数理解析研究所講究録
974巻 1996年 1-20



# An application of Draghicescu's fast summation method to vortex sheet motion

Takashi SAKAJO and Hisashi OKAMOTO

Research Institute for Mathematical Sciences
Kyoto University, Kyoto, 606-01 Japan

September 2, 1996

**Abstract**

The fast summation method of Draghicescu is applied to computation of 2D vortex sheet motions. In the present paper we report our numerical experiments which show the effectiveness as well as difficulties of Draghicescu's fast summation method in 2D vortex-sheet computations. For instance, the fast summation method is nearly five times faster than the direct summation method when we use $16,384 = 2^{14}$ vortex blobs. On the other hand, if it is used together with the Fourier filter, the method is found to be sensitive to the filter threshold.

## 1 Introduction

We perform some numerical experiment on Draghicescu's fast summation method. It is applied to a computation of 2D vortex sheet motion. Our aim is to compare the performance of the fast summation with that of the direct summation.

What we consider is the Birkhoff-Rott equation with periodic boundary condition ( Krasny [6, 7] ). Namely, we consider the following equation:

$$\frac{\partial z(t,\Gamma)^*}{\partial t} = \frac{1}{2i}\,\mathrm{p.v.}\int_0^1 \cot\pi\left(z(t,\Gamma)-z(t,\Gamma')\right)d\Gamma', \tag{1}$$

where $t$ represents time, the integral is Cauchy's principal value, $i=\sqrt{-1}$, and $*$ implies the complex conjugate.

The vortex sheet is represented by the curve $z(t,\Gamma) = x(t,\Gamma) + iy(t,\Gamma)$ with $\Gamma$ being taken along the curve. $\Gamma$ is called the circulation parameter. The physical space is filled with an inviscid incompressible fluid, whose motion is irrotational everywhere except on a certain surface. This surface is the support of the vorticity. We consider two dimensional

motions and $z(t,\Gamma)$ represent a section of the surface at time $t$. For further detail about the Birkhoff-Rott equation, see [6, 7, 9].

Krasny [6, 7] solved (1) with Chorin's vortex blob approximation and the Fourier filter. In the present paper, we add another technique, Draghicescu's fast summation method, to Krasny's method. We then compare the performances. In the course of our study, we found that Draghicescu's method directly applied to the equation (1) has some difficulty in programming. The difficulty, which we will explain in section 3, can be reduced if we put

$$w = \exp(2\pi i z)$$

and if (1) is rewritten as

$$\frac{\partial w(t,\Gamma)^*}{\partial t} = -\pi i w^* \text{p.v.} \int_0^1 \frac{w(\Gamma,t) + w(\Gamma',t)}{w(\Gamma,t) - w(\Gamma',t)} d\Gamma'. \tag{2}$$

The reason why this equation has advantage over (1) will be given in section 3.

The present paper consists of five sections. In section 2, we explain the numerical method. In section 3, we report what we obtained in our numerical experiments. Here we focus on the computational advantage/disadvantage of the method. Section 4 is devoted to phenomenological description of the sheet. The relation between the fast summation and Fourier filter is discussed in section 5. Concluding remarks are presented in section 6.

# 2 $\delta$ Equation and Discretization

In the computation, we use a smoothed version of (2). Namely, we follow Krasny's idea ([7]) of desingularization. Specifically, we consider

$$\frac{\partial w(t,\Gamma)^*}{\partial t} = \int_0^1 K_\delta(w(t,\Gamma), w(t,\Gamma')) d\Gamma'. \tag{3}$$

where

$$K_\delta(w,v) = -\pi i w^* \frac{(w+v)(w^*-v^*)}{|w-v|^2 + \delta^2}.$$

Note that the equation (3) reduces to (2) when $\delta = 0$. We solve (3) with some initial condition, $w(0,\Gamma)$.

The equation (1) is known to be ill-posed ( see, e.g., Caflisch et al. [1] ). The smoothed version (3) is, however, well-posed for any time interval if $\delta > 0$. The high frequency modes are stabilized, see for instance [7].

