An extension of Draghicescu's fast tree-code algorithm to the vortex method on a sphere *

Takashi Sakajo
Department of mathematics, Hokkaido University
&
PRESTO, Japan Science and Technology Agency

October 19, 2008

Abstract

A fast and accurate algorithm to compute interactions between N point vortices and between N vortex blobs on a sphere is proposed. It is an extension of the fast tree-code algorithm developed by Draghicescu for the vortex method in the plane. When we choose numerical parameters in the fast algorithm suitably, the computational cost of $O(N^2)$ is reduced to $O(N(\log N)^4)$ and the approximation error decreases like $O(1/N)$ as $N \to \infty$, which is demonstrated in the present article. We also apply the fast method to long-time evolution of two vortex sheets on the sphere to see the efficiency. A key point is to describe the equation of motion for the N points in the three-dimensional Cartesian coordinates.

Keywords: fast tree-code algorithms, flows on sphere, vortex method, vortex sheet

AMS subject classification(MSC): 65Y20, 65C20, 76B47, 76M23

1 Introduction

We propose a fast numerical algorithm to compute the evolution of the incompressible and inviscid fluid confined on the surface of a sphere S with radius R. Since the vorticity, which is denoted by a scalar function $\omega_0(\theta, \phi)$ in the spherical coordinates (θ, ϕ), is conserved along the path of fluid particles, it is possible to approximate the evolution of the vorticity region with that of many point vortices. Let us consider a bounded vorticity region, whose support is denoted by $\mathcal{A} \subset S$. Then, discretizing the region with N cells, we set a point vortex at a certain position (θ_m, ϕ_m) in each cell for $m = 1, 2, \ldots, N$. The strength of the point vortex Γ_m is represented by

$$\Gamma_m = \frac{m(\mathcal{A})}{N} \omega_0(\theta_m, \phi_m),$$

in which $m(\mathcal{A}) \leq 4\pi R^2$ is a measure of the vorticity region $\mathcal{A}$ on the sphere. When the vorticity field is approximated by

$$\omega_0(\theta, \phi) \approx \frac{1}{\sin\theta} \sum_{m=1}^{N} \Gamma_m \delta(\theta - \theta_m, \phi - \phi_m), \tag{1}$$

*Mailing address: Kita 10 Nishi 8, Sapporo Hokkaido, 060-0810 JAPAN, E-mail: sakajo@math.sci.hokudai.ac.jp, TEL: +81-11-706-4660, FAX: +81-11-727-3705

the Euler equations are reduced to the equations for the N point vortices [11], which are given by

$$\dot{\theta}_m = -\frac{1}{4\pi R^2}\sum_{j\neq m}^{N}\frac{\Gamma_j \sin\theta_j \sin(\phi_m-\phi_j)}{1+\sigma^2-\cos\theta_m\cos\theta_j-\sin\theta_m\sin\theta_j\cos(\phi_m-\phi_j)}, \tag{2}$$

$$\dot{\phi}_m = -\frac{1}{4\pi R^2 \sin\theta_m}\sum_{j\neq m}^{N}\frac{\Gamma_j[\cos\theta_m\sin\theta_j\cos(\phi_m-\phi_j)-\sin\theta_m\cos\theta_j]}{1+\sigma^2-\cos\theta_m\cos\theta_j-\sin\theta_m\sin\theta_j\cos(\phi_m-\phi_j)}, \tag{3}$$

for $m = 1, 2, \ldots, N$. The approximation method for the Euler equations is known as *the vortex method*. Not only is the vortex method of practical use in numerical investigations for the Euler flows, but it is also mathematically shown that the solution of the vortex method uniformly converges to that of the Euler equations in two-dimensional Euclidean space when $N \to \infty$ as long as the solution of the Euler equations is smooth [10]. Many references regarding the vortex method are found in [2]. Here, we introduce the parameter σ to regularize the singular behavior of the velocity fields (2) and (3) when one point vortex approaches another point closely. The regularization method is known as *the vortex blob method* that has been used in the numerical computations of vortex sheets [15, 16, 20, 21, 22]. The discretizing points are sometimes referred to as *the point vortices* for $\sigma = 0$ and *the vortex blobs* for $\sigma \neq 0$, respectively.

Another computational method for the incompressible and inviscid flows on the sphere is *the contour dynamics method* with a surgery technique developed by Dritschel [6]. In this method, approximating the vorticity region $\mathcal{A}$ by some subregions with piecewise constant vorticity, one has only to track the evolution of their boundaries, since the velocity field induced by the constant vorticity region is represented by the path integral along the boundary owing to Green's formula. The contour dynamics method has been used to compute the evolution of the constant vorticity strips [7] and the polar constant vorticity cap [19]. It also gives us an effective way to investigate flows on the rotating sphere, since the effect of rotation is approximated by the constant vorticity strips corresponding to the background solid-body rotation [3, 18].

Numerical computation of (2) and (3) gives rise to a serious difficulty in the computational cost. Namely, we need $O(N)$ amount of computations to evaluate the velocity fields (2) and (3) for one vortex blob. Accordingly, the total of $O(N^2)$-operations is required to finish the evaluation for all the points. The same difficulty arises in numerical computation of the interaction between N gravitational bodies and between N charges, for which fast algorithms such as the fast tree-code algorithms [1, 4] and the fast multipole methods [8, 9] have been developed. These fast methods basically work as follows. Suppose that the N points are assigned in the computational domain. Then in order to evaluate a velocity field or a force field induced by the points at a given position, the N points are divided into two groups, called *the far-field* and *the near-field*. The contribution from the far-field is evaluated by using a certain approximation such as the Taylor expansion and the multipole expansion. As for the points in the near-field, they are computed directly. Owing to the efficient far-field approximation, the fast methods reduce the computational cost to $O(N(\log N)^{d+1})$ for the fast tree-code methods and $O(N)$ for the fast multipole methods, where d is the dimension of the computational domain. As far as the order of computational cost is concerned, the fast multipole methods are better than the fast tree-code algorithms. On the other hand, however, while the fast multipole methods work only for harmonic kernels, the fast tree-code methods can be implemented for any kernel as long as its Taylor expansion is efficiently computed. Owing to its versatility, the fast tree-code algorithms have been used in the numerical computation of the Euler flows. Draghicescu et al. proposed a fast

tree-code algorithm for the point-vortex and the vortex-blob approximations of the Euler equations in two-dimensional and three-dimensional Euclidean spaces [4, 5]. The fast tree-code algorithm was extended successfully to numerical computations of three-dimensional vortex sheets [16, 21] and a vortex sheet with the periodic boundary condition [20], in which recurrence formulas were used to compute the Taylor coefficients of their kernels effectively.

In the present paper, we extend Draghicescu's tree-code algorithm so that we can compute the velocity field induced by the point vortices and the vortex blobs on the sphere. The extension is possible in theory, but it is quite hard to implement the numerical computation in practice, since the expression of the Taylor coefficients for (2) and (3) is so complicated that we can't compute them efficiently. This drawback cancels out the efficiency of the fast algorithm. In order to resolve the problem, we rewrite eqs. (2) and (3) with respect to the three-dimensional Cartesian coordinates instead of the spherical coordinates. Let the position of the discretizing points $\boldsymbol{x}_m$ for $m = 1, \ldots, N$ in $\mathbb{R}^3$ be represented by

$$\boldsymbol{x}_m = (x_m(t), y_m(t), z_m(t)) = (R\sin\theta_m\cos\phi_m, R\sin\theta_m\sin\phi_m, R\cos\theta_m).$$

Then the stream function $\psi(\boldsymbol{x})$ is recovered from the vorticity field with the inversion formula

$$\psi(\boldsymbol{x}) = \iint_S G(\boldsymbol{x}, \boldsymbol{x}')\omega(\boldsymbol{x})dA,$$

where the Green function on the sphere is given by $G(\boldsymbol{x}, \boldsymbol{x}') = -(1/4\pi)\log|\boldsymbol{x}-\boldsymbol{x}'|^2$. (See [11]) With the singular vorticity field approximation (1), the stream function gives rise to the equation for the N point vortices [17, 18]:

$$\dot{\boldsymbol{x}}_m = -\frac{1}{4\pi R}\sum_{j\neq m}^{N}\Gamma_j\frac{\boldsymbol{x}_m\times\boldsymbol{x}_j}{R^2-\boldsymbol{x}_m\cdot\boldsymbol{x}_j}, \quad m = 1, 2, \ldots, N. \tag{4}$$

The regularized equation for the vortex blobs is given by

$$\dot{\boldsymbol{x}}_m = -\frac{1}{4\pi R}\sum_{j\neq m}^{N}\Gamma_j\frac{\boldsymbol{x}_m\times\boldsymbol{x}_j}{R^2+\sigma^2-\boldsymbol{x}_m\cdot\boldsymbol{x}_j}, \quad m = 1, 2, \ldots, N. \tag{5}$$

The fast tree-code algorithm presented in this paper can be applied to both eqs. (4) and (5). The change of variables enables us to compute the Taylor expansion of the kernels in eqs. (4) and (5) easily with a simple recursive formula. A similar idea was used in the numerical computation of a two-dimensional vortex sheet with periodic boundary condition [20]. On the other hand, due to the Cartesian formulation, the discretizing points might detach from the sphere if the accuracy of the computation is too poor. In the numerical examples in this paper, we verify whether these points stay on the surface of the sphere by checking their distances from the origin each time step.

The paper consists of four sections. In the next section, we see how to extend the fast algorithm to the evaluation of the velocity field (4) and give its error estimate. In the third section, some numerical examples are shown to confirm the efficiency of the algorithm. The last section is devoted to summary and discussion on future applications. A formal description of the fast algorithm and a linear stability analysis for the two vortex sheets are given in the appendices.

