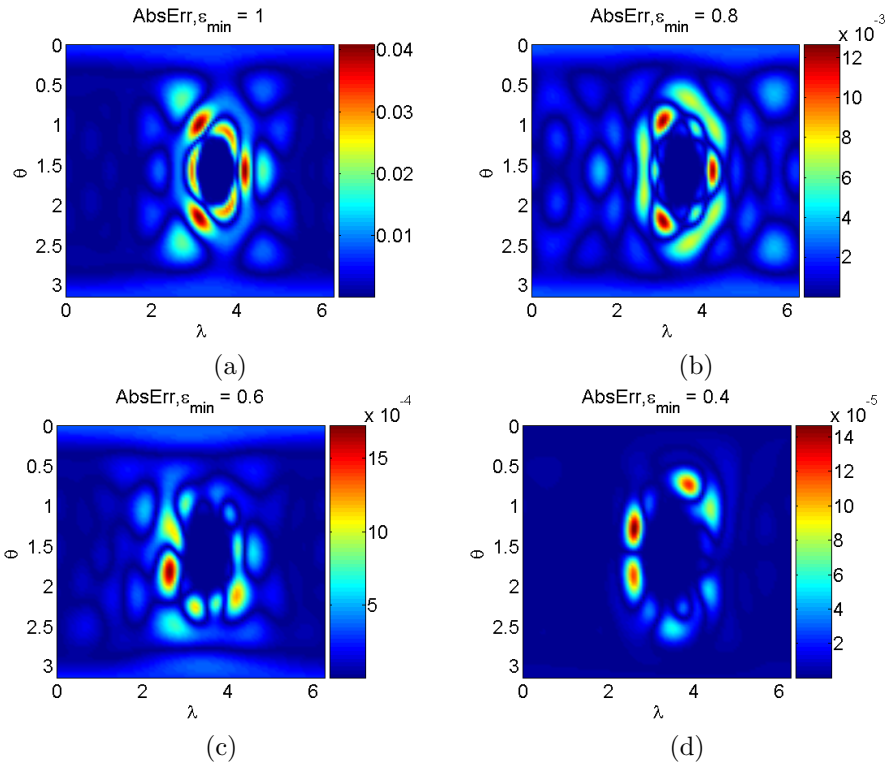


Figure 5.7: Absolute error of relative vorticity for different $\epsilon_{\min}$. Note that the maximum error occurs where the mesh changes from coarse to fine.

than the radius of the earth, we can approximate the surface of the sphere by a plane with a free boundary.

5.5.1 Initial patches

Now we explain how to get the disk-shaped patch on sphere. We first get points which form the patch on the plane then project them to the surface of the sphere by azimuthal projection

$$\boldsymbol{x}_s = \frac{\boldsymbol{x}}{|\boldsymbol{x}|}. \tag{5.27}$$

All the points are the centers of a hexagon. The center of the circular disk is one of the points that we need. The variable nNIHS controls the number of points at half side, as in Figure 5.8. For example, in Figure 5.8 (a), nNIHS equals two, that means, there are two points on the left side of the center of the circle. In Figure 5.8

(b), (c) and (d), the value of nNIHS equals 4, 8 and 16 respectively. By this way, the distance of each particle to its neighbors is the same.

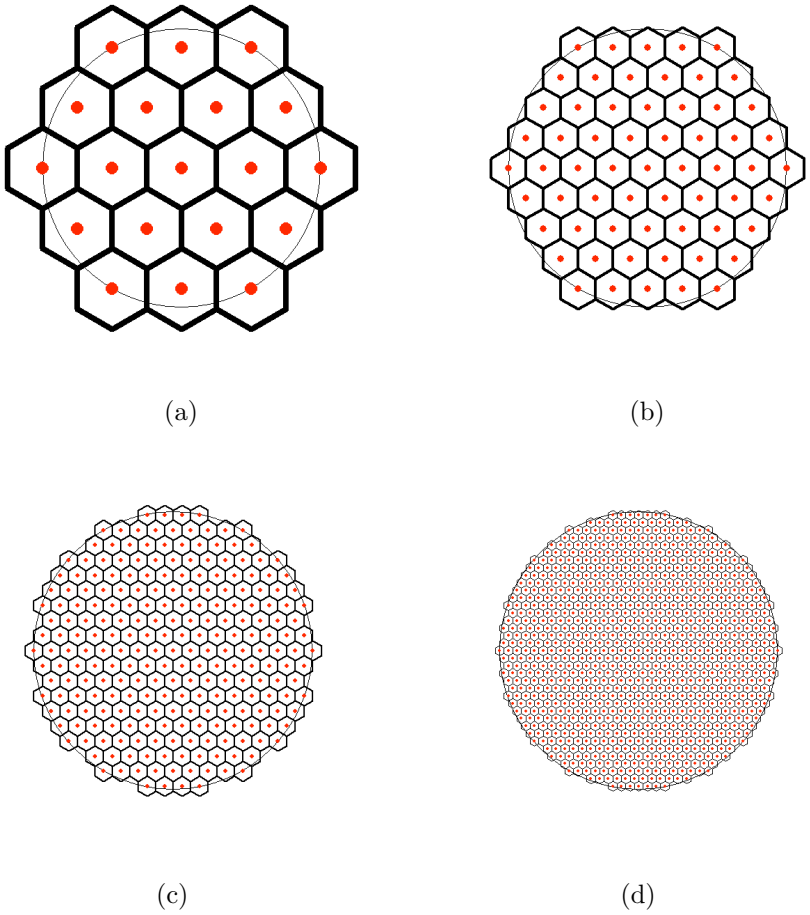


Figure 5.8: Points in a disk on a plane. The nNIHS = 2,4,8,16. Every point is at the center of a hexagon.

5.5.2 Numerical integration

In this test, we suppose two patches carry equal positive vorticities. They have same number of points $N = 1459$. The distance between the two centers is $r = 2.6$. Figure 5.9 shows how two vortex patches interact with each other at $t = 0, t = 2, t = 3$. The reason for different color of the two patches is to distinguish them. The shape of the patches agrees with the figures shown in the work of Waugh [75].

The two patch interaction test here is only limited to the non-rotating sphere. The

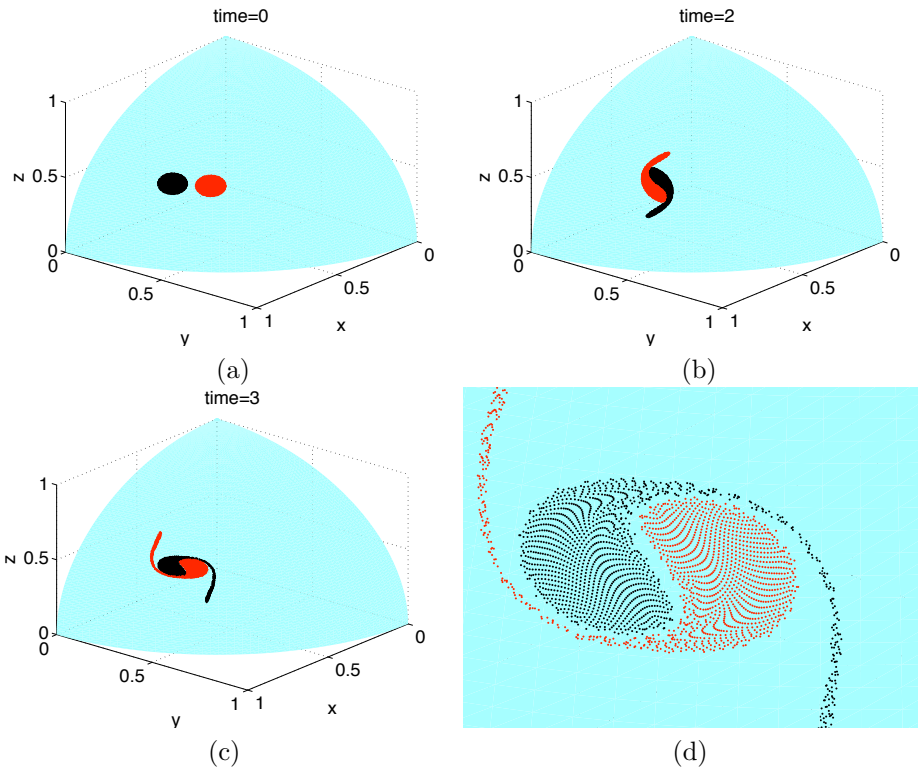


Figure 5.9: Vortex patches evolution. the ratio of the distance between two center of the patches the radius of the patch is 2.6. The total number of the particles is 2918. (a): two vortex patches at $t = 0$; (b): $t = 2$; (c): $t = 3$; (d): close up look at $t = 3$.

dynamic becomes more complicated for a rotating case since waves are generated besides the interaction of points. In this case, the local mesh we used here can not capture the behavior of the waves, which is global. Investigation two patches interacting on a rotating sphere using GARBF will be one of the future works.

In this chapter, we solve the barotropic vorticity equation by Gaussian RBF method in a Lagrangian formulation. Here, the relative vorticity is expanded by Gaussian RBF and the all centers of the RBF are moving with the flow. The method shows high accuracy for Rossby-Haurwitz wave test casee with very coarse grid spacing. Two patch interactions are tested on a non-rotating sphere.

CHAPTER VI

Solving the Barotropic Vorticity Equation by Vortex Method

6.1 Introduction

The vortex method is characterized by both considering the PDE in stream function-vorticity form and Lagrangian discretization of the vorticity. There is a variety of literature in this field [9, 21, 23, 52, 55, 68]. For problems on the surface of the earth, the application of vortex method is an active area. Zabusky and McWilliams (1982) presented a modulated point vortex model for BVE and compared the results with a finite difference model on the beta plane approximation [78]. Yao, Zabusky and Dritschel [77] compared contour surgery with pseudospectral simulations. Newton and Shokraneh (2006) investigated the dynamics of N-point vortices on a rotating unit sphere [63]. Newton and Sakajo (2007) studied the evolution of N-point vortices in a ring formation embedded in a background flow field that initially corresponds to solid-body rotation on a sphere. The evolution of the point vortices is tracked numerically as an embedded dynamical system along with the M contours which separate strips of constant vorticity [64]. Sakajo (2004) considered the motion of a vortex sheet on the surface of a unit sphere in the presence of point vortices fixed on north and south poles [70]. Sakajo (2009) presented a fast algorithm for two vortex sheets rolling-up on the surface of the sphere.

6.2 Lagrangian Fomulation of BVE

First, let us invert the Poisson equation (4.9) by expressing the stream function as a convolution of the Green's function (4.37) and the vorticity [51],

$$\psi(\boldsymbol{x},t) = -\frac{1}{4\pi}\int_{S^2} \log(1-\boldsymbol{x}\cdot\boldsymbol{x}')\zeta(\boldsymbol{x}',t)dS(\boldsymbol{x}'). \tag{6.1}$$

Applying Equation (6.1) to Equation (4.9) yields the velocity,

$$\boldsymbol{u}(\boldsymbol{x},t) = -\frac{1}{4\pi}\int_{S^2} \frac{\boldsymbol{x}\times\boldsymbol{x}'}{1-\boldsymbol{x}\times\boldsymbol{x}'}\zeta(\boldsymbol{x}',t)dS(\boldsymbol{x}'). \tag{6.2}$$

Given the velocity field $u(\boldsymbol{x},t)$, we define the flow map $\boldsymbol{x}(\boldsymbol{a},t)$ by the equation

$$\frac{\partial \boldsymbol{x}}{\partial t}(\boldsymbol{a},t) = \boldsymbol{u}(\boldsymbol{x}(\boldsymbol{a},t)), \tag{6.3}$$

where $\boldsymbol{a}$ is a set of Lagrangian parameters on the sphere. We assume that

$$\boldsymbol{x}(\boldsymbol{a},0) = \boldsymbol{a}, \tag{6.4}$$

so the flow map at $t = 0$ is the identity on the sphere. Equation (6.2) and (6.3) then yield

$$\frac{\partial \boldsymbol{x}}{\partial t}(\boldsymbol{a},t) = -\frac{1}{4\pi}\int_{S^2} \frac{\boldsymbol{x}(\boldsymbol{a},t)\times\boldsymbol{x}'}{1-\boldsymbol{x}(\boldsymbol{a},t)\cdot\boldsymbol{x}'}\zeta(\boldsymbol{x}',t)dS(\boldsymbol{x}'). \tag{6.5}$$