We discretize (3) by the point vortices. Choosing a positive integer $N$, we consider the following system of ordinary differential equations:

$$\frac{\partial w_n(t)^*}{\partial t} = \frac{1}{N} \sum_{1 \le m \le N} K_\delta(w_n, w_m) \equiv u_n(t) \qquad (1 \le n \le N) \tag{4}$$

with the following initial condition:

$$w_n(0) = \exp\left(\frac{2\pi in}{N} - \pi + 2\pi i\epsilon(1-i)\sin\left(\frac{2\pi in}{N}\right)\right) \qquad (n = 1, 2, \cdots, N). \tag{5}$$

Here $\epsilon$ is a constant, which we fix to $\epsilon = 0.01$. This initial date is a rescaled function from the one considered in Krasny [7]. The reason of rescaling is that after transformation, vortex sheet is contained in the square $-0.5 \leq x, y \leq 0.5$ in $\mathbf{C}$.

When we get $\{w_n(t)\}_{1\leq n\leq N}$, which approximate the vortex sheet at time $t$, we integrate (4) to obtain $\{w_n(t+\Delta t)\}_{1\leq n\leq N}$. A fourth order Runge-Kutta method is used to integrate (4).

Now it is immediately noticed that the summation in the right hand side of (4) consumes considerable CPU time when $N$ is large: in order to compute $u_n(t)$ for all $n$, it takes $O(N\times N)$ multiplications if it is directly computed. Greengard and Rokhlin [4] devised an algorithm which computes the the right hand side of (4) for all $n$ with $O(N)$ numbers ( not $O(N^2)$ ) of operations of multiplication/division. Its efficiency and limitation are studied well now (see, e.g., Hamilton and Majda [5] ). Later, other fast summation methods are proposed (see Introduction of [3]). Among them, Draghicescu [2] proposes a fast summation method, which we are going to use in this paper. The reason we chose Draghicescu's method is that the method can be applied to rather wide class of integral kernels. Also, the method seems to be easier to make a structured program. Our objective in this paper is to give a numerical experiment which demonstrate the usefulness of the algorithm.

Although [2] and [3] present convincing experiments, we think that two issues have escaped from the analysis of these papers. One is the periodic boundary condition and the other is the Fourier filter. Our experiments pay attention to these items, too.

Let us now briefly explain Draghicescu's algorithm. This method provides us with a velocity field with $O(N(\log N)^3)$ operations. However, it has a drawback: the velocity fields are computed only approximately. Namely, we can enjoy the fast summation only if we can tolerate the truncation error committed by the algorithm.

Consider a fixed $w_n$. Then $\{w_m\}_{1\leq m\leq N}$ are divided into two subsets: far field and near field. Far field is the set of those points whose distance from $w_n$ are greater than a certain constant. Near field is the set of those points not belonging to far field. We compute

$$\sum_{w_m \in \text{near field}} K_\delta(w_n, w_m)$$

directly. The region of far field is divided into rectangles, which are set up in advance. Then $K_\delta(w_n, w_m)$ for a $w_m$ in far field is approximated by replacing $w_m$ with the center of the rectangle containing $w_m$. Then $K_\delta$ is approximated by its Taylor expansion about the center.

This procedure contains two parameters. One is the order of the Taylor polynomial which we shall write by $\Lambda$. The other is the distance parameter which distinguishes far field and near field.

In order to explain the distance parameter, we briefly recall definitions in [2]. We assume that the number of vortex blobs is a power of 2 and set $n = 2^l$ with $l = 2l'$, where $l'$ is a

positive integer. Let $\mathbf{S}$ be the computational domain. We cover $\mathbf{S}$ with a square grid of size $h = 2^{-l'}$. An $h \times h$ mesh square is called a cell. The set $\sigma$ of all admissible collections is

$$\sigma = \sigma_0 \cup \sigma_1 \cup \cdots \cup \sigma_l,$$

where $\sigma_k$ is a set of pairwise disjoint admissible collections of level $k$ defined recursively as follows: If $k = 0$ at level 0, each collection consists of a single cell.