2 Fast tree-code algorithm

2.1 Mesh generation, far-field and near-field

Since we describe the fast algorithm for the sphere in $\mathbb{R}^3$, we set the computational domain $\mathcal{B}$ as a three-dimensional box, a little bit larger than the sphere of radius R, i.e.,

$$\mathcal{B} = [-(1+\delta)R, (1+\delta)R]^3.$$

The positive parameter δ is taken sufficiently small but not zero so that the box $\mathcal{B}$ contains the sphere completely. Then we construct a tree structure of small boxes in $\mathcal{B}$ by dividing the box region into two smaller box regions recursively. Since the points are confined on the surface of the sphere, we can eliminate the boxes that have no intersection with the sphere from the tree structure, which helps us save the working memory on the computer. In what follows, without loss of generality, we set the radius of the sphere $R = 0.5/(1+\delta)$ so that the box $\mathcal{B}$ becomes a unit box.

First, we generate the tree structure with the following recursive algorithm for a given box $\tau = [x_1, x_2) \times [y_1, y_2) \times [z_1, z_2)$. At the same time, the algorithm initializes parameters $\rho(\tau)$, $\boldsymbol{y}_\tau$, $L(\tau)$ and $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$, $C_\tau^{\boldsymbol{k}}$ for the box τ and the multi-index $\boldsymbol{k} = (k_1, k_2, k_3)$. The center point $\boldsymbol{y}_\tau$ of the box τ and its radius $\rho(\tau)$ are defined by

$$\boldsymbol{y}_\tau = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}, \frac{z_1+z_2}{2} \right), \qquad \rho(\tau) = \sup_{\boldsymbol{y}\in\tau} |\boldsymbol{y} - \boldsymbol{y}_\tau| .$$

The parameters $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$ and $C_\tau^{\boldsymbol{k}}$ are required in the far-field computation and their definitions are provided in the next subsection, while $L(\tau)$ denotes a list of the points contained in the finest box τ, which is used to compute the near-field evaluation. For a given integer l, which determines the finest level of the tree structure, and an integer λ, which is the approximation order of the Taylor expansion, we give the following mesh-generation algorithm.

Algorithm 1. *(Generate the tree structure of boxes)*

Input: box τ, integers j, λ and l;
GenerateMesh(τ, j, l)
    if $j = 3l$ return;
    $\boldsymbol{y}_\tau = (c_{\tau 1}, c_{\tau 2}, c_{\tau 3}) = \left(\frac{1}{2}(x_1+x_2), \frac{1}{2}(y_1+y_2), \frac{1}{2}(z_1+z_2)\right)$;
    Compute the radius of τ, $\rho(\tau)$;
    For all $\boldsymbol{k} = (k_1, k_2, k_3)$, $|\boldsymbol{k}| = k_1 + k_2 + k_3 \le \lambda - 1$, initialize $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$, $C_\tau^{\boldsymbol{k}}$;
    Initialize the list of the near-fields, $L(\tau)$;
    if $j \mod 3 = 1$
        $\tau_1 = [x_1, c_{\tau 1}) \times [y_1, y_2) \times [z_1, z_2)$; $\tau_2 = [c_{\tau 1}, x_2) \times [y_1, y_2) \times [z_1, z_2)$;
    else if $j \mod 3 = 2$
        $\tau_1 = [x_1, x_2) \times [y_1, c_{\tau 2}) \times [z_1, z_2)$; $\tau_2 = [x_1, x_2) \times [c_{\tau 2}, y_2) \times [z_1, z_2)$;
    else if $j \mod 3 = 0$
        $\tau_1 = [x_1, x_2) \times [y_1, y_2) \times [z_1, c_{\tau 3})$; $\tau_2 = [x_1, x_2) \times [y_1, y_2) \times [c_{\tau 3}, z_2)$;
    end
    For each τ_i, if $\tau_i \cap S \neq \emptyset$
        set τ_i as a child box of τ;
        recursively call GenerateMesh(τ_i, $j+1$, l);
        return;
    else
        set $\emptyset$ as a child of τ;
        return;

```
    end
    return;
end
```

End of Algorithm

Applying the algorithm to the computational domain $\tau = \mathcal{B}$, we obtain the tree structure consisting of a hierarchy of boxes of various sizes, which is denoted by Σ in what follows. Note that the size of a box on the finest level $3l$ becomes $h = 2^{-l}$.

Next, for a given point $\boldsymbol{x} \in S$, we define the set of boxes, called the *far-field* of $\boldsymbol{x}$, $\mathcal{F}(\boldsymbol{x})$, and the *near-field* of $\boldsymbol{x}$, $N(\boldsymbol{x})$ as in [4].

Definition 1. *(Far-field and near-field) For $\boldsymbol{x} \in S$, $\mathcal{F}(\boldsymbol{x})$ contains all boxes τ with center $\boldsymbol{y}_\tau$ such that the following condition is satisfied.*

$$\rho(\tau) \le h^\nu \left| R^2 - \boldsymbol{x} \cdot \boldsymbol{y}_\tau \right|, \tag{6}$$

and τ is maximal. Let $F(\boldsymbol{x}) = \cup \mathcal{F}(\boldsymbol{x})$. The parameter $\nu > 0$ is determined to control the efficiency of the fast algorithm and the accuracy of the numerical computation. The near-field of $\boldsymbol{x}$ is defined by $N(\boldsymbol{x}) = \Sigma \setminus F(\boldsymbol{x})$.

The definitions of the far-field and near-field are the same as those given in [4] except for the condition (6) defining the far-field. Let us remember that the original far-field condition in [4] is given by

$$\rho(\tau) \le h^\nu \left| \boldsymbol{x} - \boldsymbol{y}_\tau \right|. \tag{7}$$

The condition (7) would work in practice, but we found it convenient to use (6) instead to facilitate the error estimate below. As a matter of fact, the choice of the condition is not a serious problem for $R < 0.5$, since $\left| R^2 - \boldsymbol{x} \cdot \boldsymbol{y}_\tau \right| \le |\boldsymbol{x} - \boldsymbol{y}_\tau|$ holds for arbitrary $\boldsymbol{y}_\tau \in \mathbb{R}^3$ and $|\boldsymbol{x}| = 0.5$. Hence, the far-field of $\boldsymbol{x}$ in the present algorithm becomes automatically that of the three-dimensional fast tree-code algorithm.

2.2 Far-field approximation

The fast algorithm evaluates the following velocity field $\boldsymbol{u}_N(\boldsymbol{x}, t)$ induced by the given vortex blobs at $\boldsymbol{y}_j$ $(j = 1, \cdots, N)$ in eq. (5):

$$\boldsymbol{u}_N(\boldsymbol{x}, t) = -\frac{1}{4\pi R} \sum_{j=1}^{N} \Gamma_j \frac{\boldsymbol{x} \times \boldsymbol{y}_j}{R^2 + \sigma^2 - \boldsymbol{x} \cdot \boldsymbol{y}_j} \equiv -\frac{1}{4\pi R} \sum_{j=1}^{N} \Gamma_j \boldsymbol{\gamma}(\boldsymbol{x}, \boldsymbol{y}_j) \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_j), \tag{8}$$

where $\boldsymbol{\gamma}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x} \times \boldsymbol{y}$ and $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) = (R^2 + \sigma^2 - \boldsymbol{x} \cdot \boldsymbol{y})^{-1}$. The contribution from the near-field is computed directly, and that from the far-field is approximated by the Taylor expansion of $\mathcal{D}$ up to the $(\lambda - 1)$th order. That is to say, the velocity field $\boldsymbol{u}_N$ is approximated by

$$\boldsymbol{u}_N^\lambda(\boldsymbol{x}, t) = -\frac{1}{4\pi R} \sum_{\boldsymbol{y}_j \in N(\boldsymbol{x})} \Gamma_j \boldsymbol{\gamma}(\boldsymbol{x}, \boldsymbol{y}_j) \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_j) - \frac{1}{4\pi R} \sum_{\tau \in F(\boldsymbol{x})} \boldsymbol{u}_N^{\lambda,\tau}(\boldsymbol{x}, t), \tag{9}$$

in which

$$\boldsymbol{u}_N^{\lambda,\tau}(\boldsymbol{x}, t) = \sum_{\boldsymbol{y}_j \in \tau} \Gamma_j \boldsymbol{\gamma}(\boldsymbol{x}, \boldsymbol{y}_j) \sum_{|\boldsymbol{k}| \le \lambda - 1} a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau) (\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}. \tag{10}$$

The Taylor coefficients of $\mathcal{D}$ are given explicitly by

$$a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau) = \frac{1}{\boldsymbol{k}!} D_{\boldsymbol{y}}^{\boldsymbol{k}} \mathcal{D}(\boldsymbol{x}, \boldsymbol{y})|_{\boldsymbol{y} = \boldsymbol{y}_\tau} = \frac{|\boldsymbol{k}|!}{\boldsymbol{k}!} \left(R^2 + \sigma^2 - \boldsymbol{x} \cdot \boldsymbol{y}_\tau \right)^{-|\boldsymbol{k}| - 1} \boldsymbol{x}^{\boldsymbol{k}}, \tag{11}$$

where $\boldsymbol{k}! = k_1!k_2!k_3!$, $|\boldsymbol{k}| = k_1+k_2+k_3$ for the multi-index $\boldsymbol{k} = (k_1, k_2, k_3)$, and $D_{\boldsymbol{y}}^{\boldsymbol{k}} = \frac{\partial^{|\boldsymbol{k}|}}{\partial y_1^{k_1} \partial y_2^{k_2} \partial y_3^{k_3}}$ and $\boldsymbol{y}^{\boldsymbol{k}} = y_1^{k_1} y_2^{k_2} y_3^{k_3}$ for $\boldsymbol{y} = (y_1, y_2, y_3)$. The explicit representation (11) saves the computational time as in the previous works [4, 20, 16, 21]. In the actual numerical computation, the Taylor coefficients $a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau) = a_{(k_1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau)$ are computed with the following simple recursive formulas:

$$\begin{aligned} a_{(k_1+1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau) &= \frac{|\boldsymbol{k}|+1}{k_1+1} \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_\tau) x_1 \, a_{(k_1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau), \\ a_{(k_1,k_2+1,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau) &= \frac{|\boldsymbol{k}|+1}{k_2+1} \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_\tau) x_2 \, a_{(k_1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau), \\ a_{(k_1,k_2,k_3+1)}(\boldsymbol{x}, \boldsymbol{y}_\tau) &= \frac{|\boldsymbol{k}|+1}{k_3+1} \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_\tau) x_3 \, a_{(k_1,k_2,k_3)}(\boldsymbol{x}, \boldsymbol{y}_\tau). \end{aligned}$$

and $a_{(0,0,0)}(\boldsymbol{x}, \boldsymbol{y}_\tau) = \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}_\tau)$.