The next step is to change variables using the flow map, $\boldsymbol{x}' = \boldsymbol{x}(\boldsymbol{a}',t)$, which yields

$$\frac{\partial \boldsymbol{x}}{\partial t}(\boldsymbol{a},t) = -\frac{1}{4\pi}\int_{S^2} \frac{\boldsymbol{x}(\boldsymbol{a},t)\times\boldsymbol{x}(\boldsymbol{a}',t)}{1-\boldsymbol{x}(\boldsymbol{a},t)\cdot\boldsymbol{x}(\boldsymbol{a}',t)}d\Gamma(\boldsymbol{a}',t), \tag{6.6}$$

where the circulation element is defined by

$$d\Gamma(\boldsymbol{a}',t) = \zeta(\boldsymbol{x}(\boldsymbol{a}';t),t)dS(\boldsymbol{x}(\boldsymbol{a}';t)). \tag{6.7}$$

The goal is to discretize the evolution equation for the flow map (6.6). We discretize the surface of the earth by panels (spherical triangular or quadrilateral). The

panels evolve in time under the action of the flow map, and their shape changes, but they retain their identity in Lagrangian parameter space. Each panel has a particle at its center and we obtain an equation of motion for the particles by discretizing Equation (6.5) for the flow map using midpoint rule. The result is a set of ODEs for the motion of the particles,

$$\frac{d\boldsymbol{x}_j}{dt} = -\frac{1}{4\pi}\sum_{k=1,k\neq j}^{N}\frac{\boldsymbol{x}_j \times \boldsymbol{x}_k}{1-\boldsymbol{x}_j\cdot\boldsymbol{x}_k}\zeta_k A_k, \tag{6.8}$$

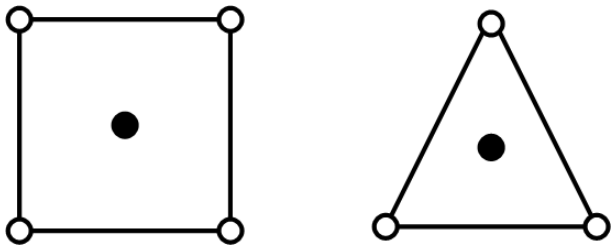
where ζ_k is the relative vorticity at kth panel and A_k is the area of the panel, which is a triangle or quadrilateral. We will discuss the area formula later. Note that $k \neq j$ means we skip the singular panels. Now, introduce two sets of points, active points and passive points. Active points are at the center of the panels and carry vorticitiy. The relative vorticity of kth panel is the same as the relative vorticity at kth active point. Passive points are at the vertices of the panels and advected by active points. Figure 6.1 shows active points and the passive points for both triangle and quadrilateral panels. The absolute vorticity is conserved for each active point (4.10), that is, $\eta_k = \eta_{k0}$, where η_{k0} denotes kth active point's initial absolute vorticity. With the help of the relation between absolute vorticity and relative vorticity $\eta_k = \zeta_k + 2\Omega z_k$, we have

$$\zeta_k + 2\Omega z_k = \zeta_{k0} + 2\Omega z_{k0}, \tag{6.9}$$

then,

$$\zeta_k = \zeta_{k0} + 2\Omega z_{k0} - 2\Omega z_k, \tag{6.10}$$

where $\zeta_{k0} + 2\Omega z_{k0}$ is the absolute vorticity of kth active point at $t = 0$. Next, we will discuss the meshes on the surface of a sphere.



(a) (b)

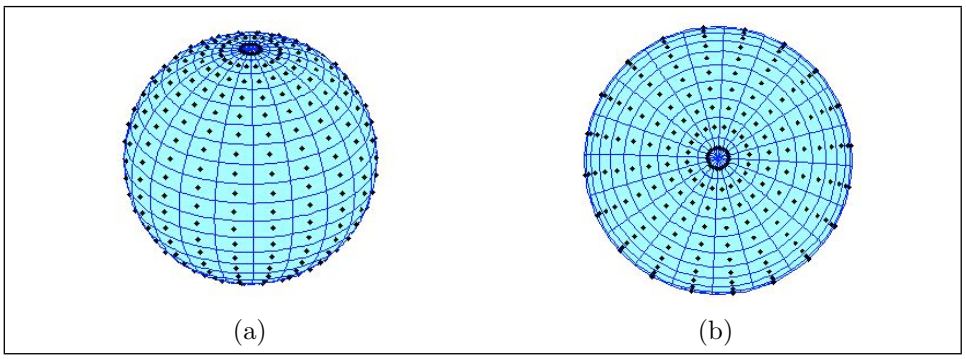
Figure 6.1: Active points (•) and passive points (○) of a triangle and quadrilateral panel.

6.3 Meshes on the Surface of the Sphere

We need to discretize the flow map equation (6.6) on sphere. The first question is: what is the mesh? Here we investigate four different types of meshes on the surface of the sphere: Longitude-Latitude (LL), Icosahedral Triangle (IT), Icosahedral Hexagon (IH) and Cubed Sphere (CS).

6.3.1 Longitude-latitude (LL)

The longitude-latitude mesh is the most nature mesh for the spherical geometry. However, we can see from Figure 6.2 that the mesh introduces the polar singularity, that is, all the meridian lines converge to points on both the south and the north poles. Points are clustered around the poles, on the other hand, equatorial resolution is much poorer.



(a) (b)

Figure 6.2: Longitude-Latitude Panel. (a): side view; (b): view from north pole. Points are clustered at the pole area.

6.3.2 Icosahedral triangles (IT)

First we project the 12 vertices of an icosahedron to a unit sphere. The Cartesian coordinates define the vertices of an icosahedron with edge-length 2, centered at the origin:

$$(0, \pm 1, \pm gr), (\pm 1, \pm gr, 0), (\pm 1, 0, \pm gr)$$

where $gr = (1 + \sqrt{5})/2$ is the golden ratio. For the IT mesh and the following icosahedral hexagon and cubed-sphere mesh, we will use a parameter L which controls the level of global refinement and the total number of panels N. We call the projection points of 12 vertices of the icosahedron the vertices of icosahedral triangles. The first level $L = 0$ of the IT mesh is obtained by connecting the vertices with their closest points using great circles, as in the first graph in Figure 6.4 [69, 76]. The center c of the triangle is obtained by averaging 3 vertices $v_i, i = 1, 2, 3$,

$$\tilde{c} = \frac{v_1 + v_2 + v_3}{3}, \tag{6.11}$$

where v_1, v_2, v_3 are the Cartesian coordinates of three vertices, then projecting $\tilde{c}$ to the unit sphere

$$c = \frac{\tilde{c}}{|\tilde{c}|}. \tag{6.12}$$

This mesh can be nested refined by connecting the center of each edge by great circles as shown in Figure 6.3. For example, v_{12} is the center of the edge $v_1 v_2$. The coordinate of the new center c_1 is the average of three vertices v_c, v_{12}, v_{31} by equation (6.11) then project to the sphere by equation (6.12). Figure 6.4 and Figure 6.5 shows the global and local refinement. The red and the yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed. Figure

6.6 shows the ratio of the minimum area of maximum area for each global refinement, the mesh becomes more uniform as the number of points increases.

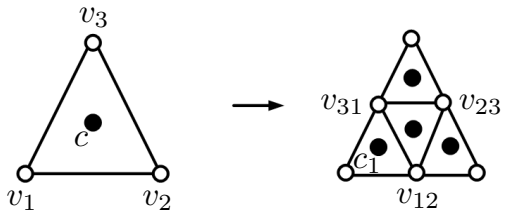


Figure 6.3: Icosahedral triangle mesh refinement stencils. For example, new vertices v_{12} is the center of the edge v_1v_2. The coordinate of the new center c_1 is the average of three vertices v_1, v_{12}, v_{31} by equation (6.11) then project to the sphere by equation (6.12).

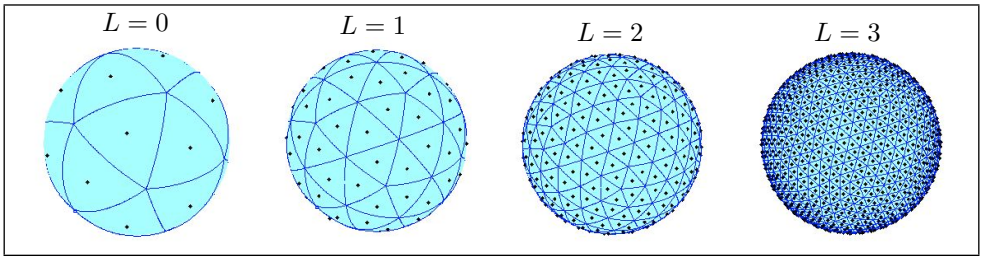


Figure 6.4: Icosahedral triangles. From left to right: $L = 0, 1, 2, 3$ and $n = 20, 80, 320, 1280$. Vertices are projections from 12 vertices of an icosahedron to a unit sphere.

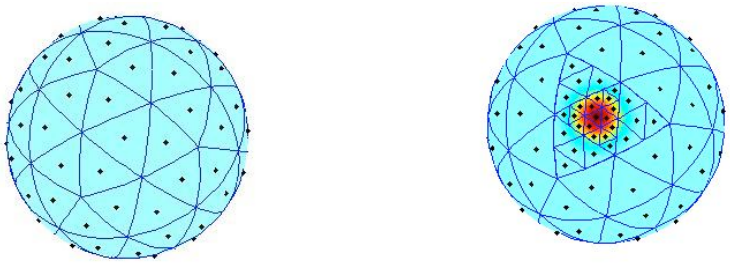


Figure 6.5: Icosahedral triangles with local refinement. The red and yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed.

6.3.3 Icosahedral hexagon (IH)

The icosahedral hexagon is a dual mesh for icosahedral triangle mesh, as in Figure 6.7. It is also called spherical geodesic mesh in [67]. The IH mesh is built upon IT mesh in the following way. Suppose we created the icosahedral triangle mesh, then

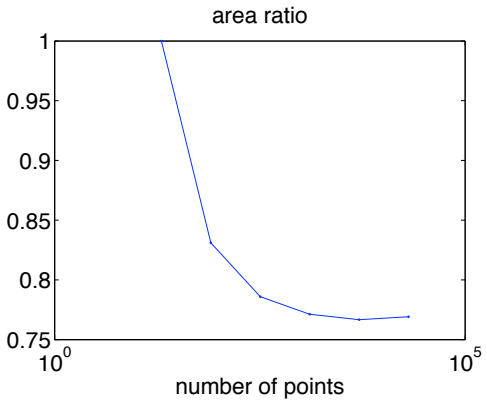


Figure 6.6: The ratio of minimum area and maximum area in Icosahedral triangle mesh. This shows the mesh becomes more uniform as the number of points increases.

connect the centers of each neighbor panel, where neighbor means that the panels share the same edge, that forms the IH panels. The difficulty with this icosahedral hexagon mesh is that it is hard to implement nested refinement. We can refine the hexagon to triangles by connecting the centers and the vertices of the hexagon as an alternative way. This is the subject of future work.

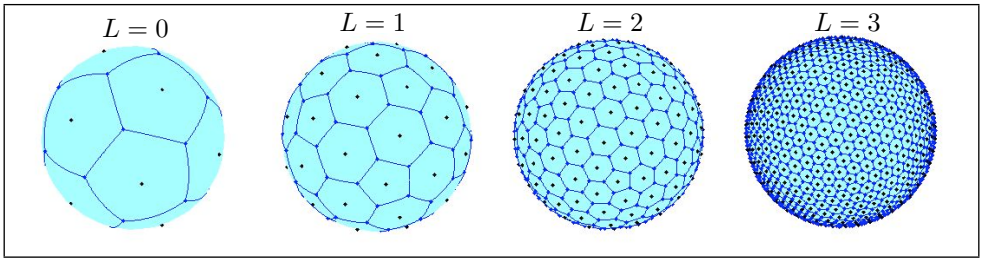


Figure 6.7: Icosahedral hexagon. From left to right: $L = 0, 1, 2, 3$ and $n = 12, 42, 162, 642$. The vertices of the hexagon/pendagon are the centers of the spherical triangle in figure (6.4).

6.3.4 Cubed-sphere (CS)

The last spherical grid which we shall consider here is the cubed-sphere mesh. Figure 6.8 shows a circumscribed cube in a sphere, where eight vertices of the cube are on the sphere. The cubed-sphere mesh is defined by a radial projection from the center of the cube inscribed into the sphere (a so-called "gnomonic projection"). There are various versions of the cubed-sphere mesh. First we will explain the method

in [65] and then the approach that implemented in this thesis.