$$\sigma_0 = \{[(i-1)h, ih] \times [(j-1)h, jh] \; ; \; i, j = 1, \cdots, 2^{l'}\}.$$

If $k \geq 1$, an admissible collection at level $k$ is obtained by joining together two adjacent level $k-1$ admissible collections such that each level $k-1$ collection is included in exactly one level $k$ collection. The joining is along a vertical if $k$ is even and along a horizontal if $k$ is odd. More precisely, for $k = 2k'$:

$$\sigma_k = \{[2^{k'}(i-1)h, 2^{k'}ih] \times [2^{k'}(j-1)h, 2^{k'}jh] \; ; \; i, j = 1, \cdots, 2^{l'-k'}\}$$

and for $k = 2k' - 1$:

$$\sigma_k = \{[2^{k'}(i-1)h, 2^{k'}ih] \times [2^{k'+1}(j-1)h, 2^{k'+1}jh] \; ; \; i = 1, \cdots, 2^{l'-k'}, j = 1, \cdots, 2^{l'-k'-1}\}$$

The only admissible collection at level $l$ is $\mathbf{S}$.

Let $\tau$ be an arbitrary admissible collection. Define the radius of $\tau$ with center $y_\tau$ as

$$\rho(\tau) = \sup\{|y - y_\tau| : y \in \tau\}.$$

For a fixed $x \in \mathbf{S}$ and a positive parameter $\nu$, $F(x)$ is defined as the set of all collections of $\tau$ with center $y_\tau$ such that the following holds

$$\rho(\tau) \leq h^\nu |x - y_\tau|,$$

and $\tau$ is maximal, i.e., it is not strictly included in any other collection satisfying this condition. Then we define far field as $\cup F(x)$. The parameter $\nu > 0$ is determined in order to obtain the smallest possible number of operations while preserving the desired order of accuracy. In our computation below, we set $\nu = 0.245$ in all the experiments. We define a near field $\mathbf{N}(\mathrm{x})$ as $\mathbf{S} \setminus \mathbf{F}(\mathrm{x})$. For more details, see [2].

# 3 Numerical Experiments

Since our objective is to show the effectiveness of Draghicescu's algorithm and our change of variables from $z$ to $w$, our experiments are limited to what are illustrative to numerical analysts.

## 3.1 The property of Draghicescu's fast summation

In order to show quantitatively the effectiveness of Draghicescu's algorithm, we make some definitions. In this section, we define execution time as the time to evaluate the right hand side of (4) at all $w_n$. Evaluation error $e_{vec}$ is defined as

$$e_{vec} = \max_{1 \le n \le N} |u_n^{fast}(t) - u_n^{direct}(t)|,$$

where $u_n^{fast}(t)$ and $u_n^{direct}(t)$ are the velocity vectors at the position of $n$-th vortex blob evaluated by the fast and direct summations, respectively. We define the word "efficiency" as the reduction rate of execution time:

$$\text{efficiency } (\%) = \frac{\text{the execution time by fast summation}}{\text{the execution time by direct summation}} \times 100$$

If efficiency is less than 100%, this fast algorithm is effective. The parameters we can change are the number of vortex blobs($N$), the amplitude of disturbance($\epsilon$), the desingularization parameter($\delta$), and approximation order of Taylor expansion of integral kernel($\Lambda$). $\epsilon$ is fixed to 0.01 in this paper. The following experiments are concerned with (4). They are performed with double precision on Hewlett-Packard workstation J210. We discretize (4) by the fourth order Runge Kutta method and compute the elapsed time needed to go one step forward (i.e., from from $t = 0$ to $t = \Delta t$ ). $\Delta t$ is 0.01. "time" in each table means CPU time by seconds.

1. approximation order ($\Lambda$):
   Table 1 shows the execution time and error $e_{vec}$ with various approximation orders of Taylor expansion of integral kernel. $\Lambda$ is varied from 4 to 14. We make other numerical parameters fixed; $N = 4096$ and $\delta = 0.01$. It is obvious that the higher-order approximation makes error small. However, it takes more time to evaluate the vector field as $\Lambda$ grows. If $\Lambda$ is greater than 11, efficiency exceeds 100.0%.