With the notation $\boldsymbol{y}_j = (y_{j1}, y_{j2}, y_{j3})$, we reduce the far-field approximation (10) to

$$\boldsymbol{u}_N^{\lambda,\tau}(\boldsymbol{x}, t) = \sum_{|\boldsymbol{k}| \le \lambda - 1} a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau) \left(x_2 C_\tau^{\boldsymbol{k}} - x_3 B_\tau^{\boldsymbol{k}}, x_3 A_\tau^{\boldsymbol{k}} - x_1 C_\tau^{\boldsymbol{k}}, x_1 B_\tau^{\boldsymbol{k}} - x_2 A_\tau^{\boldsymbol{k}} \right), \quad (12)$$

in which $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$ and $C_\tau^{\boldsymbol{k}}$ are defined by

$$A_\tau^{\boldsymbol{k}} = \sum_{\boldsymbol{y}_j \in \tau} \Gamma_j y_{j1} (\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}, B_\tau^{\boldsymbol{k}} = \sum_{\boldsymbol{y}_j \in \tau} \Gamma_j y_{j2} (\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}, C_\tau^{\boldsymbol{k}} = \sum_{\boldsymbol{y}_j \in \tau} \Gamma_j y_{j3} (\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}. \quad (13)$$

These coefficients are assigned to each box $\tau \in F(\boldsymbol{x})$ that contains the point $\boldsymbol{y}_j$. They are computed for all the points in advance and reused in the far-field approximation, which is a key device to evaluate the velocity field faster. Let us note that, in the numerical code, the computation of the Taylor polynomial $(\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}$ is the most time-consuming step. We compute it with a recursive multiplication, which is faster than with the built-in *power* function.

The description of the fast tree-code algorithm is given in Appendix A, which is the same as that of the paper [4] except for the definitions of the far-field condition and the coefficients $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$ and $C_\tau^{\boldsymbol{k}}$. Hence, the computational cost becomes $O(N\lambda^3 \log N)$ as in [4]. Moreover, for example, when we choose the order of the Taylor approximation as $\lambda = O(\log N)$, it becomes $O(N(\log N)^4)$.

2.3 Error estimate

We give an error estimate of the fast algorithm for the point-vortex approximation, i.e. $\sigma = 0$, which is easily extended to the vortex-blob approximation. Let the constant C be defined by

$$C = \max_{1 \le j \le N} \frac{m(\mathcal{A})}{4\pi R} \omega_0(\theta_j, \phi_j) = N \max_{1 \le j \le N} \frac{\Gamma_j}{4\pi R}.$$

Then, the error estimate becomes

$$\begin{aligned} \left| \boldsymbol{u}_N(\boldsymbol{x}, t) - \boldsymbol{u}_N^\lambda(\boldsymbol{x}, t) \right| &= \left| \frac{1}{4\pi R} \sum_{\tau \in F(\boldsymbol{x})} \sum_{\boldsymbol{y}_j \in \tau} \Gamma_j \boldsymbol{\gamma}(\boldsymbol{x}, \boldsymbol{y}_j) \sum_{|\boldsymbol{k}| = \lambda} a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau) (\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}} \right| \\ &\le \frac{C}{N} \sum_{\tau \in F(\boldsymbol{x})} \sum_{\boldsymbol{y}_j \in \tau} |\boldsymbol{\gamma}| \sum_{|\boldsymbol{k}| = \lambda} |a_{\boldsymbol{k}}(\boldsymbol{x}, \boldsymbol{y}_\tau)| |(\boldsymbol{y}_j - \boldsymbol{y}_\tau)^{\boldsymbol{k}}|. \quad (14) \end{aligned}$$

Let us remember that $|\boldsymbol{\gamma}(\boldsymbol{x},\boldsymbol{y}_j)| = |\boldsymbol{x}\times\boldsymbol{y}_j| \le R^2$ due to $|\boldsymbol{x}| = |\boldsymbol{y}_j| = R$ and $|\boldsymbol{y}_j - \boldsymbol{y}_\tau| \le \rho(\tau)$ for $\boldsymbol{y}_j \in \tau$, and the Taylor coefficient $a_{\boldsymbol{k}}(\boldsymbol{x},\boldsymbol{y}_\tau)$ with $|\boldsymbol{k}| = \lambda$ is estimated by

$$|a_{\boldsymbol{k}}(\boldsymbol{x},\boldsymbol{y}_\tau)| \le \frac{\lambda!}{\boldsymbol{k}!}|R^2 - \boldsymbol{x}\cdot\boldsymbol{y}_\tau|^{-\lambda-1}R^{\lambda+1} \le \frac{\lambda!}{\boldsymbol{k}!}\rho^{-\lambda-1}(\tau)h^{\nu(\lambda+1)}R^{\lambda+1}.$$

Here, it follows from

$$(a+b+c)^\lambda = \sum_{|\boldsymbol{k}|=\lambda}\frac{\lambda!}{\boldsymbol{k}!}a^{k_1}b^{k_2}c^{k_3}$$

that $\sum_{|\boldsymbol{k}|=\lambda}\lambda!/\boldsymbol{k}! = 3^\lambda$. We also note that $\rho(\tau) \ge \frac{\sqrt{3}}{2}h$ for arbitrary τ, and $\sum_{\tau\in F(\boldsymbol{x})}\sum_{\boldsymbol{y}_j\in\tau} = \sum_{\tau\in F(\boldsymbol{x})} n_\tau \le N$, where n_τ is the number of the points contained in the box $\tau \in F(\boldsymbol{x})$. Hence, we obtain

$$\begin{aligned}
\left|\boldsymbol{u}_N(\boldsymbol{x},t) - \boldsymbol{u}_N^\lambda(\boldsymbol{x},t)\right| &\le \frac{CR^2}{N}\sum_{\tau\in F(\boldsymbol{x})}\sum_{\boldsymbol{y}_j\in\tau}\sum_{|\boldsymbol{k}|=\lambda}|a_{\boldsymbol{k}}(\boldsymbol{x},\boldsymbol{y}_\tau)|\rho^\lambda(\tau)\\
&\le \frac{CR^{\lambda+3}}{N}\sum_{\tau\in F(\boldsymbol{x})}\sum_{\boldsymbol{y}_j\in\tau}\sum_{|\boldsymbol{k}|=\lambda}\frac{\lambda!}{\boldsymbol{k}!}\rho^{-1}(\tau)h^{\nu(\lambda+1)}\\
&\le \frac{CR^{\lambda+3}}{N}3^\lambda\frac{2}{\sqrt{3}}h^{-1}\sum_{\tau\in F(\boldsymbol{x})}\sum_{\boldsymbol{y}_j\in\tau}h^{\nu(\lambda+1)}\\
&\le C'R^{\lambda+3}3^\lambda h^{\nu(\lambda+1)-1} \le C''h^{\nu\lambda-1}, \qquad (15)
\end{aligned}$$

for some constants C' and C''. Hence, for $\nu = O(3/\lambda)$, we finally have

$$\left|\boldsymbol{u}_N(\boldsymbol{x},t) - \boldsymbol{u}_N^\lambda(\boldsymbol{x},t)\right| \le C''h^2. \qquad (16)$$

The error estimate is the same as the fast algorithm for the planar case [4].

3 Numerical tests

3.1 Efficiency of the algorithm

To confirm the theoretical results given in the previous section, we consider the vortex-blob approximation (5) with $\sigma = 0.05$ as a test problem. Suppose that vortex blobs with the unit strength are uniformly located along M lines of latitude of the sphere with radius $R = 0.5$,

$$z^{(i)} = R - \frac{i}{M+1}, \quad \text{for} \quad i = 1,\dots,M.$$

The positions of the vortex blobs are specified by

$$\boldsymbol{x}_j^{(i)} = \left(\sqrt{R^2 - \left(z^{(i)}\right)^2}\cos 2\pi j/N, \sqrt{R^2 - \left(z^{(i)}\right)^2}\sin 2\pi j/N, z^{(i)}\right), \qquad (17)$$

for $j = 1,\dots,N'$. Thus we obtain the $N = MN'$ vortex blobs, to which we apply the fast algorithm. The configuration is suitable for the numerical test of the fast algorithm, since the points spread over the whole sphere.