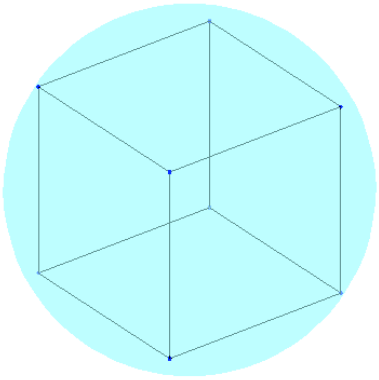


Figure 6.8: a circumscribed cube in a sphere

6.3.4.1 gnomonic projection

A local Cartesian coordinate system is determined for each of the six faces of the cube. Suppose the local coordinates are $(x, y)_i$, which rang from $[-a, a]$ for cube faces $i \in 1:6$ where $a = \frac{\sqrt{3}}{3}$ for unit sphere. Then the spherical coordinates (X, Y, Z) for the local face centered in the $-Y$ is

$$(X, Y, Z) = \frac{1}{r}(x, -a, y), \ r = \sqrt{a^2 + x^2 + y^2}, \tag{6.13}$$

By this way, each point on the cube is connected to a point on the spherical surface.

6.3.4.2 chord projection

In numerical tests in this thesis, we use a different way to generate the cubed-sphere mesh. Let us start with eight vertices of a circumscribed cube as in Figure 6.8. The first level of panels is obtained by connecting the vertices which share the same side by great circles. Figure 6.9 shows the refinement stencils of the cubed-sphere mesh. For example, new vertices v_{12} is the center of the edge v_1v_2. The coordinate of the new center c_1 is the average of three vertices v_1, v_{12}, v_c, v_{41} by equation (6.11) then project to the sphere by equation (6.12), where v_c is at the same position as the

center c. Figure 6.10 is the global refined cubed-sphere mesh with $n = 6, 24, 96, 384$. Figure 6.11 shows local refinement of the cubed-sphere mesh, where the red and yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed.

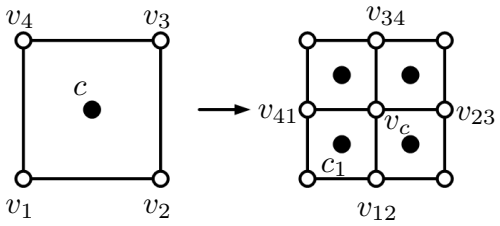


Figure 6.9: Cubed-sphere mesh refinement stencils. For example, new vertices v_{12} is the center of the edge v_1v_2. The coordinate of the new center c_1 is the average of four vertices v_1, v_{12}, v_c, v_{41} by equation (6.11) then project to the sphere by equation (6.12), where v_c is at the same position as the center c.

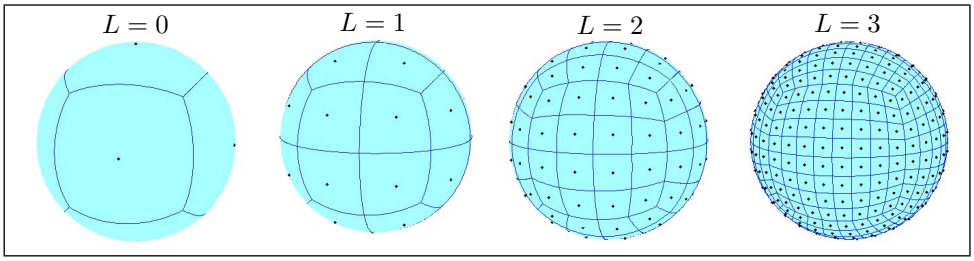


Figure 6.10: Cubed-sphere mesh with $n = 6, 24, 96, 384$. The first eight vertices of quadrilateral panels are the vertices of a circumscribed cube.

Figure 6.11: Cubed-sphere local refinement. The red and yellow colors indicate some local features that we want to resolve and where local mesh refinement is needed.

In this chapter, the numerical tests are performed in both icosahedral triangle mesh and cubed-sphere mesh. Next, we will talk about the area of the panels on the surface of the sphere.

6.4 Area for Panels on Sphere

As we mentioned before, the midpoint method is applied in evaluating the Biot-Savart integral (6.5). Then we need a way to calculate the area for panels on sphere in equation (6.8). The Cartesian coordinates of the vertices of each panel are available. Let A, B and C be row vectors which contain the Cartesian coordinates of the vertices of a spherical triangle as shown in Figure 6.12, that is, $A = (x_A, y_A, z_A)$, $B = (x_B, y_B, z_B)$, $C = (x_C, y_C, z_C)$. α, β, γ are angles of spherical triangle. The sides of the triangle a, b and c are part of great circles. Since A, B, C are unit vectors, we have

$$\cos a = B \cdot C; \tag{6.14}$$

$$\cos b = A \cdot C; \tag{6.15}$$

$$\cos c = A \cdot B; \tag{6.16}$$

We suppose a, b, c are less than π so we take arccos directly. The formula for the spherical excess E is presented in ([1]) (Pg. 468-469),

$$\tan \frac{E}{4} = \sqrt{\tan \frac{s}{2} \tan \frac{s-a}{2} \tan \frac{s-b}{2} \tan \frac{s-c}{2}}, \tag{6.17}$$

where $s = (a + b + c)/2$. There is a relation between the spherical excess E and the area of a spherical triangle ([1])

$$\text{Area} = \mathrm{R}^2\mathrm{E}, \tag{6.18}$$

where R is the radius of the sphere ($R = 1$ in our case). That is the way that we obtain the area of the panel from the Cartesian coordinates of the vertices of the panel.

Four panels have other shapes, we first decompose them to triangles then sum the area of triangles. For example, the area of the quadrilateral is computed by treating

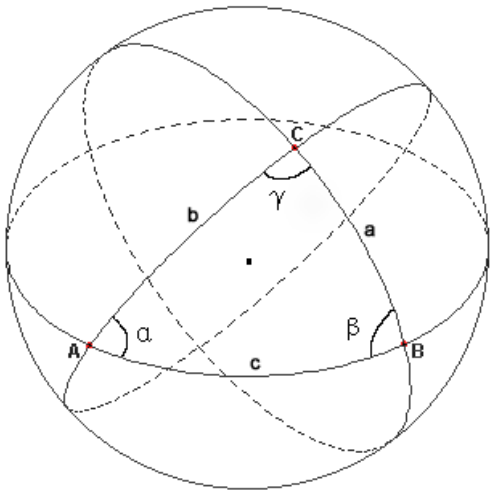


Figure 6.12: Angles of the spherical triangles.

the quadrilateral as two spherical triangles and applying the formula (6.18) twice. In our numerical tests, the spherical triangular area formula (6.18) will be applied in equation (6.8). Next, we will discuss the regularized approximation of the Green's function that we applied in the numerical experiments.

6.5 Regularized Approximation

As we discussed in Chapter Four, the point vortex methods approximate the vorticity by the Dirac delta function. The vortex blob method has more advantages since the basis functions have more overlap. Here we use a regularized Green's function (4.41) as we introduced in Chapter Four, the stream function ψ_δ

$$\psi_\delta(\boldsymbol{x},t) = -\frac{1}{4\pi}\int_{S^2} \log(1-\boldsymbol{x}\cdot\boldsymbol{x}'+\delta^2)\zeta(\boldsymbol{x}',t)dS(\boldsymbol{x}'), \tag{6.19}$$

where $\delta > 0$ is a small number. Discretization on the surface of the sphere leads to

$$\psi_\delta(\boldsymbol{x}_i,t) = -\frac{1}{4\pi}\sum_{j=1}^{N} \log(1-\boldsymbol{x}_i\cdot\boldsymbol{x}_j+\delta^2)\zeta(\boldsymbol{x}_j,t)A_j, \tag{6.20}$$

where $\boldsymbol{x}_j$ and A_j are the center and the area of the jth panel. The ODE system for the flow map becomes

$$\frac{d\boldsymbol{x}_j}{dt} = -\frac{1}{4\pi}\sum_{k=1,k\neq j}^{N}\frac{\boldsymbol{x}_j \times \boldsymbol{x}_k}{1-\boldsymbol{x}_j\cdot\boldsymbol{x}_k+\delta^2}\zeta_k A_k. \tag{6.21}$$

For future referece, the approximate vorticity is

(6.22)

$$\begin{aligned}
-\zeta_\delta(\theta,\lambda) &= \Delta_s\psi_\delta(\theta,\lambda)\\
&= \Delta_s\left(-\frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi}\log(1-\cos(\theta,\lambda,\theta',\lambda')+\delta^2)\zeta(\theta',\lambda')d\theta'd\lambda'\right)\\
&= -\frac{1}{4\pi}\left(\int_0^{2\pi}\int_0^{\pi}\Delta_s\log(1-\cos(\theta,\lambda,\theta',\lambda')+\delta^2)\zeta(\theta',\lambda')d\theta'd\lambda'\right)\\
&= -\frac{1}{4\pi}\left(\int_0^{2\pi}\int_0^{\pi}\left(\frac{-(1-\cos(\theta,\lambda,\theta',\lambda'))^2+2\delta^2\cos(\theta,\lambda,\theta',\lambda')}{(1-\cos(\theta,\lambda,\theta',\lambda')+\delta^2)^2}\right)\zeta(\theta',\lambda')d\theta'd\lambda'\right)\\
&\approx -\frac{1}{4\pi}\sum_j\left(\left(\frac{-(1-\cos(\theta,\lambda,\theta_j,\lambda_j))^2+2\delta^2\cos(\theta,\lambda,\theta_j,\lambda_j)}{(1-\cos(\theta,\lambda,\theta_j,\lambda_j)+\delta^2)^2}\right)\zeta_j A_j\right),
\end{aligned}$$

where $\zeta_j = \zeta(\theta_j,\lambda_j)$ and A_j is the area of the jth panel. The relative vorticity at any point of the sphere can be calculated by equation (6.22). This formula is useful when we apply the remeshing strategy, which we will discuss later.

Next, we start the numerical experiments. The first one is the Rossby-Haurwitz wave.

6.6 Rossby-Haurwitz Wave Experiments

As we mentioned in Chapter Four, the Rossby Haurwitz (RH) wave is an exact solution for the BVE. It is a reasonable first test case for our numerical method. Here and in the following subsections, we focus on the simplest case as in Chapter Four, that is $m = 1, n = 1$ and amplitude $A = 0.05$ RH wave. Without loss of generality, we fix $\Omega = 1/2$, then $t = 4\pi$ represents one revolution. Fourth-order Runge-Kutta is used for time integration.

6.6.1 Definition of error

Since the Rossby-Haurwitz wave is the exact solution of the BVE, we can investigate the errors. For the stream function ψ, let us the define relative error

$$\text{RelaErr} = \frac{||\psi_{\text{ex}} - \psi_\delta||_\infty}{||\psi_{\text{ex}}||_\infty}. \tag{6.23}$$

For the position of particles, we suppose $P_i = (P_{xi}, P_{yi}, P_{zi})$ and $Pe_i = (Pe_{xi}, Pe_{yi}, Pe_{zi})$ as our numerical position and exact position at time t for point i. Let

$$\text{AbsErr} = \max_{\text{i}} ||\text{P}_\text{i} - \text{Pe}_\text{i}||_\infty \tag{6.24}$$

be our error criteria.

6.6.2 Mesh comparison

We apply three different meshes: icosahedral triangle, icosahedral hexagon and cubed-sphere to calculate the evolution of the RH wave. Figure 6.13 shows the absolute error for the particle position after one revolution. The error is comparable for the three meshes. Then we will focus on the icosahedral triangle mesh for the remaining Rossy-Haurwitz wave tests.

6.6.3 Timestep experiments

Table 6.1 and 6.2 show the absolute error defined in (6.24) for fixed number of points $N = 320, N = 1280$ respectively. We can see that the error converges to a saturation error which is due to the spatial discretization. For the later experiments with the RH wave, we fix $\Delta t = 4\pi/1600$. Figure 6.14 shows the fourth-order convergence in time for the difference between absolute error and the saturation error, where we choose the error from $\Delta t = 4\pi/3200$ as the saturation error.

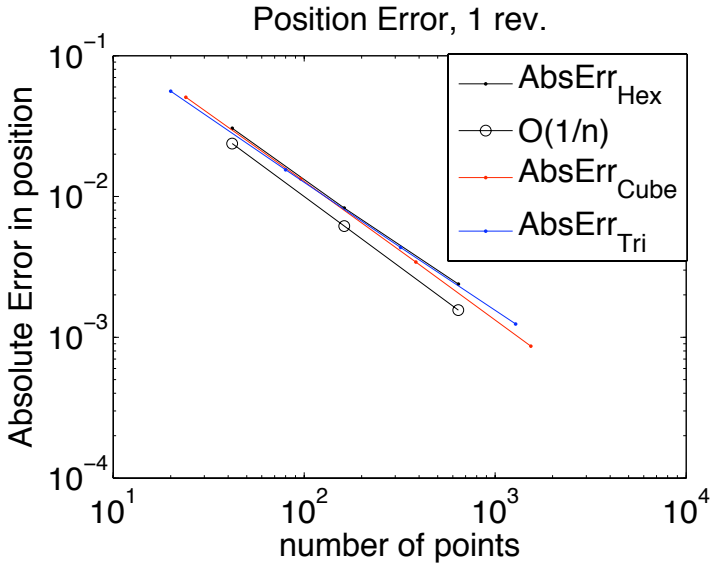


Figure 6.13: Absolute error in particle positions after one revolution for different meshes: icosahedral triangle, icosahedral hexagon and cubed-sphere for RH wave. The error is comparable for the three meshes. All three lines are parallel to the line $O(1/n)$, which means the vortex method we use here is first order accurate in terms of the number of points.