2. number of vortex blobs ($N$):
   Table 2 shows the execution time and $e_{vec}$ for various numbers of vortex blobs. We execute numerical computation with two cases: $\Lambda = 7$ and $\Lambda = 4$. $\delta$ is fixed to 0.01. If errors of order $10^{-6}$ are tolerated, when $\Lambda = 7, N = 2^{14}$ yields a five times faster computation.

3. desingularization parameter ($\delta$):
   Table 3 shows evaluation errors with varying $\delta$. Two cases are considered:(1) $N = 4096$ and $\Lambda = 8$, (2) $N = 16384$ and $\Lambda = 8$. As $\delta$ increases, error gets smaller.

In order to make accurate computations, we need higher-order approximation. However, since efficiency is reduced when $N$ is small, approximation order is restricted by the number of vortex blobs. The more vortex blobs make accurate and effective computation possible. From these results, we can conclude that this fast summation method is very effective when $N$ and $\delta$ is large. These are already known in [2, 3]. However, our experiments seems to be more quantitative.

| $\Lambda$ | time | efficiency | $e_{vec}$ |
|---|---|---|---|
| 4 | 19 | 35.8% | 2.092626e-04 |
| 5 | 21 | 39.6% | 3.895580e-05 |
| 6 | 24 | 45.2% | 2.046922e-05 |
| 7 | 28 | 52.8% | 3.994516e-06 |
| 8 | 32 | 60.4% | 1.265064e-06 |
| 9 | 37 | 69.8% | 3.038494e-07 |
| 10 | 43 | 81.1% | 1.135958e-07 |
| 11 | 49 | 92.5% | 3.319052e-08 |
| 12 | 55 | 103.8% | 2.562842e-08 |
| 13 | 62 | 117.0% | 6.060725e-09 |
| 14 | 70 | 132.1% | 2.417960e-09 |
| $\infty$ | 53 | 100.0% | |

Table 1: Execution time and evaluation error with various approximation order of Taylor expansion $\Lambda$. $N = 4096$ and $\delta = 0.01$. $\infty$ means CPU time by direct summation. Efficiency is lost, if $\Lambda \geq 12$, .

| $N$ | time($\infty$) | time($\Lambda$=7) | $e_{vec}$($\Lambda$=7) | time($\Lambda$=4) | $e_{vec}$($\Lambda$=4) |
|---|---|---|---|---|---|
| 1024 | 3 | 4(133.3%) | 1.014e-05 | 2(66.7%) | 3.031e-04 |
| 2048 | 13 | 10(76.9%) | 1.189e-05 | 6(46.2%) | 3.703e-04 |
| 4096 | 53 | 28(52.8%) | 3.994e-06 | 19(35.8%) | 2.092e-04 |
| 8192 | 215 | 69(32.1%) | 4.603e-06 | 49(22.8%) | 2.642e-04 |
| 16384 | 886 | 186(20.1%) | 1.701e-06 | 131(14.8%) | 1.493e-04 |

Table 2: Execution time and error when the number of vortex blobs $N$ is varied. $\Lambda = 7$ and 4. $\delta = 0.01$. Time($\infty$) is execution time using direct summation. Large number of vortex blobs enable us to make effective and precise computation.

| $\delta$ | $e_{vec}(N = 4096)$ | $e_{vec}(N = 16384)$ |
|---|---|---|
| 0.01 | 1.265064e-06 | 5.713271e-07 |
| 0.02 | 9.078623e-07 | 3.638098e-07 |
| 0.04 | 3.606127e-07 | 2.612488e-07 |
| 0.06 | 1.663242e-07 | 9.797515e-08 |
| 0.08 | 8.577281e-08 | 3.976212e-08 |
| 0.10 | 4.123106e-09 | 1.414214e-09 |

Table 3: The relation between evaluation error $e_{vec}$ and desingularization parameter $\delta$. Two combinations of parameters; (1) $(\Lambda, N) = (8, 4096)$, and $(\Lambda, N) = (8, 16384)$ are tested.