The parameters ν, λ and l in the fast algorithm, which appear in the far-field condition (6), the approximation order of the Taylor expansion, and the level of the tree structure of the computational boxes respectively, are determined as $\nu = \frac{1}{n}$ and $l = n$ from the number of the vortex blobs $N = 2^n$. The numerical computation is carried out by Opteron 275 processor with 6GB memory. We vary the

(a) N	$\lambda = 4$	$\lambda = 6$	$\lambda = 8$	$\lambda = 10$
4096	3.25e-4	1.84e-5	1.18e-6	8.50e-8
16384	4.31e-4	2.37e-5	1.49e-6	1.07e-8
65536	4.51e-4	2.50e-5	1.58e-6	1.15e-7
262144	4.74e-4	2.65e-5	1.72e-6	1.27e-7
1048576	4.89e-4	2.74e-5	1.80e-6	1.36e-7
(b) N	$\lambda = 4$	$\lambda = 6$	$\lambda = 8$	$\lambda = 10$
4096	4.23e-4	3.32e-5	1.35e-6	8.35e-8
16384	6.91e-4	3.60e-5	2.09e-6	1.35e-8
65536	7.78e-4	4.07e-5	2.35e-6	1.48e-7
262144	8.05e-4	4.17e-5	2.41e-6	1.53e-7
1048576	8.17e-4	4.20e-5	2.41e-6	1.53e-7

Table 1: (a) L^2 relative error $E_N^{(2)}$ and (b) maximum relative error $E_N^{(\infty)}$ of the velocity field (4) between the fast algorithm and the direct summation for the configuration (17).

approximation order λ from 4 to 10 and calculate the L^2 relative error $E_N^{(2)}$ and the maximum relative error $E_N^{(\infty)}$ between the velocity field evaluated by the fast tree-code algorithm $\boldsymbol{u}_N^\lambda(\boldsymbol{x}_i)$ and that by the direct summation $\boldsymbol{u}_N(\boldsymbol{x}_i)$, which are defined by

$$E_N^{(2)} = \frac{\left(\sum_{i=1}^N \left| \boldsymbol{u}_N(\boldsymbol{x}_i) - \boldsymbol{u}_N^\lambda(\boldsymbol{x}_i) \right|^2 \right)^{\frac{1}{2}}}{\left(\sum_{i=1}^N \left| \boldsymbol{u}_N(\boldsymbol{x}_i) \right|^2 \right)^{\frac{1}{2}}}, \qquad E_N^{(\infty)} = \frac{\max\limits_{1 \le i \le N} \left| \boldsymbol{u}_N(\boldsymbol{x}_i) - \boldsymbol{u}_N^\lambda(\boldsymbol{x}_i) \right|}{\max\limits_{1 \le i \le N} \left| \boldsymbol{u}_N(\boldsymbol{x}_i) \right|},$$

for the given configuration of vortex blobs $\boldsymbol{x}_i$, $i = 1, \ldots, N$. Table 1 shows the L^2 relative error and the maximum relative error for various N and λ. It indicates that the relative errors decrease as λ increases for fixed N. On the other hand, the errors stay in the same order for fixed λ.

Table 2 shows that the computation time of the fast algorithm is smaller than that of direct summation for large N, although it is ineffective for small N. In order to compare the speed-up rate of the present fast tree-code algorithm with the previous one [20], we pick the result for $N = 65536$, $\lambda = 6$ and $\nu = 0.00625$. This shows that the fast tree-code algorithm computes the velocity field 2.94 times faster and the approximation error is 4.1×10^{-5}. On the other hand, the fast tree-code for the two-dimensional vortex sheet with periodic boundary condition computes the velocity field about 50 times faster for $N = 65536$, $\lambda = 8$ and $\nu = 0.05$ and the approximation error is 3.7×10^{-5} [20]. Hence, the present fast algorithm on the sphere is less effective than that for the two-dimensional vortex sheet. This is because, for a given approximation order of the Taylor expansion λ, we need $O(\lambda^3)$ operations to compute the Taylor coefficients (11) and the parameters (13), whereas $O(\lambda^2)$ computations is required for the two-dimensional case.

Now, in order to check the accuracy and the computational efficiency of the fast algorithm given theoretically in the previous section, we rearrange the computational results in Table 3 so that $\lambda\nu \sim 0.5$, and plot them in Figure 1. They indicate that the maximum error decreases like $O(h^2) \sim O(1/N)$ and the computation time increases like $O(N(\log N)^4)$, which supports the theoretical results.

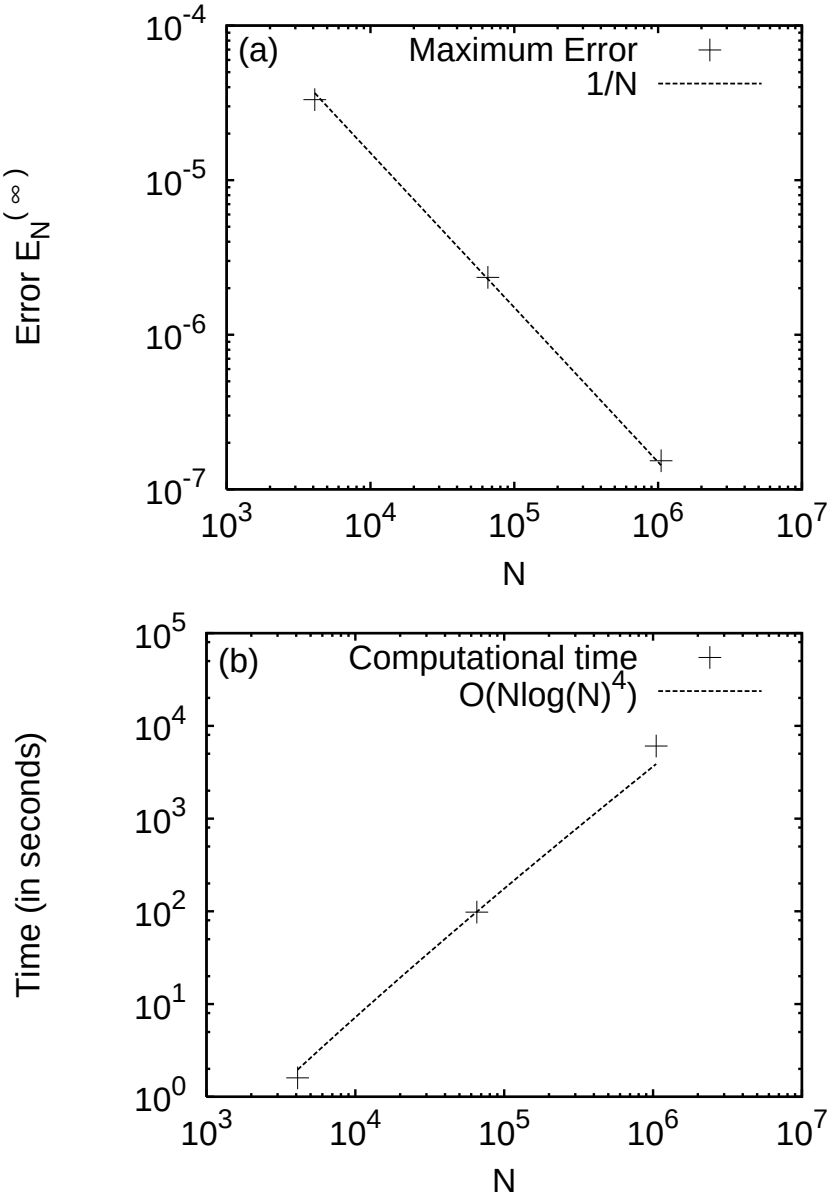

Figure 1: Plot of the maximum error $E_N^{(\infty)}$ and computation time vs. the number of the vortex blobs N. Numerical parameters are given in Table 3.

N	$\lambda = 4$	$\lambda = 6$	$\lambda = 8$	$\lambda = 10$	direct	ν
4096	0.7	1.6	3.1	6.5	0.6	$\nu = 0.083$
16384	4.9	11.9	20.3	34.7	10.2	$\nu = 0.071$
65536	40.3	71.8	98.0	156.2	164.3	$\nu = 0.0625$
262144	310.5	433.2	576.5	908.8	2629.3	$\nu = 0.055$
1048576	2034.4	2694.5	4018.2	6050.5	42060.5	$\nu = 0.05$

Table 2: Computation time in seconds for the evaluation of the velocity fields for the configuration (17).

N	4096	65536	1048576
λ	6	8	10
ν	0.083	0.0625	0.05
$E_N^{(\infty)}$	3.32e-5	2.35e-6	1.53e-7
Time (s)	1.6	98.0	6050.5

Table 3: Error estimate and computation time for the evaluation of the velocity field for the configuration (17), rearranged so that $\lambda\nu$ keeps constant.

3.2 Long time evolution of two vortex sheets

We compute the long-time evolution of two vortex sheets on the sphere as another test. A vortex sheet is a surface across which the velocity field of the incompressible and inviscid fluid changes discontinuously. It has been studied numerically in many papers as not only a mathematical model for the sheet-like coherent vortex structures [14, 15, 21, 22], but also a numerical test problem for the fast tree-code algorithms [4, 5, 16, 20].