6.6.4 Test spatial error

Table 6.3 shows the absolute error defined in (6.24) for the fixed timestep $\Delta t = 4\pi/1600$. Figure 6.15 shows that the numerical method is first order accurate in the number of the panels on sphere.

Figure 6.16 is the trajectory of particles after five revolutions with $N = 80, \Delta t = 4\pi/1600$, which is agree to the one we obtained from exact ODE equations (4.72) to (4.74) and Figure 4.6.

6.6.5 Stream function

The initial stream function is given by (4.61). We first solve the stream function given by (6.1) using the midpoint numerical integral based on spherical triangle (IT) mesh, then compare it with the exact one (4.61). Figure 6.17 shows that the relative error (6.23) is almost proportional to the area of the spherical triangular panel, which agrees with the fact that the midpoint rule is a second-order method. Figure 6.18

N	number of time step	dt	AbsErr
320	50	0.25132741228718	0.00434076420937
320	100	0.12566370614359	0.00433954877181
320	200	0.06283185307180	0.00433947278137
320	400	0.03141592653590	0.00433946803681
320	800	0.01570796326795	0.00433946774052
320	1600	0.00785398163397	0.00433946772201
320	3200	0.00392699081699	0.00433946772085

Table 6.1: Absolute Error for particle positions after one revolution, N = 320. The error saturates around 0.004339 due to the spatial discretization.

shows the exact and approximate stream function given by equation (6.20) for RH wave at $t = 0, t = 2\pi, t = 3\pi, t = 4\pi$ with $N = 80$ and $\delta = 0.02$. Note that $\delta = 0.02$ is just one example of the choice of regularized parameter. The investigation of the effect of the value of δ is one of the future works.

6.6.6 Nonuniform mesh test

There is a problem that is related to the changing of mesh size in numerical computation of wave problem. A fine mesh can resolve finer scale waves compared to the coarse mesh. When the finer scale waves travel to the coarse region, they can not be resolved correctly. Furthermore, waves may also be reflected during the jump of the mesh scale. Hence grid size jumps can cause new numerical errors. Here we test the RH wave error on the meshes in Figure 6.19, in which the meshes are getting more uniform from left to right. Figure 6.20 shows the absolute error defined in (6.24) for both a uniform mesh as in Figure 6.10 and nonuniform meshes as in Figure 6.20. The error for nonuniform meshes decreases when the mesh gets more

N	number of time step	dt	AbsErr
1280	50	0.25132741228718	0.00124543628511
1280	100	0.12566370614359	0.00124419067101
1280	200	0.06283185307180	0.00124411317947
1280	400	0.03141592653590	0.00124410835356
1280	800	0.01570796326795	0.00124410805258
1280	1600	0.00785398163397	0.00124410803379
1280	3200	0.00392699081699	0.00124410803262

Table 6.2: Absolute error for particle positions after one revolution, N = 1280. The error saturate around 0.001244 due to the spacial discretization.

N	AbsErr
20	0.05602181188586
80	0.01549499404301
320	0.00433946772201
1280	0.00124410803379

Table 6.3: Absolute error for particle position after one revolution with fixed $\Delta t = 0.00785$ in icosahedral triangular mesh for different spacial resolution $N = 20, 80, 320, 1280$.

uniform. Define

$$\text{ratio} = \frac{\text{AbsErr}_{\text{CR}}}{\text{AbsErr}_{\text{C}}}, \tag{6.25}$$

where $\text{AbsErr}_{\text{CR}}$ denotes the absolute error calculated on a cubed-sphere refined mesh in Figure 6.19 and AbsErr_{C} denotes the absolute error calculatd on a uniform cubed-sphere mesh in figure 6.10. Figure 6.20 (b) shows the ratio of the error defined in (6.25). When the ratio less then one, it means that the nonuniform mesh has larger error than the uniform mesh. However, we also see that the nonuniform mesh has smaller errors when $N = 456$ and $N = 1572$, which means that the non-uniformity of the mesh does not cause severe problem in our method.

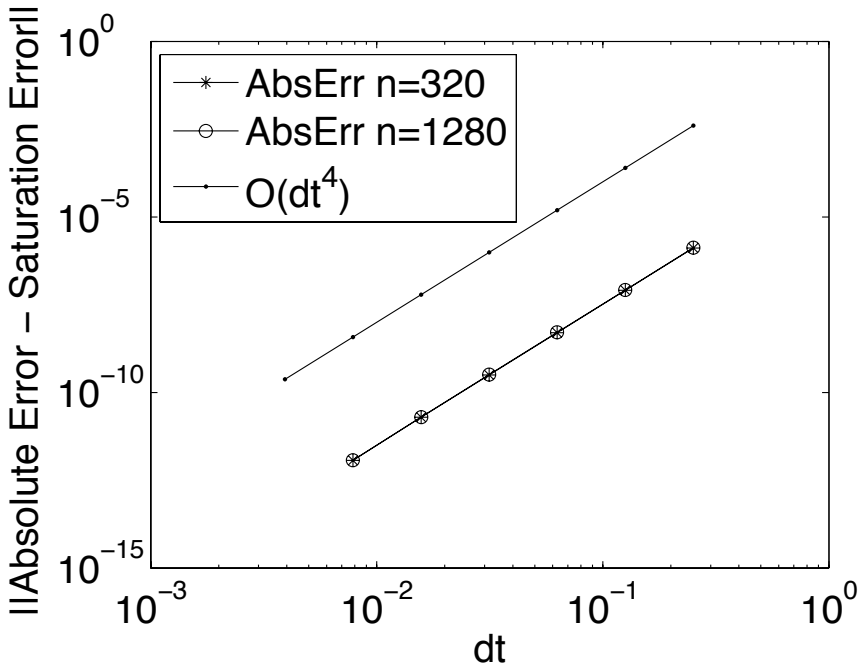


Figure 6.14: Plot $|\text{AbsErr} - \text{SaturationError}|$ versus dt, which shows the 4th order accuracy of RK4 with $N = 320, 1280$. star: $N = 320$; circle: $N = 1280$.

Here, we finish the numerical tests for the Rossby-Haurwitz wave. Now we will focus on problems that don't have an analytical solution, but based around physical phenomenon [56]: vortex patch evolution. Before we start the numerical experiments, we first discuss the adaptive mesh refinement strategy that we will apply.

6.7 Adaptive Mesh Refinement (AMR)

For computational fluid dynamics, adaptive local mesh refinement is an attractive strategy since it offers better spatial resolution for resolving small scale features and does not require a fine mesh for the whole computational domain [13, 14, 47, 48]. Here we will discuss the AMR for vortex method [30] and focus on local refinement for the icosahedral triangle and the cubed-sphere mesh. We use two refinement criteria ϵ_Γ and ϵ_d. When the circulation of the panel is larger than ϵ_Γ or the panel's side length is larger than ϵ_d, the panel is divided into four smaller ones as shown

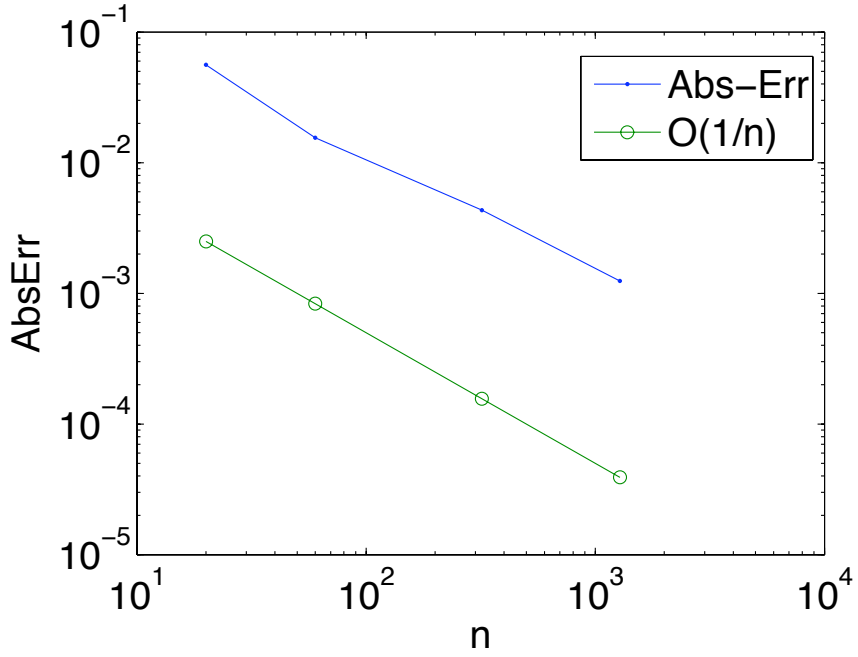


Figure 6.15: RH wave: Absolute error for particle position after one revolution in spherical triangular mesh for $\Delta t = 4\pi/1600$. n is the number of the panels on sphere.

in Figure 6.3 and 6.9 for triangular and quadrilateral mesh respectively. Here we explain the quadrilateral refinement as an example, v_1, v_2, v_3, v_4 are the vertices of the panel and also passive points, c is the center of the panel and the active points. $v_{12}, v_{23}, v_{34}, v_{41}, v_c$ are the new vertices and passive points, where v_c is the center c that before the refinement. The coordinates and the absolute vorticity of v_{12} is obtained by averaging the coordinates and the absolute vorticity of v_1 and v_2 then projected to the surface of the sphere. The same strategy is applied to v_{23}, v_{34} and v_{41}. The coordinates and the absolute vorticity of c_1 are obtained by averaging coordinates and absolute vorticity of v_1, v_{12}, v_c and v_{41} then projecting to the unit sphere. Similar strategy is applied to the triangle panel refinement.

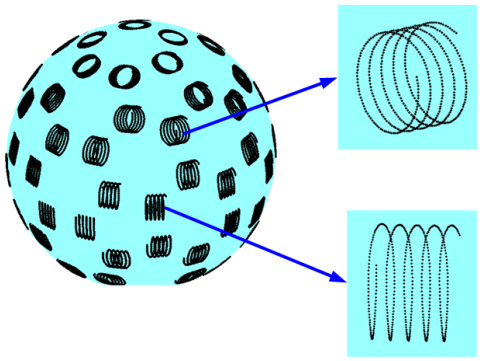


Figure 6.16: Trajectory of particles after five revolutions with $N = 80, dt = 4\pi/1600$, which is agree to the one we obtained from exact ODE equations (4.72) to (4.74) and figure (4.6).