### 3.2 The effect of changing variable

It is natural to ask if we need to change variables from $z$ to $w$. To show the effectiveness of the change, we apply the fast summation algorithm to the discrete version to the original equation (1), too.

Table 4 is the list of execution time and error with various number of vortex blobs. $\Lambda = 7$ and $\Lambda = 4$ are tested. $\delta$ is fixed to 0.01. Time step size for Runge-Kutta method, $\Delta t$, is 0.01. For small $N$'s, there is no advantage in using fast summation because of loss of efficiency. If we compare Table 4 with Table 2, we see that the efficiency of Draghicescu's algorithm is improved very much by changing variables. One reason for this is as follows: Because of periodic boundary condition, two points which are located at each edge of computational domain ( $\mathrm{Re}[z] = 0$ and $\mathrm{Re}[z] = 1$ must be judged as those in near field. Some points which were included in far field if it were not for the periodic boundary condition, are regarded as those in near field. Consequently, number of points in near field increase, which is the reason why it takes much time to evaluate the velocity field in original equation.

In addition to the reason above, there is another advantage of (2) over (1). In order to approximate the vector field accurately we have to obtain higher order Taylor expansion of the integral kernel. However, it is difficult to get higher order Taylor coefficients of the integral kernel of (1), since it involves complicated combinations of trigonometric functions. Even if we can obtain higher order Taylor coefficients, say, by some computer manipulation, it consumes CPU time because of very many trigonometric function calls. On the other hand, we can easily obtain higher order coefficients of integral kernel of (2). Note that they are rational functions. Therefore, changing variable from $z$ to $w$ makes it possible to approximate the kernel with little pain.

## 4 The motion of transformed vortex sheet

Figure 1 shows the time evolution of transformed vortex sheet by Draghicescu's fast summation algorithm. Efficiency is 69.8%. Numerical parameters are $N = 4096$, $\Lambda = 9$, $\delta = 0.01$,

| $N$ | time($\infty$) | time($\Lambda=7$) | $e_{vec}(\Lambda=7)$ | time($\Lambda=4$) | $e_{vec}(\Lambda=4)$ |
|---|---|---|---|---|---|
| 1024 | 4 | 13(325.0%) | 4.421e-05 | 7(175.0%) | 1.599e-03 |
| 2048 | 19 | 29(152.6%) | 5.005e-05 | 17(89.5%) | 2.204e-03 |
| 4096 | 78 | 88(112.8%) | 1.260e-05 | 52(66.7%) | 9.361e-04 |
| 8192 | 315 | 184(58.4%) | 1.892e-05 | 111(35.2%) | 9.958e-04 |
| 16384 | 1276 | 555(43.5%) | 6.630e-06 | 334(26.2%) | 5.988e-04 |

Table 4: Execution time and evaluation error $e_{vec}$ with various $N$ when we apply Draghicescu's algorithm to the original equation of a vortex sheet with periodic boundary condition. Desingularization parameter $\delta = 0.01$. Time($\infty$) is execution time obtained by direct summation.

and $\Delta t = 0.05$. The equation (4) is solved with the initial condition (5). Fourier filter is not used. While $t$ is small, the sheet is nearly flat. At about $t = 0.8$ , vortex sheet begins to roll-up and a spiral grows afterwards. Figure 2 is the time evolution of transformed vortex sheet by direct summation. When we compare these two figures, no difference is observed. When we reduce $\Lambda$ from 7 to 4, the numerical result is shown in Figure 3. In this case, although its efficiency is 35.8%, spurious spirals appear and complex pattern is formed by vortex sheet at $t = 2.0$. Thus $\Lambda = 4$ is totally unacceptable as a numerical mean.

In order to study the accuracy, we define the position error, $e_{pos}$, as follows.

$$e_{pos} = \max_{1 \le n \le N} |w_n^{fast}(t) - w_n^{direct}(t)|,$$

where $w_n^{fast}(t)$ and $w_n^{direct}(t)$ are the position of $n$-th vortex blob by Draghicescu's fast summation and direct summation, respectively. In Figure 4, we show position error $e_{pos}$ when $\Lambda = 9$ and $\Lambda = 4$. The position error is smaller than 0.0001 at $t = 2.0$ when $\Lambda = 9$. This error seems to be tolerable for some purposes.