Suppose that the two vortex sheets lie along the lines of latitude Θ_1 and Θ_2. Since the vorticity exists only in the vortex sheets, we just discretize each of them with the vortex blob method and track their evolution numerically. The initial position of the ith vortex sheet on the sphere of radius $R = 0.5$ is given by

$$\begin{aligned} z^{(i)}(\alpha) &= R(\cos\Theta_i + \epsilon \sin 2\pi\alpha), \\ x^{(i)}(\alpha) &= \sqrt{R^2 - \left(z^{(i)}\right)^2}\cos 2\pi\alpha, \\ y^{(i)}(\alpha) &= \sqrt{R^2 - \left(z^{(i)}\right)^2}\sin 2\pi\alpha, \end{aligned} \tag{18}$$

for $i = 1$ and 2. The parameter $\alpha \in [0, 1)$ is a Lagrangian variable moving with the fluid particle. A small first-mode perturbation is imposed on $z^{(i)}$ in (18), since the two vortex sheets rotate in the longitudinal direction with some constant speeds without the perturbation and thus we observe no complex interaction between them. The stability of the first-mode perturbation is given in Appendix B, which indicates that the stability depends on the values of Θ_1 and Θ_2. The amplitude of the disturbance is $\epsilon = 0.02$. Then we discretize $0 \le \alpha \le 1$ by N points to obtain the N vortex blobs. Since the strengths of the vortex sheets are assumed to be uniform, the strengths of the vortex blobs are identical. Accordingly, the initial positions of the vortex blobs are given by $\boldsymbol{x}_m^{(i)} \approx \boldsymbol{x}^{(i)}(\frac{m}{N})$ with the same strength $\Gamma_m^{(i)} = 1/N \equiv \Gamma$ for $i = 1, 2$ and $m = 1, 2, \ldots, N$. With the initial data, we solve numerically the

following $2N$-dimensional ordinary differential equations for $m = 1, \ldots, N$.

$$\frac{d\boldsymbol{x}_m^{(1)}}{dt} = -\frac{\Gamma}{4\pi R}\sum_{j\neq m}^{N}\frac{\boldsymbol{x}_m^{(1)}\times\boldsymbol{x}_j^{(1)}}{R^2+\sigma^2-\boldsymbol{x}_m^{(1)}\cdot\boldsymbol{x}_j^{(1)}} - \frac{\Gamma}{4\pi R}\sum_{j=1}^{N}\frac{\boldsymbol{x}_m^{(1)}\times\boldsymbol{x}_j^{(2)}}{R^2+\sigma^2-\boldsymbol{x}_m^{(1)}\cdot\boldsymbol{x}_j^{(2)}} \quad (19)$$

$$\frac{d\boldsymbol{x}_m^{(2)}}{dt} = -\frac{\Gamma}{4\pi R}\sum_{j\neq m}^{N}\frac{\boldsymbol{x}_m^{(2)}\times\boldsymbol{x}_j^{(2)}}{R^2+\sigma^2-\boldsymbol{x}_m^{(2)}\cdot\boldsymbol{x}_j^{(2)}} - \frac{\Gamma}{4\pi R}\sum_{j=1}^{N}\frac{\boldsymbol{x}_m^{(2)}\times\boldsymbol{x}_j^{(1)}}{R^2+\sigma^2-\boldsymbol{x}_m^{(2)}\cdot\boldsymbol{x}_j^{(1)}} \quad (20)$$

The first summations on the right hand side of the equation represent the self-induced velocity, while the second ones are the interaction between the two vortex sheets. The temporal integration is carried out with the fourth order Runge-Kutta method, whose time step size is $\Delta t = 0.05$. Each of the vortex sheets is discretized by $N = 131072$, i.e. we use 262144 vortex blobs in total. The regularization parameter is $\sigma = 0.05$. We set the parameters of the fast tree-code algorithm as $\nu = 0.055$ and $\lambda = 10$. Then, according to Table 1 and Table 2, the fast tree-code algorithm evaluates the velocity field 3 times faster than the direct summation method with approximation error of $O(10^{-7})$, which is required accuracy for the computation of vortex sheets. As we have mentioned in Introduction, we verify if the vortex blobs stay on the surface of sphere by checking the condition $||\boldsymbol{x}_m| - R| < 1.0 \times 10^{-7}$ for all m every time step. We hardly observe the move-off of the vortex blobs from the surface of sphere in the numerical examples given here.

Before showing numerical results, let us discuss a numerical difficulty in the numerical computation of vortex sheets. According to the linear stability analysis of a single vortex sheet on the sphere [22], a small perturbation grows very rapidly due to the Kelvin-Helmholtz instability and thus the evolution of vortex sheets becomes an ill-posed problem in the sense of Hadamard. Hence, the accumulation of round-off error inevitably deteriorates the accuracy of the numerical computation. What makes the matter worse, since the fast algorithm gives rise to not only round-off error but also the approximation error, these errors affect the computational result more seriously. In order to maintain the accuracy of the numerical computation, we adopt *the Fourier filtering technique*. As for the detailed description of the technique, we would like the readers to refer to the paper [15]. The threshold of the filter is set to 1.0×10^{-9}.

Figure 2 shows the evolution of the two vortex sheets up to $t = 26$. The base latitudes of the two vortex sheets are given by $\Theta_1 = 0.3\pi$ and $\Theta_2 = 0.5\pi$. The first-mode disturbance in (18) is linearly unstable as is shown in Appendix B. Since the two vortex sheets are well separated compared to the amplitude of the initial disturbance $\epsilon = 0.02$, each of the vortex sheets becomes unstable independently due to the Kelvin-Helmholtz instability before they interact with each other. The vortex sheet in the northern hemisphere becomes unstable earlier than that near the equator. Then they evolve into structures with many rolling-up spirals and very thin filaments. After that, the two vortex sheets interact strongly and become complex.

Another computational result is shown in Figure 3 for the configuration (18) with $\Theta_1 = \pi/3 - \pi/20$ and $\Theta_2 = \pi/3 + \pi/20$ and $\epsilon = 0.02$. Linear stability analysis in Appendix B shows that the first-mode disturbance is linearly unstable. Since the growth rate of the disturbance is larger than that of the first example, the northern vortex sheet becomes unstable earlier. Afterwards it rolls up due to the Kelvin-Helmholtz instability immediately, the two vortex sheets begin interacting with each other before the southern vortex sheet rolls up, since the initial distance between them is closer than the first example. As a result of the complex interaction, the two vortex sheets gather in the northern polar region and a big coherent spiral structure is separated out of the region and stays around the equator, which is also a complicated pattern.

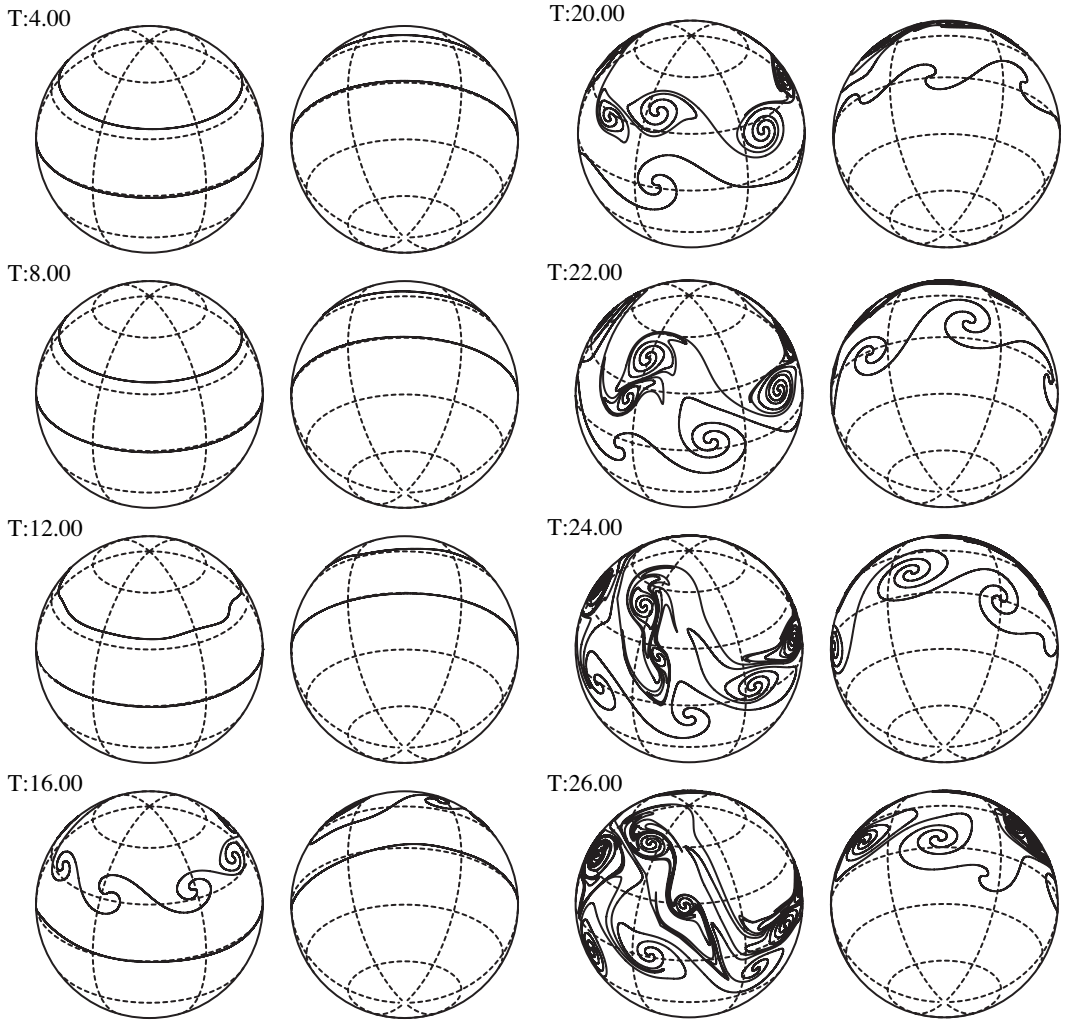

Figure 2: Long time evolution of two vortex sheets for $\Theta_1 = 0.3\pi$ and $\Theta_2 = 0.5\pi$ up to $t = 26$. Each vortex sheet is approximated by 131072 vortex blobs.

Both numerical results with $N = 262144$ vortex blobs keep the fine resolution up to $t = 26$. It takes twenty days to compute the evolution with the Opteron 275 processor. It is quite difficult to compute the evolution for such a long time without the fast tree-code algorithm. Let us finally mention how the regularization parameter affects the numerical results. As was shown in [15], in some range of σ, the numerical result is qualitatively similar except that the spirals tend to have infinitely many windings as $\sigma \to 0$. On the other hand, for excessively small σ, the numerical computation loses its accuracy due to the ill-posedness of the problem. In order to keep the accuracy for smaller σ, more discretizing vortex blobs are required.