6.8 Gaussian Vortex Tests

Suppose the initial vorticity is a Gaussian centered at $\boldsymbol{x}_c = (x_c, y_c, z_c)$

$$\zeta(\boldsymbol{x}, \boldsymbol{x}_c) = \exp(-2\epsilon^2(1 - \boldsymbol{x} \cdot \boldsymbol{x}_c)) - C_{Gauss}, \tag{6.26}$$

where ϵ controls the width of the Gaussian. We fix $\epsilon = 4$ in the following sections. C_{Gauss} is the Gauss constraint ([18]), which is a constant that only depends on ϵ,

$$C_{Gaussian} = \frac{1 - \exp(-4\epsilon^2)}{4\epsilon^2}, \tag{6.27}$$

which guarantees that the total vorticity over the surface of the sphere is zero. Since the initial vorticity is highly non-uniform, local adaptive mesh refinement is necessary. Here we employ three user-specified parameters $L, \epsilon_\Gamma, \epsilon_d$ to control the mesh. The goal is to have a relatively fine mesh at the place where small-scale events develop. We start with a uniform mesh with L levels. L should not be too small or too large. Large L means heavy computational cost, however, if L is too small, it can not catch the features of the Gaussian. Parameter ϵ_Γ is criterion for the circulation of the panel. Recall the procedure for panel refinement was explained in section 6.3, Figure 6.9 and Figure 6.3. The panel is refined to several smaller ones if the circulation is larger than ϵ_Γ. As time goes on, the panels are stretched by the flow and if the

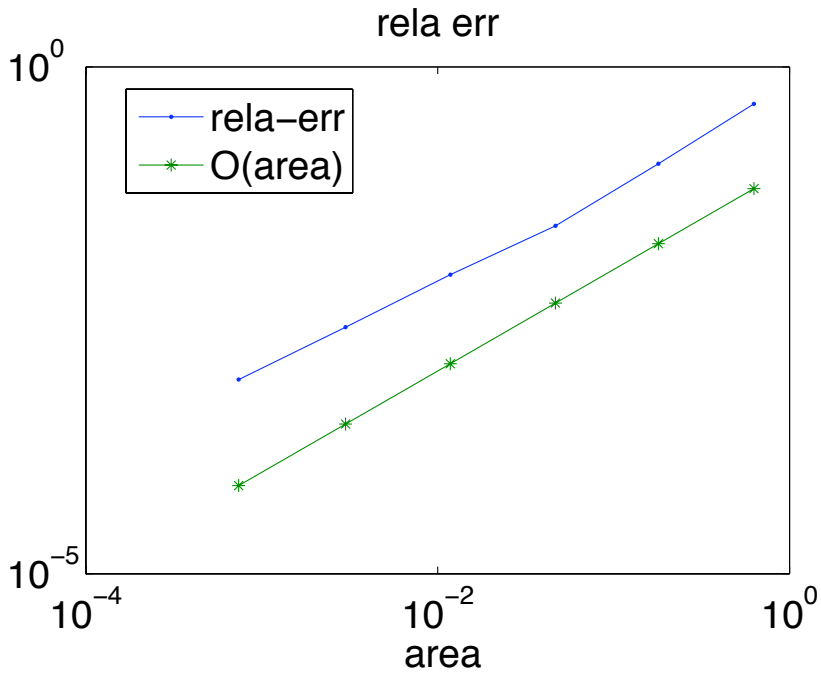


Figure 6.17: Relative error (6.23) of the stream function (6.1) using midpoint numerical integral based on spherical triangle (ST). Area means the maximum area of the spherical triangle since they are not exactly the same.)

length of any side is larger than ϵ_d, the panel is refined. The investigation of other refinement criterion is the subject of future work. Table 6.4 shows the parameters we use for the test cases involving the Gaussian patch. First we test the non-rotating sphere then the rotating one.

6.8.1 Non-rotating sphere

When the background sphere is not rotating, that is, $\Omega = 0$, the Gaussian patch is self-rotating. Figure 6.21 shows the test case 1 in table 6.4. The center of the patch is at $(0, 1, 0)$. We can see from the number of panels N that under the same refinement criteria, there is more refinement for the icosahedral triangle mesh than for the cubed-sphere mesh. The patch preserves the symmetric shape as time goes on. We also observe the artificial diffusion in this test case, which is due to the linear interpolation we used here for adaptive refinement. High order interpolation scheme

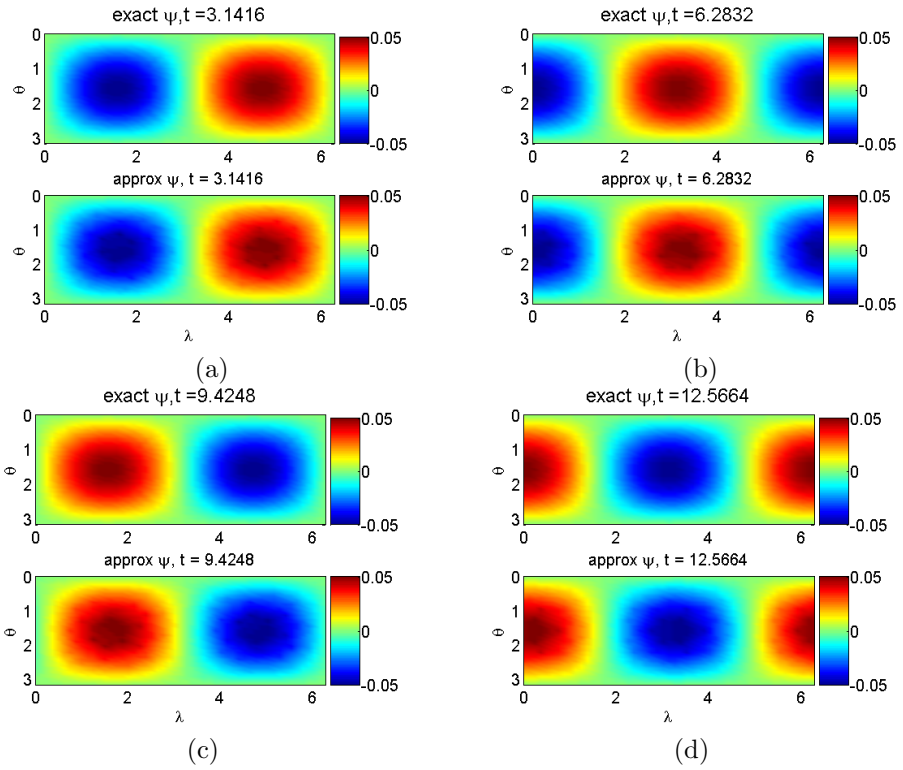


Figure 6.18: Exact and approximate (6.20) stream function for RH wave at $t = 0, t = 2\pi, t = 3\pi, t = 4\pi$ with $n = 80$ and $\delta = 0.02$

will be employed in the future.

6.8.2 Rotating sphere

Figure 6.22 shows the evolution of the Gaussian patch at $t = 0, t = 2\pi, t = 4\pi$ for cubed-sphere and icosahedral triangle mesh respectively with $\Omega = 1/2, L = 4, \epsilon_\Gamma = 0.0002, \epsilon_d = 0.02$ (test case 2 in table 6.4). In both cubed-sphere and icosahedral triangle mesh, the Gaussian patch moves to the northwest direction due to the Coriolis force, which is agree to our expectation [56].

6.8.3 Convergence check

We test the Gaussian patch on a rotating sphere $\Omega = 1/2$ after one revolution $t = 4\pi$ by systematically changing the refinement criterion ϵ_Γ, leaving L and ϵ_d fixed. Figure 6.23 shows the Gaussian vortex patch after one revolution with fixed

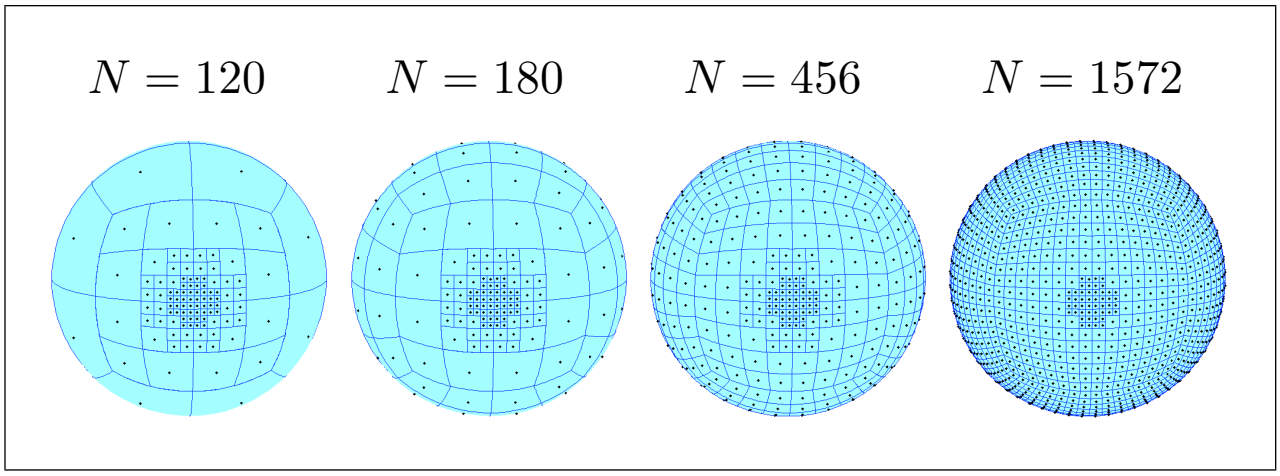
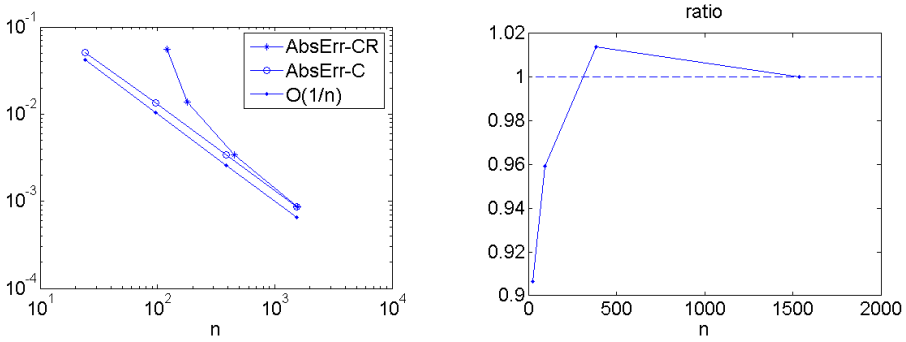


Figure 6.19: Meshes for $N = 120, 180, 456, 1572$ for testing the effect of mesh non-uniformity.



(a) (b)

Figure 6.20: (a): absolute error for particle positions for uniform cubed-sphere mesh as in figure (6.10) and nonuniform cubed-sphere meshes as in figure (6.19); Right : ratio of two absolute errors defined in (6.25).

$L = 4, \epsilon_d = 0.02$ and $\epsilon_\Gamma = 0.001, 0.0005, 0.0002, 0.0001$. It shows that the shape of the patch converges to a similar pattern. More rigorous convergence check will be done in the future.

6.9 Two Gaussian Vortices Test

Suppose now we initially have two Gaussian patches centered at $(x_{c1}, y_{c1}, z_{c1}) = (-0.3670, 0.8548, 0.3670)$ and $(x_{c2}, y_{c2}, z_{c2}) = (0.3670, 0.8548, 0.3670)$.

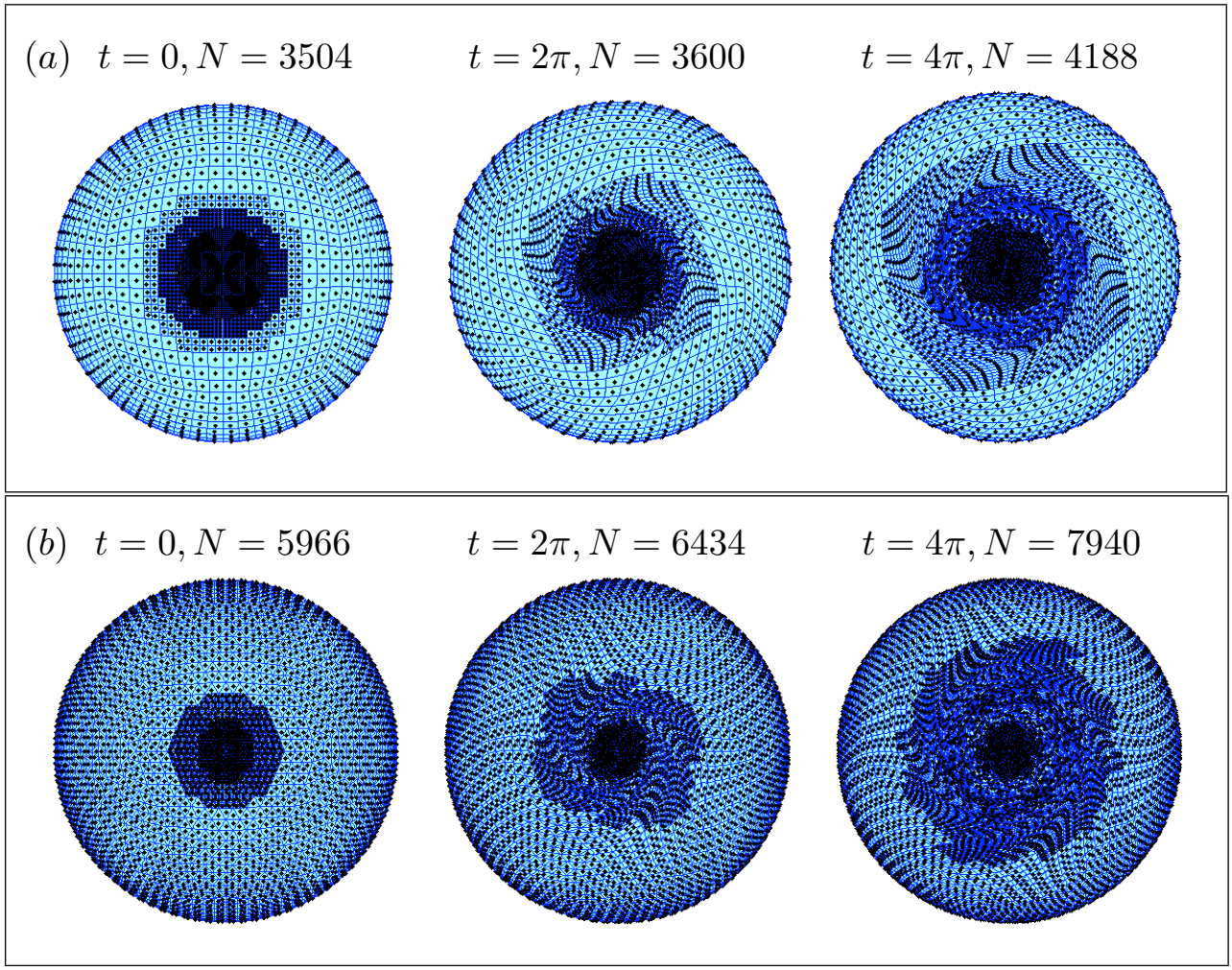


Figure 6.21: Gaussian patch test case 1 at $t = 0, 2\pi, 4\pi$ on a nonrotating sphere using (a) cubed-sphere mesh and (b) icosahedral triangle mesh. The patch is self-rotating and preserves the symmetric shape.