We need more vortex blobs to compute accurately for a long time and to make efficiency better. Figure 5 is the long time evolution when $(\Lambda, N) = (10, 8192)$. Other parameters are the same as before. In this case, we can compute up to $t = 4.5$. A spiral gets bigger as time grows. Efficiency is 46.0%.

# 5 Fourier filter

As Krasny [7] notes, the numerical computations for small $\delta$ requires a technique called Fourier filter. Figure 5 shows a reasonable computation, but we must note that this is because, due to the largeness of $\Delta t$ ( 0.05 ), the number of iterations ( 90 ) is rather small, hence the accumulation of evaluation errors is negligible. If $\Delta t$ is small, we observe numerical instability in early stages. Also, if $\delta$ is small, numerical instability becomes obvious for small $t$. In such situations, we have to resort to Fourier filter.

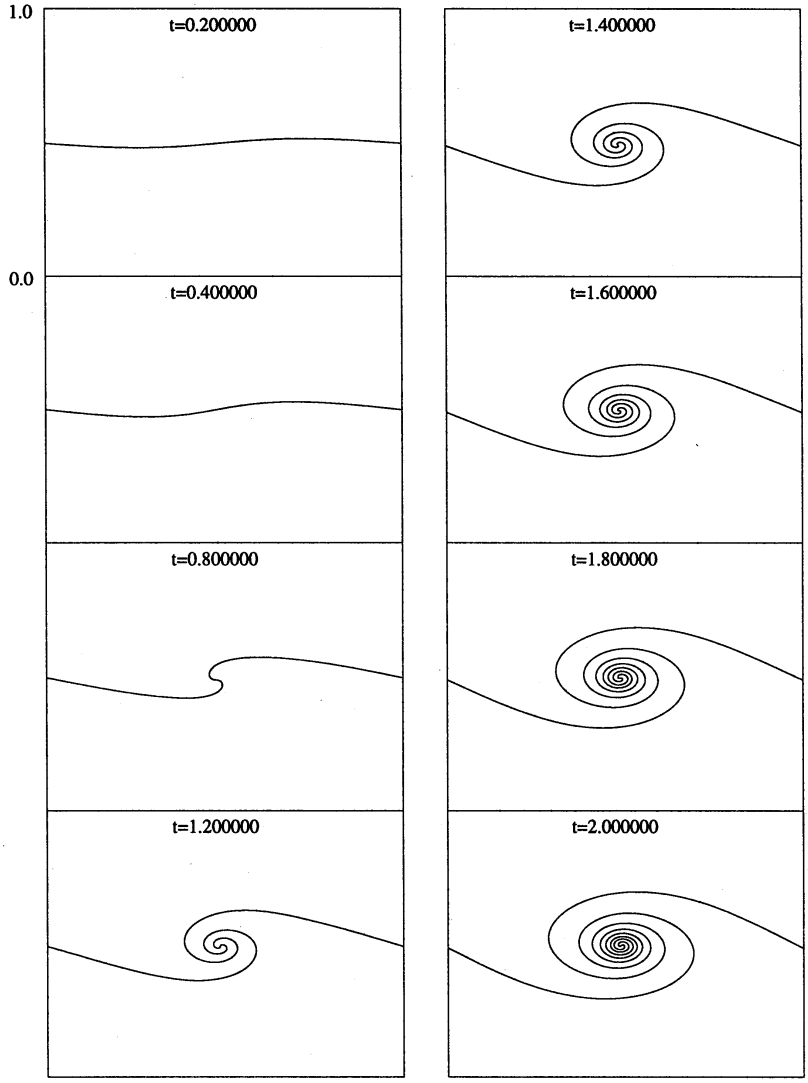


Figure 1: The time evolution of transformed vortex sheet by Draghicescu's fast evaluation. Numerical parameters: $N = 4096$, $\Lambda = 9$, $\delta = 0.01$, and $\Delta t = 0.05$. A spiral is formed.