4 Summary and discussion

A fast tree-code algorithm to evaluate the interaction between point vortices and vortex blobs on a sphere has been described. The fast method is applicable to the numerical computation of inviscid incompressible flow on the sphere, since the

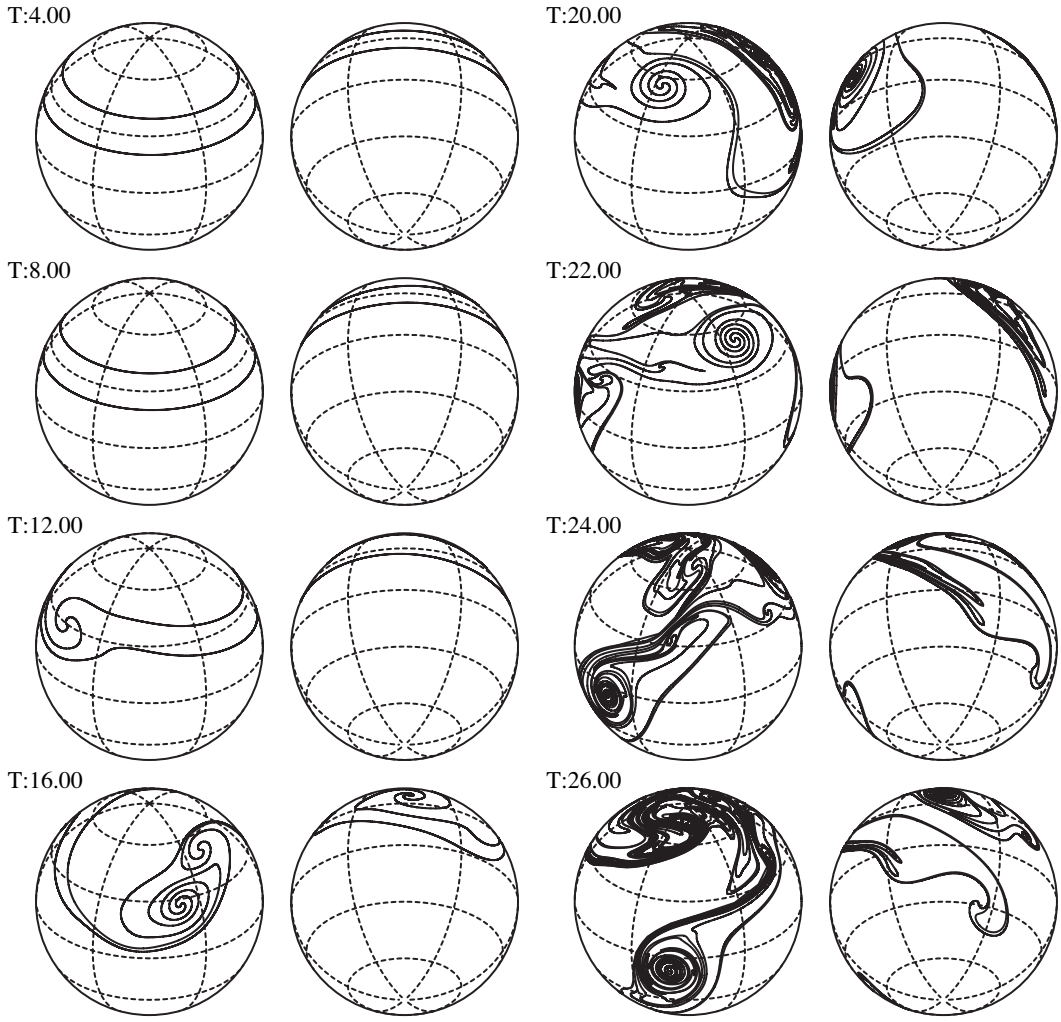

Figure 3: Long time evolution of two vortex sheets up to $t = 26$. The initial base latitudes for them are given by $\Theta_1 = \pi/3 - \pi/20$ and $\Theta_2 = \pi/3 + \pi/20$ respectively.

motion of the N discretizing points is derived when the Euler equations are approximated by the vortex method.

The tree-code algorithm to the use of flow over a sphere is an extension of the fast algorithm developed by Draghicescu [4] for the two-dimensional and three-dimensional Euler equations in Euclidean space. However, a naive implementation of the algorithm to eqs. (2) and (3) on the sphere is practically ineffective, since the high-order Taylor coefficients of the velocity field in the equations are expressed in a complicated way. We resolve the difficulty by representing the equations of the N points in the three-dimensional Cartesian coordinates, for which the Taylor coefficients are easily computed with a simple formula. Owing to the representation of the equation (5), the fast algorithm works very effectively. The error estimate shows that the approximation error decreases as $O(1/N)$ with an appropriate choice of the parameters in the algorithm. The computational cost is largely reduced to $O(N(\log N)^4)$, which is the same order as the three-dimensional tree-code algorithm [4]. The efficiency of the fast method is examined and thus the theoretical error estimate and the reduction of computational cost are confirmed. We also compute the long-time evolution of two vortex sheets, which shows very complex interactions between them for a long time.

Let us finally discuss possible future directions. The fast algorithm gives us a new way to investigate many interesting problems such as the statistical theory of point vortices on the sphere [12]. On the other hand, it is unavailable to compute the Euler equations with the Coriolis force on the rotating sphere, since the vorticity is no longer a conserved quantity. However, we have two possible extensions of the vortex method to the rotating problem. The first method is that we approximate the effect of rotation by strips of constant vorticity corresponding to the solid body rotation of the sphere and then compute the interaction between the point vortices and the vorticity strips with the vortex method and the contour dynamics method. This idea was used to observe the interaction between a coherent vortex structure and the background rotation [3, 18]. Another possibility is to develop a Lagrangian method based on the invariance of the potential vorticity for the Euler flows on the rotating sphere. The fast tree-code algorithm is available to compute the velocity field at the grid points for a given distribution of the point "potential" vortices. This idea will be examined in the future research.

Acknowledgments

This work is partially supported by Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (A) #17684002 2007, Grant-in-Aid for Exploratory # 17654018 2007, and Grant-in-Aid for formation of COE at Hokkaido University.

A Description of the fast tree-code algorithm

Here we give the formal description of the tree-code algorithm. First, we define the following two recursive algorithms called in the main algorithm.

Algorithm 2. *(Compute the coefficients $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$, $C_\tau^{\boldsymbol{k}}$ in (13) for all the node boxes τ containing the discretizing point at $\boldsymbol{y}$ with strength Γ)*

Input: box τ, integer k, position of the discretizing point $\boldsymbol{y}$ with strength Γ;
ComputeNodeCoefficients(τ, $\boldsymbol{k}$, $\boldsymbol{y}$, Γ)
if $\tau = \emptyset$ return;
if $\boldsymbol{y} \in \tau$ then for all $\boldsymbol{k}$, $|\boldsymbol{k}| \leq \lambda - 1$

add $\Gamma y_1(\boldsymbol{y}-\boldsymbol{y}_\tau)^{\boldsymbol{k}}$ to $A_\tau^{\boldsymbol{k}}$;
add $\Gamma y_2(\boldsymbol{y}-\boldsymbol{y}_\tau)^{\boldsymbol{k}}$ to $B_\tau^{\boldsymbol{k}}$;
add $\Gamma y_3(\boldsymbol{y}-\boldsymbol{y}_\tau)^{\boldsymbol{k}}$ to $C_\tau^{\boldsymbol{k}}$;
if $k = 3l$ then
add $\boldsymbol{y}$ to the list of the near-field, $L(\tau)$;
return;
else
Recursively call ComputeNodeCoefficients(τ_i, k+1, $\boldsymbol{y}$, Γ) for all the children of τ, τ_1 and τ_2;
return;
end
end
end

End of Algorithm

Algorithm 3. *(Compute the velocity field at $\boldsymbol{y}$)*

Input: box τ, integer k, the position of the discretizing point $\boldsymbol{y}$;
ComputeFarNearField(τ, k, $\boldsymbol{y}$)
if $\tau = \emptyset$ return;
if $\rho(\tau) < h^\nu |R^2 - \boldsymbol{y}\cdot\boldsymbol{y}_\tau|$ then
Compute the far-field approximation according to (12);
return;
else
if $k = 3l$ then
Compute the contribution from the points in the near-field list $L(\tau)$ directly;
return;
else
Recursively call ComputeFarNearField(τ_i, $k+1$, $\boldsymbol{y}$) for all the children of τ, τ_1 and τ_2;
return;
end
end
end

End of Algorithm

Then we finally have the description of the fast tree-code algorithm in a simple form.

Algorithm 4. *(Fast Tree-code for the vortex method on the sphere)*

Input: integer n; real λ; real ν; real Γ_j; real $\boldsymbol{y}_j$, $j = 1,\ldots,n$;
Output: $\boldsymbol{u}_h^\lambda(\boldsymbol{x},t)$ for all $\boldsymbol{x} = \boldsymbol{y}_j$, $j = 1,\ldots,n$;
Stage 0 (mesh generation, done only once)
GenerateMesh($\mathcal{B}$, 0);
end
Stage 1 (Compute coefficients $A_\tau^{\boldsymbol{k}}$, $B_\tau^{\boldsymbol{k}}$, $C_\tau^{\boldsymbol{k}}$)
For $j = 1,\ldots,n$, call ComputeNodeCoefficients($\mathcal{B}$, 0, $\boldsymbol{y}_j$, Γ_j);
end
Stage 2 (Compute the velocity field (8))
For $j = 1,\ldots,n$, call ComputeFarNearField($\mathcal{B}$, 0, $\boldsymbol{y}_j$);
end

End of Algorithm

B Linear stability of the first-mode perturbations for two vortex sheets

Linear stability analysis for a single vortex sheet on the sphere with pole vortices [22] revealed that the low-mode spectra of perturbations become neutrally stable due to the effect of the curvature of the sphere or the outer flow induced by the pole vortices. On the other hand, high-mode spectra are always linearly unstable and their exponential growth rate is proportional asymptotically to their mode-number, which is known as the Kelvin-Helmholtz instability.