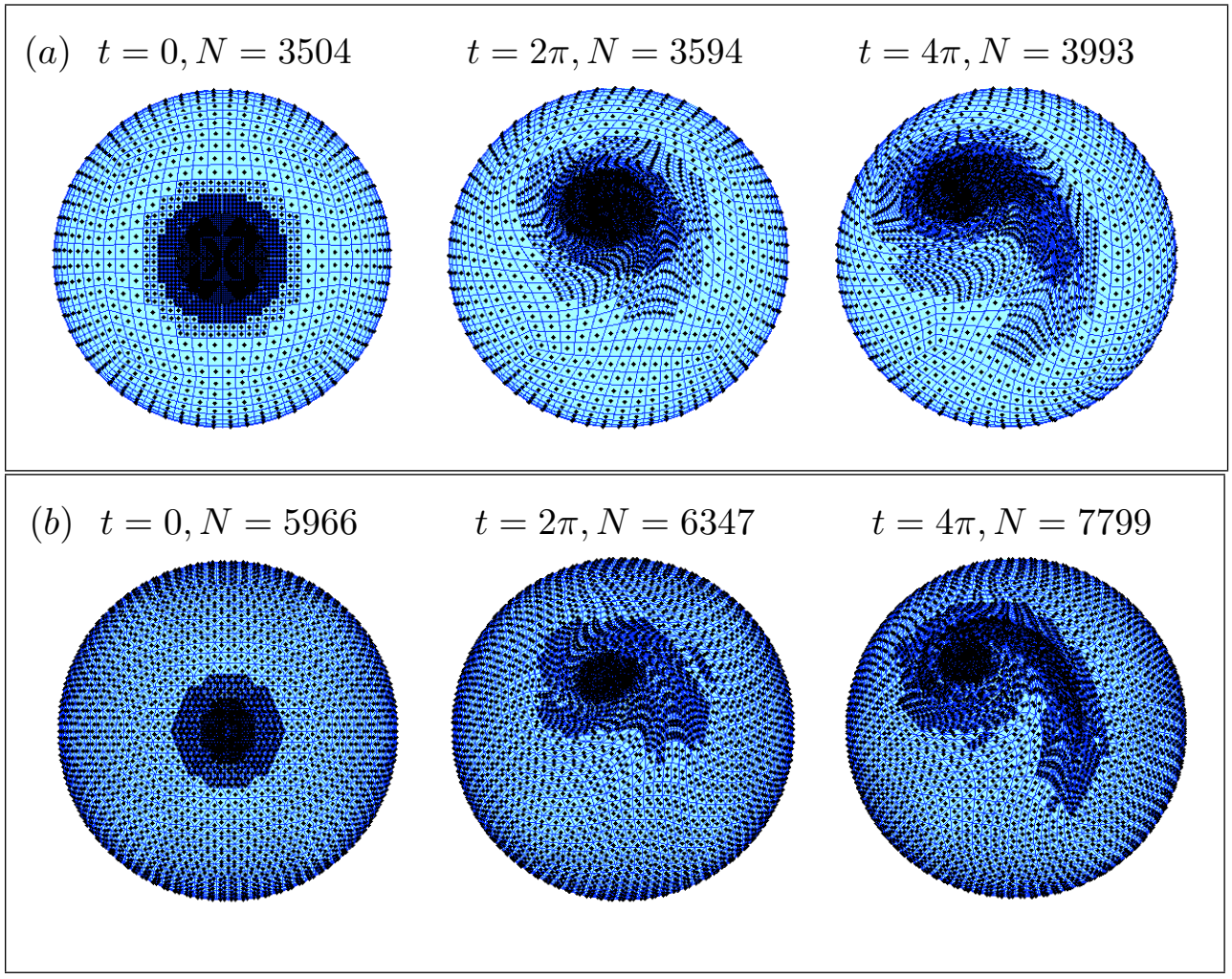


Figure 6.22: Gaussian patch test case 2 at $t = 0, 2\pi, 4\pi$ on a rotating sphere. (a): cubed-sphere mesh; (b) icosahedral triangle mesh. Gaussian patch moves in the northwest direction which agrees with the physical expectation.

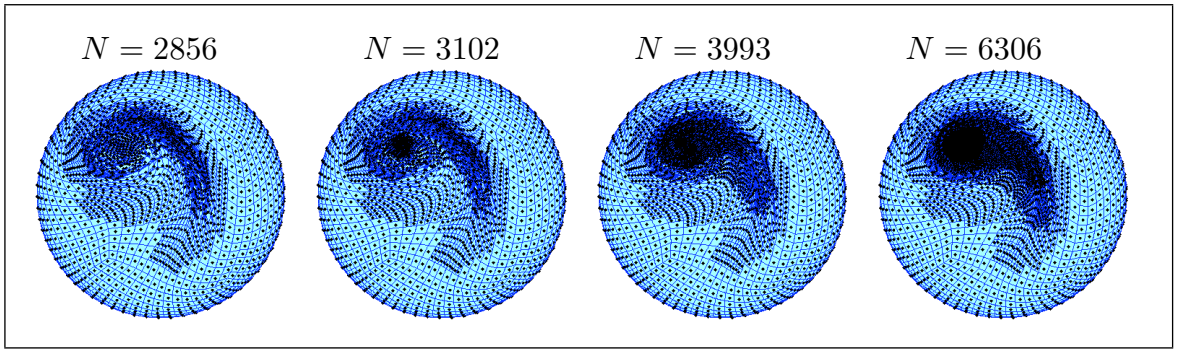


Figure 6.23: Gaussian patch test case 3. From left to right, $\epsilon_\Gamma = 0.001, 0.0005, 0.0002, 0.0001$ at $t = 4\pi$. We can observe qualitative convergence property.

Gaussian Patch	Mesh	Ω	L	ϵ_Γ	ϵ_d
Case 1	CS, IT	0	4	0.0002	0.02
Case 2	CS, IT	1/2	4	0.0002	0.02
Case 3	CS	1/2	4	0.001, 0.0005, 0.0002, 0.0001	0.02
Case 4	CS	0	4	0.0004	0.04
Case 5	CS	1/2	4	0.0004	0.04

Table 6.4: Parameters for Gaussian patch tests. CS = cubed-sphere mesh; IT = icosahedral triangle mesh; Ω = angular velocity; L = levels of the mesh; ϵ_Γ = circulation refinement criterion; ϵ_d = side length refinement criterion; $\Delta t = 4\pi/1000$ and the regulation parameter $\delta = 0.02$.

6.9.1 Non-rotating sphere

For a non-rotating sphere, the interaction of two vortex patches are very similar to the behavior as on the plane [5, 40, 61, 75]. Figure 6.24 shows the interaction of two patches with $\Omega = 0, \delta = 0.02, L = 4, \epsilon_\Gamma = 0.0004, \epsilon_d = 0.04$. The two patches rotate around each other and preserve the symmetric between them. There is a critical distance between the centers of two patches, which controls the patches merge or not [56]. More numerical tests will be conducted in the future for the critical distance of patches on sphere.

6.9.2 Rotating sphere

Figure 6.25 shows the interaction on a rotating sphere $\Omega = 1/2$ within one revolution $t = 4\pi$. The corresponding parameters are $\delta = 0.02, L = 4, \epsilon_\Gamma = 0.0004, \epsilon_d = 0.04$. The center is the same as the nonrotating case. We can see that two patches are not only rotate around each other but also drift towards the northwest direction because of the Coriolis force. Compare to the test on a non-rotating sphere, the two patches are no longer symmetry.

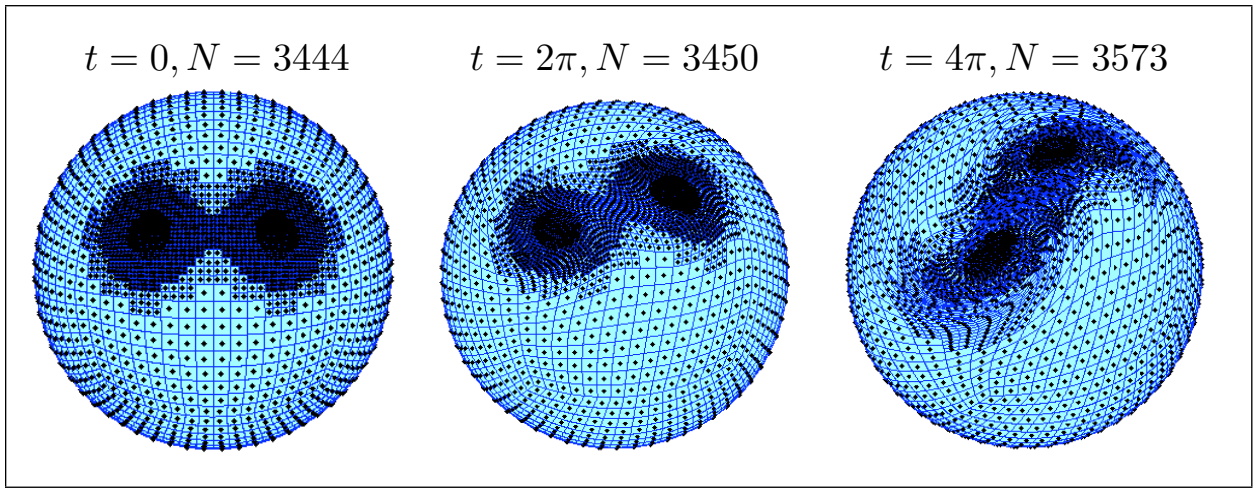


Figure 6.24: Gaussian patch test case 4 at $t = 0, \pi, 3\pi$ on a nonrotating sphere. The initial centers of two patches are $(-0.3670, 0.8548, 0.3670)$ and $(0.3670, 0.8548, 0.3670)$. The two patches rotate around each other and merge.

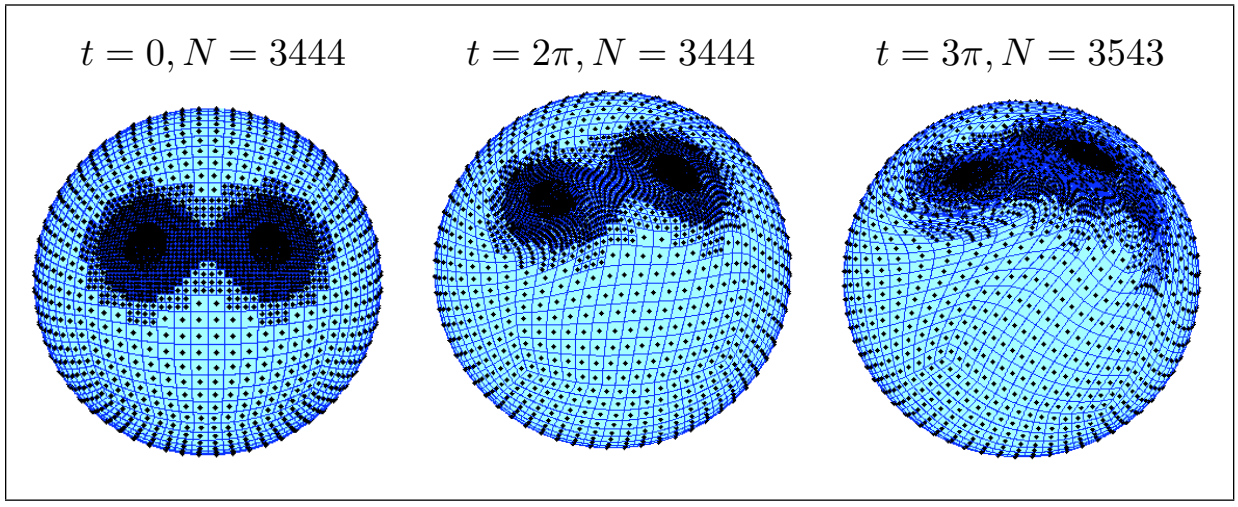


Figure 6.25: Gaussian patch test case 5 at $t = 0, \pi, 3\pi$ on a rotating sphere. The initial centers of two patches are $(-0.3670, 0.8548, 0.3670)$ and $(0.3670, 0.8548, 0.3670)$. The two patches rotate around each other and merge and both are drift towards the northwest direction.