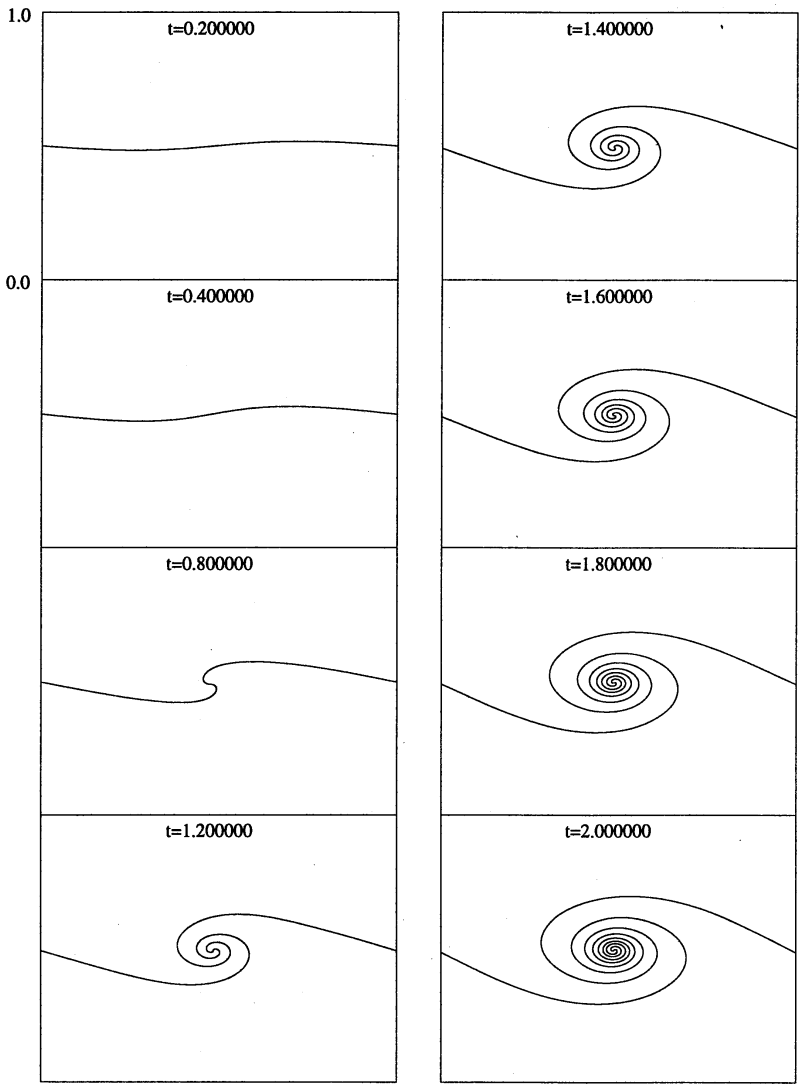


Figure 2: The time evolution of transformed vortex sheet by direct summation. Numerical parameters: $N = 4096$, $\delta = 0.01$, and $\Delta t = 0.05$. A spiral is formed.