In this appendix, we consider the linear stability of the first-mode spectra of perturbations imposed on two vortex sheet. Suppose that the positions of the two vortex sheets are represented by $(\theta_i(\alpha,t),\phi_i(\alpha,t))$ for $i=1,2$ in the spherical coordinates, and their strengths are identical. Then, with the following two functions F and G,

$$\begin{aligned} F(\theta,\phi,\theta',\phi') &= -\frac{1}{4\pi}\frac{\sin\theta'\sin(\phi-\phi')}{1-\cos\theta\cos\theta'-\sin\theta\sin\theta'\cos(\phi-\phi')}, \\ G(\theta,\phi,\theta',\phi') &= -\frac{1}{4\pi\sin\theta}\frac{\cos\theta\sin\theta'\cos(\phi-\phi')-\sin\theta\cos\theta'}{1-\cos\theta\cos\theta'-\sin\theta\sin\theta'\cos(\phi-\phi')}, \end{aligned}$$

the equations of motion of the two vortex sheets on the unit sphere are described as follows.

$$\frac{\partial\theta_1}{\partial t} = \mathrm{PV}\int_0^{2\pi} F(\theta_1,\phi_1,\theta_1',\phi_1')d\beta + \int_0^{2\pi} F(\theta_1,\phi_1,\theta_2',\phi_2')d\beta, \tag{21}$$

$$\frac{\partial\theta_2}{\partial t} = \mathrm{PV}\int_0^{2\pi} F(\theta_2,\phi_2,\theta_2',\phi_2')d\beta + \int_0^{2\pi} F(\theta_2,\phi_2,\theta_1',\phi_1')d\beta, \tag{22}$$

$$\frac{\partial\phi_1}{\partial t} = \mathrm{PV}\int_0^{2\pi} G(\theta_1,\phi_1,\theta_1',\phi_1')d\beta + \int_0^{2\pi} G(\theta_1,\phi_1,\theta_2',\phi_2')d\beta, \tag{23}$$

$$\frac{\partial\phi_2}{\partial t} = \mathrm{PV}\int_0^{2\pi} G(\theta_2,\phi_2,\theta_2',\phi_2')d\beta + \int_0^{2\pi} G(\theta_2,\phi_2,\theta_1',\phi_1')d\beta, \tag{24}$$

in which $\theta_i=\theta_i(\alpha,t)$, $\phi_i=\phi_i(\alpha,t)$, $\theta_i'=\theta_i(\beta,t)$ and $\phi_i'=\phi_i(\beta,t)$. The first terms on the right hand side of the equations are defined in the sense of Cauchy's principal value integral. See [22] for the derivation of the equations. Let us note that we have eqs. (19) and (20), when we discretize the equations (21) – (24) with the vortex blob method and describe them in the three-dimensional Cartesian coordinates.

When the two vortex sheets lie along the lines of latitude, i.e. $\theta_i(\alpha,t)=\Theta_i$ and $\phi_i(\alpha,t)=\alpha$ for $i=1,2$, the configuration is a stationary solution for the equations. We also assume that $\Theta_1<\Theta_2$ without loss of generality. Then we consider the following linear stability of the first-mode spectra $\vartheta_{\pm1}^{(i)}$ and $\varphi_{\pm1}^{(i)}$ for $i=1,2$:

$$\theta_i(\alpha,t)=\Theta_i+\vartheta_{\pm1}^{(i)}(t)\exp(\pm\mathrm{i}\alpha),\quad \phi_i(\alpha,t)=\alpha+\varphi_{\pm1}^{(i)}(t)\exp(\pm\mathrm{i}\alpha).$$

The linearized terms for the first singular integrals in the equations have already been given in [22]. For the equations (21) and (22), they are

$$\frac{1}{2\sin\Theta_i}\varphi_{\pm1}^{(i)}, \tag{25}$$

and for the equations (23) and (24),

$$\left(\frac{1}{2\sin^3\Theta_i}-\frac{1+\cos^2\Theta_i}{2\sin^3\Theta_i}\right)\vartheta_{\pm1}^{(i)}(t). \tag{26}$$

Regarding the linearized terms for the second integrals, we compute them with the computer software, MATHEMATICA 6; For the equation (21), it follows from (25) and

$$\begin{aligned}
\int_0^{2\pi} \frac{\partial F}{\partial \theta_1}(\Theta_1, \Theta_2, \alpha, \beta) d\beta &= \int_0^{2\pi} \frac{\partial F}{\partial \phi_1}(\Theta_1, \Theta_2, \alpha, \beta) d\beta = 0, \\
\int_0^{2\pi} \frac{\partial F}{\partial \theta_2'}(\Theta_1, \Theta_2, \alpha, \beta) \exp(\pm \mathrm{i}\beta) d\beta &= \frac{\pm \mathrm{i} AB}{2 \sin\Theta_1 \sin\Theta_2} \exp(\pm \mathrm{i}\alpha), \\
\int_0^{2\pi} \frac{\partial F}{\partial \phi_2'}(\Theta_1, \Theta_2, \alpha, \beta) \exp(\pm \mathrm{i}\beta) d\beta &= -\frac{AB}{2 \sin\Theta_1} \exp(\pm \mathrm{i}\alpha),
\end{aligned}$$

in which A and B are defined by

$$A = \cot\Theta_2 + \frac{1}{\sin\Theta_2}, \qquad B = \cot\Theta_1 - \frac{1}{\sin\Theta_1},$$

that we have the linearized equations for $\vartheta_{\pm 1}^{(1)}(t)$.

$$\frac{d\vartheta_{\pm 1}^{(1)}}{dt} = \frac{\pm \mathrm{i} AB}{2 \sin\Theta_1 \sin\Theta_2} \vartheta_{\pm 1}^{(2)} + \frac{1}{2 \sin\Theta_1} \varphi_{\pm 1}^{(1)} - \frac{AB}{2 \sin\Theta_1} \varphi_{\pm 1}^{(2)}.$$

Similarly, we have the linearized equations for (22).

$$\frac{d\vartheta_{\pm 1}^{(2)}}{dt} = \frac{\mp \mathrm{i} AB}{2 \sin\Theta_1 \sin\Theta_2} \vartheta_{\pm 1}^{(1)} - \frac{AB}{2 \sin\Theta_2} \varphi_{\pm 1}^{(1)} + \frac{1}{2 \sin\Theta_2} \varphi_{\pm 1}^{(2)}.$$

The linearized equation of (23) for $\varphi_{\pm 1}^{(1)}$ becomes

$$\frac{d\varphi_{\pm 1}^{(1)}}{dt} = -\frac{2\cos^2\Theta_1 - 2\cos\Theta_1 + 1}{2\sin^3\Theta_1} \vartheta_{\pm 1}^{(1)} - \frac{AB}{2\sin^2\Theta_1 \sin\Theta_2} \vartheta_{\pm 1}^{(2)} \mp \frac{\mathrm{i} AB}{2\sin^2\Theta_1} \varphi_{\pm 1}^{(2)},$$

in which the linearized terms for the second integrals in (23) come from the following integrals.

$$\begin{aligned}
\int_0^{2\pi} \frac{\partial G}{\partial \phi_1}(\Theta_1, \Theta_2, \alpha, \beta) d\beta &= 0, \\
\int_0^{2\pi} \frac{\partial G}{\partial \theta_1}(\Theta_1, \Theta_2, \alpha, \beta) d\beta &= -\frac{\cos^2\Theta_1 - 2\cos\Theta_1 + 1}{2\sin^3\Theta_1}, \\
\int_0^{2\pi} \frac{\partial G}{\partial \phi_2'}(\Theta_1, \Theta_2, \alpha, \beta) \exp(\pm \mathrm{i}\beta) d\beta &= \frac{\mp \mathrm{i} AB}{2\sin^2\Theta_1} \exp(\pm \mathrm{i}\alpha), \\
\int_0^{2\pi} \frac{\partial G}{\partial \theta_2'}(\Theta_1, \Theta_2, \alpha, \beta) \exp(\pm \mathrm{i}\beta) d\beta &= -\frac{AB}{2\sin^2\Theta_1 \sin\Theta_2} \exp(\pm \mathrm{i}\alpha).
\end{aligned}$$

The linearized equation for $\varphi_{\pm 1}^{(2)}$ is similarly obtained as follows.

$$\frac{d\varphi_{\pm 1}^{(2)}}{dt} = -\frac{AB}{2\sin^2\Theta_2 \sin\Theta_1} \vartheta_{\pm 1}^{(1)} - \frac{2\cos^2\Theta_2 + 2\cos\Theta_2 + 1}{2\sin^3\Theta_2} \vartheta_{\pm 1}^{(2)} \pm \frac{\mathrm{i} AB}{2\sin^2\Theta_2} \varphi_{\pm 1}^{(1)},$$

in which the linearized term for $\vartheta_{\pm 1}^{(2)}$ is derived from

$$\int_0^{2\pi} \frac{\partial G}{\partial \theta_2}(\Theta_1, \Theta_2, \alpha, \beta) d\beta = -\frac{\cos^2\Theta_2 + 2\cos\Theta_2 + 1}{2\sin^3\Theta_2},$$

for the equation (24). Consequently, we have the following linearized matrix for the first-mode spectra.