6.10 Remeshing

From the test cases for the Gaussian patches for example as in Figure 6.21, we can see that the panel meshes are no longer regular after a long time integration. Remeshing is one of the ideas to restore a regular mesh and preserve the accuracy of computation. Since the initial mesh is constructed by initial vorticity distribution, we reconstruct the current mesh using the current vorticity distribution. The question is: we know the vorticity that each active point carries, how to calculate it at a regular mesh point? First we choose the number of levels L, then calculate ζ at each center of the new panels by (6.22) for a specified parameter δ. The ζ_j in equation (6.22) is the relative vorticity that carries by active points. Then repeat the meshing procedure we employed at $t = 0$. We conduct remeshing at $t = 2\pi$ and $t = 4\pi$ for Gaussian patch on a non-rotating sphere. Figure 6.26 shows two preliminary results of remeshing with $\delta = 0.025$. In this figure, the panels in (a) and (c) before the remeshing are severely deformed. They are no longer regular quadrilaterals in physical space. From (b) and (d), we can see that the remeshing is doing is good job since it captures the vorticity and also preserves some symmetry of the vorticity field. Developing accurate and efficient remeshing strategy is the subject of our future work.

6.11 Comparison of Vortex Method with GARBF Method

In Chapters five and six, we presented two methods for solving the Barotropic Vorticity Equation. They are both Lagrangian methods since all the grid points are moving with the flow and they both utilize a fourth-order Runge Kutta method in time. Here we present a brief comparison.

- Accuracy/Convergence. The vortex method is first order accurate in terms of

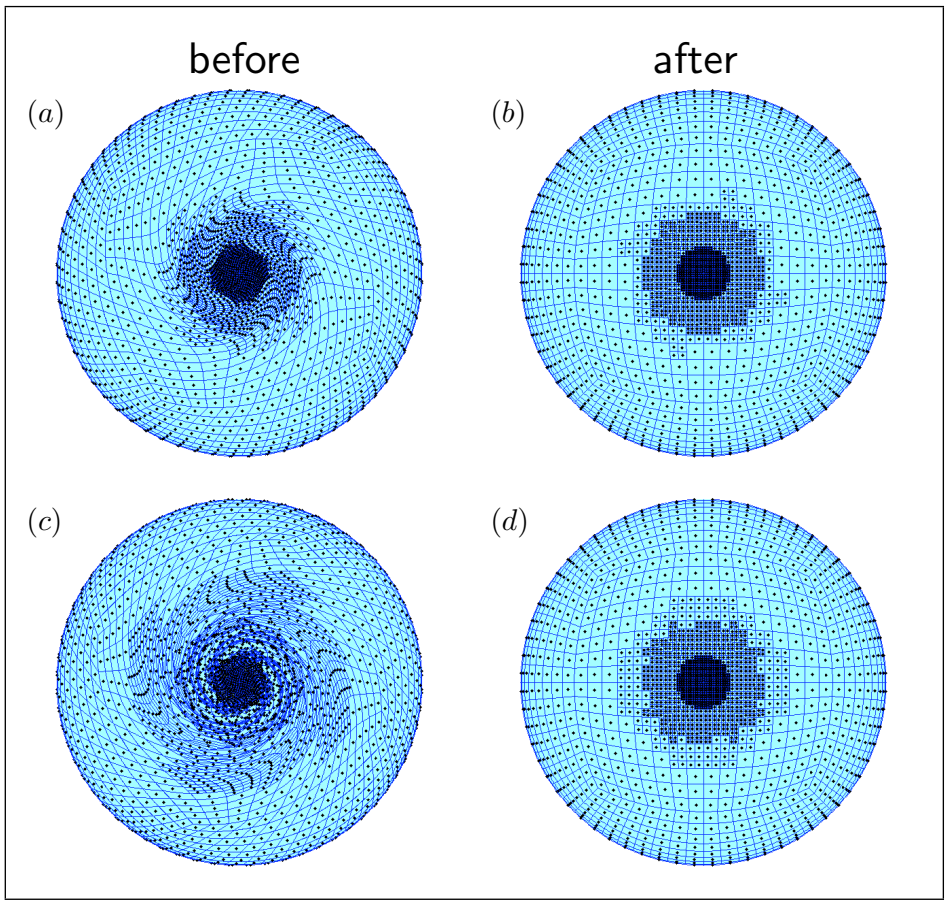


Figure 6.26: Preliminary results of remeshing. Retain the relative vorticity at $L = 4$ uniform mesh with $\delta = 0.02$ using equation (6.22) then refined with $\epsilon_\Gamma = 0.0004$ and $\epsilon_d = 0.02$. (a) and (c) are the Gaussian patch test at $t = 2\pi$ and $t = 4\pi$ with $\Omega = 0, L = 4, \epsilon_\Gamma = 0.0004$ and $\epsilon_d = 0.02, \Delta t = 4\pi/1000, \delta = 0.02$; (b) and (d) are the meshs after remeshing for (a) and (c) respectively.

the number of panels as shown in Figure 6.13. This is because midpoint rule is a second-order scheme in a uniform grid spacing, which means it is a first-order method in terms of the area of the panels. On the other hand, the accuracy of Gaussian RBF method depends on the choice of the RBF parameter ϵ. It can achieve much smaller error if we have good choice of ϵ as shown in Figure 5.3 for only 20 points on the sphere.

- Computational efficiency. Both methods cost $O(N^2)$ floating-point operations per time step for calculating the velocity field, where N is the number of points. There is higher cost for GARBF since we need to invert a matrix to get the RBF coefficients at every time step. If we solve for the RBF coefficients directly using LU factorization, the computational cost is $O(N^3)$, which can be reduced to $O(N^2)$ if iterative methods such as GMRES are implemented. Note that we can apply the treecode discussed in Chapter Three to reduce the computational cost to $O(N \log N)$ for both methods.

- Adaptivity. Both methods can employ the adaptive mesh refinement strategy easily. We can put grid points at the places where high resolution is needed by setting appropriate refinement criteria.

In this chapter, we present a Lagrangian vortex method for the barotropic vorticity equation on a rotating sphere. The method tracks the flow map and absolute vorticity using Lagrangian particles and panels. The velocity is computed from the Biot-Savart integral on the sphere. An adaptive refinement strategy is implemented to resolve small-scale features. Both Rossby-Haurwitz wave and Gaussian vortex patch interaction are tested. The numerical results show the algorithm is first order accuracy in terms of the number of points.

CHAPTER VII

Summary and Future work

7.1 Thesis Summary

In Chapter One we introduce the motivation and the outline of the thesis. Chapter Two gives an introduction to the RBF method, including its background, advantage and related problems. The most significant property of RBF approximation is the trade-off between exponential convergence and ill-conditioning. We also present an RBF cardinal function for Gaussian in one dimensional unbounded evenly spaced grid. At the end of this chapter, we compare the accuracy of RBF and finite differences in terms of Fourier analysis of the eigen-function $\exp(iKX)$. One of the major concerns of the applications of RBF is the computational cost. Chapter three presents a fast treecode algorithm for evaluating RBFs. Instead of using direct summation between each pair of points, it applies a divide-and-conquer strategy to use particle- cluster interactions. Taylor approximation is applied for the far-field expansion. The treecode reduces the computational cost from $O(N^2)$ to $O(N \log N)$, where N is the number of particles in the system. The following three chapters are applications related to the fluid flow on the surface of the sphere. Chapter four introduces the Barotropic Vorticity Equation(BVE), which is the simplest atmospheric model. Chapters five and six solve BVE by Gaussian RBF and vortex methods in

Lagrangian form. The Rossby-Haurwitz wave and Gaussian patch are tested in both chapters.

7.2 Improving the Vortex/RBF Method for solving BVE

We will keep improving the vortex/RBF method for solving BVE in the following aspects.

- ϵ for nonuniform grid on sphere. ϵ is important for solving PDEs by RBF method accurately. For nonuniform grid, ϵ needs to vary according to the grid size. We did experiments for two ϵ choices reported in the literature [31], [38]. They both don't work well since ill-conditioning is still an issue. Hence, looking for a better choice of ϵ is a task for future work.

- Remeshing. Remeshing is an important step to preserve the accuracy of the computation. There are two major points that we need to consider. The first is the experiments of choosing a reasonable value of δ. The second one is the efficiency of the remeshing step.

- Check energy conservation. Energy conservation a very important property for a numerical algorithm. The vortex method conserves circulation, which is a very good property.

- High order mesh refinement strategy. Our current strategy for local mesh refinement is using linear interpolation to find the position of the new points. We may apply high order interpolation to increase the accuracy. Figure 7.1 shows a way to have quadratic interpolation. For example, each cubic panel has eight passive points and one active point in Figure 7.1(a). There are three points on each side, then quadratic interpolation can be applied to calculate the position

of new points.

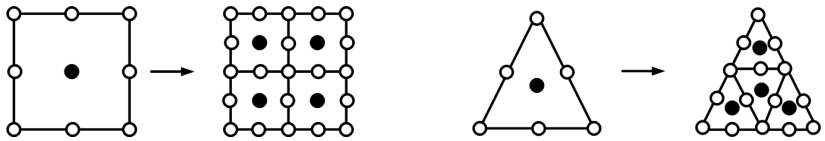


Figure 7.1: Stencils for high order refinement. (a): one active point and eight passive points in quadrilateral panel; (b): one active point and 6 passive points in triangular panel.

7.3 Shallow Water Equation (SWE)

The SWEs are an ideal testbed for numerical method in geophysical fluid flow. The equations are

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \boldsymbol{f} \times \boldsymbol{u} = -g \nabla h \tag{7.1}$$

$$\frac{\partial h}{\partial t} + \boldsymbol{u} \cdot \nabla h + h \nabla \cdot u = 0, \tag{7.2}$$

where $\boldsymbol{u}$ is the horizontal velocity, h is the layer height, $\boldsymbol{f} = f\boldsymbol{k} and f = 2\Omega \cos\theta$ is the Coriolis parameter [74, 72]. The velocity can be recovered from vorticity and divergence by solving

$$\Delta \psi = \zeta, \tag{7.3}$$

$$\Delta \chi = \delta, \tag{7.4}$$

$$\boldsymbol{u} = \boldsymbol{k} \times \nabla \psi + \nabla \chi \tag{7.5}$$

[72]. We can do this by a strategy similar to that for the barotropic vorticity equation, which uses Green's function in vortex method and the solution of the Poisson equation with Gaussian forcing in Gaussian RBF.

7.4 Application and Extension of Treecode

In our current application projects, both Gaussian RBF and Vortex methods are $O(N^2)$ system, where N is the number of points. A treecode can reduce the

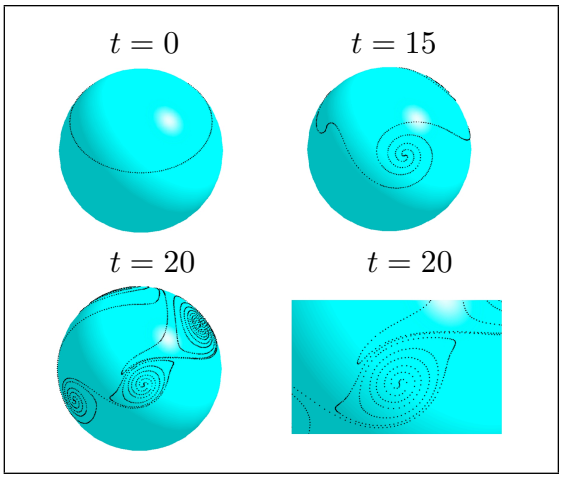


Figure 7.2: Vortex sheet evolution $t = 0, 15, 20$ and the last one is a close-up at $t = 20$.

computational cost to $O(N \log N)$. Our current treecode works for particles in three dimension. It works for particles both in a cube and on a surface of the sphere. Since we are interested in geophysical problems, which are on the spherical geometry, we will tailor the treecode to fit our simulation. For example, instead of using cubic box as our clusters, we can use spherical triangles as an alternative. We believe it will be more efficient since we will be using the quad-tree structure on a two dimension manifold instead of an oct-tree structure in space.

7.5 Jet Simulation

A vortex sheet is a material surface in the fluid across which the tangential component of fluid velocity has a jump discontinuity [29]. They are frequently used as an asymptotic model for parallel shear flow. It can also be applied to jet simulation, which is a very common phenomenon in atmosphere. Figure 7.2 is a vortex sheet roll-up on the sphere at $t = 0, 15, 20$. This shows we can do jet simulation using two vortex sheets with opposite sign. The other way to simulate jet is using barotropic vorticity equation, where the initial vorticity is two rings which are parallel to the latitude circle.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] *CRC Standard Mathematical Tables and Formulae*. CRC Press, 1996.

[2] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions*. Dover, New York, 1965.

[3] J. Adem. A series solution for the barotropic vorticity equation and its application in the study of atmospheric vortices. *Tellus*, 3:364–372, 1956.

[4] A. Arakawa and V.R. Lamb. *Computational Design of the Basic Dynamical Processes of the UCLA General Circulation Model*. San Francisco: Academic Press, 1977.

[5] L. A. Barba and A. Leonard. Emergence and evolution of tripole vortices from net-circulation initial conditions. *Phys. Flui*, 19.

[6] J.E. Barnes. A modified tree code: Don't laugh; it runs. *J. Comput. Phys.*, 87:161–170, 1990.

[7] J.E. Barnes and P. Hut. A hierarchical $O(N \log N)$ force-calculation algorithm. *Nature*, 324(4):446–449, 1986.

[8] J.T. Beale. *On the accuracy of vortex methods at large times*. Springer-Verlag, New York, 1988.

[9] J.T. Beale and A. Majda. Vortex methods I: Convergence in three dimensions. *Math. Comput.*, 29(159):1–27, 1982.

[10] R.K. Beatson and L. Greengard. *A Short Course on Fast Multipole Method*. Oxford University Press, 1997.

[11] R.K. Beatson and G.N. Newsam. Fast evaluation of radial basis functions: I. *Comput. Math. Applic.*, 24(12):7–19, 1992.

[12] R. Bellman. *A Brief Introduction to Theta Functions*. Holt, Rinehart and Winston, New York, 1961.

[13] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84, 1989.

[14] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

[15] J.P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover, 1990.

[16] J.P. Boyd and L. Wang. An analytic approximation to the cardinal functions of Gaussian radial basis functions on an infinite lattice. *Appl. Math. Compu.*, 215:2215–2223, 2009.

[17] J.P. Boyd and L. Wang. Truncated gaussian RBF differences are always inferior to finite differences of the same stencil width. *Commu. Comput. Phys.*, 5:42–60, 2009.

[18] J.P. Boyd and C. Zhou. Three ways to solve the possion equation on a sphere with Gaussian forcing. *J. of Comput. Phys.*, 228:4702–4713, 2009.

[19] M.D. Buhmann. *Radial Basis Functions: Theory and Implementations, Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, 2003.

[20] J.G. Charney, R. Fjortoft, and J.G. von Neumann. Numerical integration of the barotropic vorticity equation. *Tellus*, 2:237–254, 1950.

[21] H. Chen and J.S. Marshall. A Lagrangian vorticity method for two-phase particulate flows with two-way phase coupling. *J. Comput. Phys.*, 148.

[22] J.B. Cherrie, R.K. Beatson, and G.N. Newsam. Fast evaluation of radial basis function methods for generalized multiquadrics in R^n. *SIAM J. Sci. Comput*, 23(5):1549–1571, 2002.

[23] A.J. Chorin and P.S. Bernard. Discretization of a vortex sheet. *J. Comput. Phys.*, 13:423–429, 1973.

[24] D.G. Crowdy. Gaussian vortices on a sphere. *private communication*.

[25] D.G. Crowdy. Stuart vortices on a sphere. *J.Fluid Mech.*, 498:381–402, 2004.

[26] M. DeMaria. Tropical cyclone motion in a nondivergent barotropic model. *Mon. Wea. Rev.*, 113:1199–1210, 1985.

[27] J. Duchon. Splines minimizing rotation invariant seminorms in sobolev spaces. *constructive theory of functions of several variables*, 1:85–100, 1976.

[28] K.G. Evans, D.W. Rouson, A.G. Salinger, M.A. Taylor, W. Weijer, and J.B. White. *A Scalable and Adaptable Solution Framework within Components of the Community Climate System Model*. Springer-Verlag Berlin, Heidelberg, 2009.

[29] H. Feng. *Vortex Sheet Simulations of 3D Flows Using an Adaptive Triangular Panel/Particle Method*. PhD thesis, University of Michigan, Ann Arbor, 2007.

[30] H. Feng, L. Kaganovskiy, and R. Krasny. Azimuthal instability of a vortex ring computed by a vortex sheet panel method. *Fluid Dyn. Res.*, 41:051405, 2009.

[31] N. Flyer and E. Lehto. Rotational transport on a sphere: Local node refinement with radial basis functions. *J. Comput. Phys.*, 229:1954–1969, 2009.

[32] N. Flyer and G. Wright. Transport schemes on a sphere using radial basis functions. *J. Comput. Phys.*, 229:1954–1969, 2007.

[33] N. Flyer and G. Wright. A radial basis function method for the shallow water equations on a sphere. *Proc. R. Soc. A*, 465:1949–1976, 2009.

[34] B. Fornberg and N. Flyer. Accuracy of radial basis function interpolation and derivative approximations on 1-d infinite grids. *Adv. Comput. Math.*, 23:5–20, 2005.

[35] B. Fornberg, N. Flyer, S. Hovde, and C. Piret. Localization properties of rbf expansion coefficients for cardinal interpolation. i. equispaced nodes. *Adv. Comput. Math.*, 47:5–20, 2006.

[36] B. Fornberg and C. Piret. A stable algorithm for flat radial basis functions on a sphere. *SIAM J. Sci. Comp.*, 30:60–80, 2007.

[37] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Comp. Math. Appl.*, 48:853–867, 2004.

[38] B. Fornberg and J. Zuev. The runge phenomenon and spatially variable shape parameters in RBF interpolation. *Adv. Comput. Math.*, 47:5–20, 2006.

[39] R. Franke. Scattered data interpolation: Tests of some methods. *Math. Comp.*, 38:181–200, 1982.

[40] O.U. Velasco Fuentes. Vortex filamentation: its onset and its role on axisymmetrization and merger. *Dyna. Atmo. Ocea.*, 40:23–42, 2005.

[41] I.S. Gradshteyn and I.M. Ryzhik. *Table of Integrals, Series, and Products, fourth ed.* Academic Press, 1965.

[42] L. Greengard. The numerical solution of the N-body problem. *Comput. Phys.*, pages 142–152, 1990.

[43] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325–348, 1987.

[44] L. Greengard and J. Strain. The fast gauss transform. *SIAM J. Sci. Stat. Comput.*, 12(1):79–94, 1991.

[45] R.L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophy. Res.*, 76:1905–1915, 1971.

[46] J.R. Holton. *An Introduction to Dynamic Meteorology.* Elsevier, 2004.

[47] C. Jablonowski. *Adaptive Grids in Weather and Climate Modeling.* PhD thesis, University of Michigan, 2004.

[48] C. Jablonowski, M. Herzog, J. E. Penner, R. C. Oehmke, Q. F. Stout, B. van Leer, and K. G. Powell. Block-structured adaptive grids on the sphere: Advection experiments. *Mon. Wea. Rev.*, 134:3691–3713, 2006.

[49] E.J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid-dynamics. i. surface approximations and partial derivative estimates. *Comput. Math. Appl.*, 19:127–145, 1990.

[50] E.J. Kansa. Radial basis function for the sphere. *Comput. Math. Appl.*, 19:147–161, 1990.

[51] Y. Kimura and H. Okamoto. Vortex motion on a sphere. *J. Phys. Soc. Japan*, 56(12):4203–4206, 1987.

[52] R. Krasny. Desingularization of periodic vortex sheet roll-up. *J. Comput. Phys.*, 65(2):292–313, 1986.

[53] R. Krasny and L. Wang. A treecode for fast summation of multiquadric RBFs. *SIAM J. Sci. Compu., submitted.*

[54] E. Larsson and B. Fornberg. A numerical study of some radial basis function based solution methods for elliptic PDEs. *Comput. Math. with Appl.*, 46(6):891–902, 2003.

[55] A. Leonard. Vortex methods for flow simulation. *J. Comput. Phys.*, 37:289–335, 1980.

[56] M.M. Levy, R.D. Nair, and H.M. Tufo. A high-order element-based Galerkin method for the barotropic vorticity equation. *Int. J. Numer. Meth. Fluids*, 59:1369–1387, 2009.

[57] P. Li, H. Johnston, and R Krasny. A Cartesian treecode for screened coulomb interactions. *J. Comput. Phys.*, 228:3858–3868, 2009.

[58] K. Lindsay and R. Krasny. A particle method and adaptive treecode for vortex sheet mothion in three-dimensional flow. *J. Comput. Phys.*, 172:879–907, 2001.

[59] O.E. Livne and G.B. Wright. Fast multilevel evaluation of smooth radial basis function expansions. *ETNA*, 23:263–287, 2006.

[60] J.B. Lundberg and B.E. Schutz. Recursion formulas of Legendre functions for use with non-singular geopotential models. *J. Guid., Contr., and Dynam. (ISSN 0731-5090)*, 11:31–38, 1988.

[61] P. Meunier, U. Ehrenstein, T. Leweke, and M. Rossi. A merging criterion for two-dimensional co-rotating vortices. *Phys. Fluid*, 14(8).

[62] C.A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite finctions. *Constr. Approx.*, 2:11–22, 1986.

[63] P. K. Newton and H. Shokraneh. The n-vortex problem on a rotating sphere: I. multi-frequency configurations. *Proc. R. Soc.*, 462:149–169, 2006.

[64] P.K. Newton and T. Sakajo. The n-vortex problem on a rotating sphere: III. ring configuration coupled to a background filed. *Proc. R. Soc.*, 463:961–977, 2007.

[65] W.M. Putman and S.J. Lin. Finite-volume transport on various cubed-sphere grids. *J. Comput. Phys.*, 227:55–78, 2007.

[66] D.A. Randall. *General Circulation Model Development: Past, Present and Future.* Academic Press, 2000.

[67] T.D. Ringler and D.A. Randall. A potential enstrophy and energy conserving numerical scheme for solution of the shallow-water equations on a geodesic grid. *Mon. Wea. Rev.*, 130(5):1397–1410, 2002.

[68] G. Russo and J.A. Strain. Fast triangulated vortex methods for the 2D Euler equation. *J. Comput. Phys.*, 111:291–323, 1994.

[69] R. Sadourny, A. Arakawa, and Y. Mintz. Integration of the nondivergent barotropic vorticity equation with an icosahedral-hexagonal grid for the sphere. *Mon. Wea. Rev.*, pages 351–355, 1968.

[70] T. Sakajo. Motion of vortex sheet on a sphere with pole vortices. *Phys. Fluids*, 16(3):717–727, 2004.

[71] S.E. Shin, J.Y. Han, and J.J. Baik. On the critical separation distance of binary vortices in a nondivergent barotropic atmoshphere. *J. Meteo. Soc. Japan*, 84:853–869, 2006.

[72] R.K. Smith and D.G. Dritschel. Revisiting the Rossby-Haurwitz wave test case with contour advection. *J. Comput. Phys.*, 217:473–484, 2006.

[73] M.J. Stock. Summary of vortex methods literature. *http://markjstock.org/research.*

[74] G. K. Vallis. *Atmospheric and Oceanic Fluid Dynamics.* Cambridge University Press, 2006.

[75] D.W. Waugh. The efficiency of symmetric vortex merger. *Phys. Fluids A*, 4(8):1745–1758, 1992.

[76] D.L. Williamson. Integration of the barotropic vorticity equation on a spherical geodesic grid. *Tellus*, 20(4):642–653, 1968.

[77] H.B. Yao, N.J. Zabusky, and D.G. Dritschel. High gradient phenomena in two-dimensional vortex interactions. *Phys. Fluids*, 7(3), 1993.

[78] N.J. Zabusky and J.C. McWilliams. A modulated point-vortex model for geostrophic, β-plane dynamics. *Phys. Fluid*, 25(12):2175–2182, 1982.