$$\frac{d}{dt}\begin{pmatrix} \vartheta_{\pm 1}^{(1)} \\ \vartheta_{\pm 1}^{(2)} \\ \varphi_{\pm 1}^{(1)} \\ \varphi_{\pm 1}^{(2)} \end{pmatrix} = \mathcal{S}^{\pm}\begin{pmatrix} \vartheta_{\pm 1}^{(1)} \\ \vartheta_{\pm 1}^{(2)} \\ \varphi_{\pm 1}^{(1)} \\ \varphi_{\pm 1}^{(2)} \end{pmatrix},$$

where

$$\mathcal{S}^{\pm} = \begin{pmatrix} 0 & \frac{\pm \mathrm{i} AB}{2\sin\Theta_1 \sin\Theta_2} & \frac{1}{2\sin\Theta_1} & -\frac{AB}{2\sin\Theta_1} \\ \frac{\mp \mathrm{i} AB}{2\sin\Theta_1 \sin\Theta_2} & 0 & -\frac{AB}{2\sin\Theta_2} & \frac{1}{2\sin\Theta_2} \\ -\frac{2\cos^2\Theta_1 - 2\cos\Theta_1 + 1}{2\sin^3\Theta_1} & -\frac{AB}{2\sin^2\Theta_1 \sin\Theta_2} & 0 & \frac{\mp \mathrm{i} AB}{2\sin^2\Theta_1} \\ -\frac{AB}{2\sin^2\Theta_2 \sin\Theta_1} & -\frac{2\cos^2\Theta_2 + 2\cos\Theta_2 + 1}{2\sin^3\Theta_2} & \frac{\pm \mathrm{i} AB}{2\sin^2\Theta_2} & 0 \end{pmatrix}.$$

Suppose that λ^+ is the eigenvalue of $\mathcal{S}^+$, then $\lambda^- = \bar{\lambda}^+$ is the eigenvalue of $\mathcal{S}^-$ due to $\mathcal{S}^+ = \bar{\mathcal{S}^-}$. Hence, the spectra $\vartheta_{\pm 1}^{(i)}$ and $\varphi_{\pm 1}^{(i)}$ are linearly unstable if $\mathrm{Re}(\lambda^+) > 0$.

Let us apply the stability analysis to the examples given in this paper. We compute the eigenvalues of $\mathcal{S}^+$ numerically. For the first example in which $\Theta_1 = 0.3\pi$ and $\Theta_2 = 0.5\pi$, the four eigenvalues become $\lambda^+ = -0.573599\mathrm{i}$, $0.5\mathrm{i}$, $0.36977 + 0.0367995\mathrm{i}$ and $-0.36977 + 0.0367995\mathrm{i}$, which indicates that the first-mode disturbance is linearly unstable. As for the second case where $\Theta_1 = \pi/3 - \pi/20$ and $\Theta_2 = \pi/3 + \pi/20$, they are $\lambda^+ = 0.839196\mathrm{i}$, $-0.643002\mathrm{i}$, $0.684347 - 0.980968\mathrm{i}$ and $-0.684347 - 0.980968\mathrm{i}$. The growth rate of the first-mode spectra for this case is larger than that for the first example.

Finally, we note that the first-mode spectra are not always linearly unstable. Figure B1 shows the maximum of the real part of the four eigenvalues of $\mathcal{S}^+$, in which we fix $\Theta_1 = 0.3\pi$ and change $\Theta_1 < \Theta_2 < \pi$. For $\Theta_2 < \Theta_c \approx 0.62\pi$, the two eigenvalues are pure imaginary numbers and the other two have non-zero real parts. However, for $\Theta_2 > \Theta_c$, all the eigenvalues become pure imaginary numbers and thus the first-mode spectra are neutrally stable.

References

[1] J. Barnes and P. Hut, "A hierarchical $O(N \log N)$ force-calculation algorithm", Nature, vol. 324, pp. 446–449, 1986.

[2] G.-H. Cottet and P. D. Koumoutsakos, "Vortex methods, theory and practice", Cambridge Univ. Press, 1994.

[3] M. T. DiBattista and L. M. Polvani, "Barotropic vortex pairs on a rotating sphere", J. Fluid Mech., vol. 358, pp. 107–133, 1998.

[4] C. I. Draghicescu, "An efficient implementation of particle methods for the incompressible Euler equations", SIAM J. Numer. Anal., vol. 31 No. 4, pp. 1090-1108, 1994.

[5] C. I. Draghicescu and M. Draghicescu, "A fast algorithm for vortex blob interactions", J. Comput. Phys., vol. 116, pp. 69–78, 1995.

[6] D. G. Dritschel, "Contour dynamics/surgery on the sphere", J. Comput. Phys., vol. 79, pp.477-483, 1988.

[7] D. G. Dritschel and L. M. Polvani, "The roll-up of vorticity strips on the surface of a sphere", J. Fluid Mech., vol. 234, pp. 47–69, 1992.

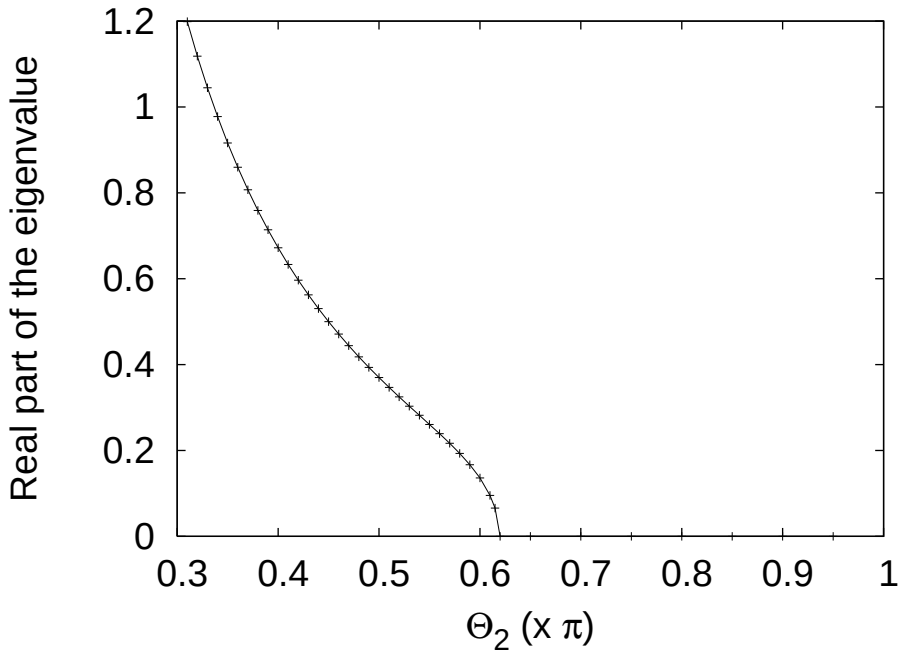

Figure B1: Maximum real part among the eigenvalues λ^+ of the linearized matrix $\mathcal{S}^+$ for $\Theta_2 \in (\Theta_1, \pi)$ with $\Theta_1 = 0.3\pi$.

[8] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations", J. Comput. Phys., vol. 73, pp. 325–348, 1987.

[9] L. Greengard and V. Rokhlin, "Rapid evaluations of potential fields in three dimensions", Springer Lecture Notes in Mathematics, vol. 1360, Springer, Berlin, pp. 121–141, 1988.

[10] J. Goodman, T. Y. Hou and J. Lowengrub, "Convergence of the point vortex method for the 2-D Euler equations", Comm. Pure Appl. Math., vol. 43, pp. 415–430, 1990.

[11] Y. Kimura and H. Okamoto, "Vortex motion on a sphere", J. Phys. Soc. Japan, vol. 56, pp, 4203–4206, 1988.

[12] C.C. Lim and J. Nebus, "Vorticity, Statistical Mechanics, and Monte Carlo Simulation", Springer Monographs in Mathematics, Springer-Verlag, New York, 2006.

[13] T. Y. Hou, J. Lowengrub and R. Krasny, "Convergence of a point-vortex method for vortex sheets", SIAM J. Numer. Anal., vol. 28, pp. 308–320, 1991.

[14] R. Krasny, "A study of singularity formation in a vortex sheet by the point-vortex approximation", J. Fluid Mech., vol. 167, pp. 65–93, 1986.

[15] R. Krasny, "Desingularization of periodic vortex sheet roll-up", J. Comput. Phys, vol. 65, pp. 292–313, 1986.

[16] K. Lindsay and R. Krasny, "A particle method and adaptive treecode for vortex motion in three-dimensional flow", J. Comput. Phys., vol. 172, pp. 879–907, 2001.

[17] P. K. Newton, "The N-vortex problem, analytical techniques", Springer-Verlag, 2001.

[18] P. K. Newton and T. Sakajo, "The N-vortex problem on a rotating sphere. III. Ring configurations coupled to a background field", Proc. R. Soc. A, vol. 463, pp. 961–977, 2007.

[19] L. M. Polvani and D. G. Dritschel, "Wave and vortex dynamics on the surface of a sphere", J. Fluid Mech., vol. 255, pp. 35–64, 1993.

[20] T. Sakajo and H. Okamoto, "An application of Draghicescu's fast summation method to vortex sheet motion", J. Phys. Soc. Japan, vol 67, No. 2, pp.462–470, 1998.

[21] T. Sakajo, "Numerical computation of a three-dimensional vortex sheet in a swirl flow", Fluid Dyn. Res., vol. 28, pp.423–448, 2001.

[22] T. Sakajo, "Motion of a vortex sheet on a sphere with pole vortices", Phys. Fluids, vol. 16, pp. 717–727, 2004.