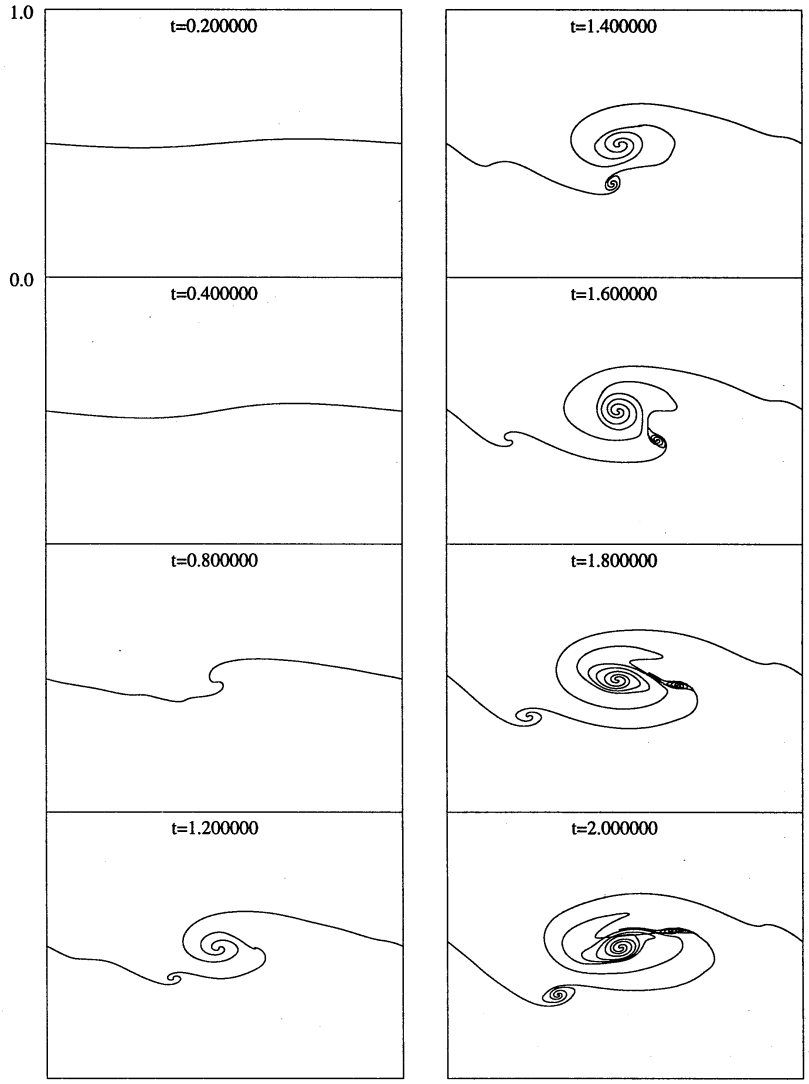


Figure 3: The time evolution of transformed vortex sheet by Draghicescu's fast method. Numerical parameters: $N = 4096$, $\Lambda = 4$, $\delta = 0.01$, and $\Delta t = 0.05$. Because of the loss of precision, the numerical solution is quite different from the directly evaluated solution for $t \geq 0.80$ .

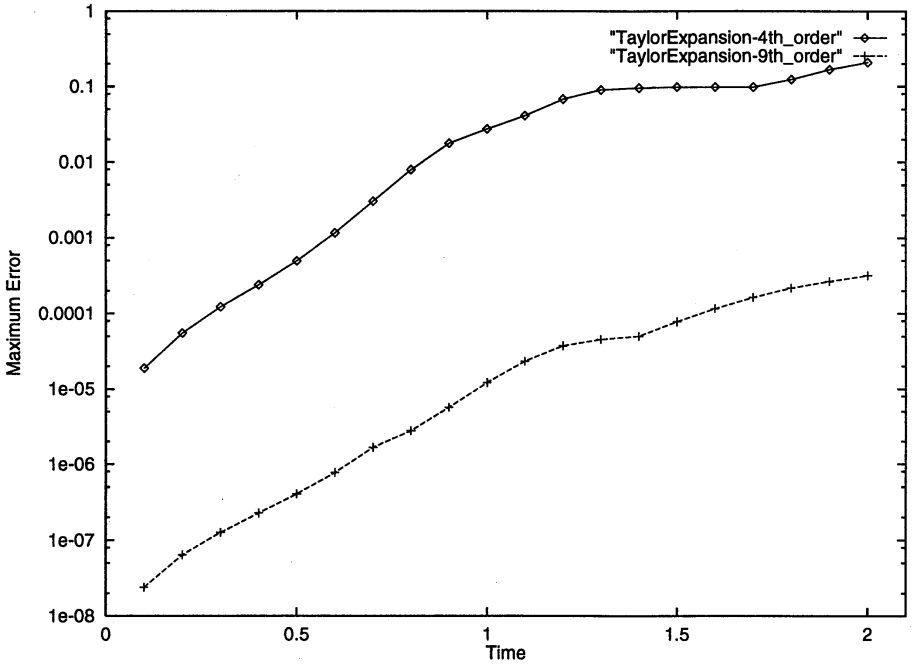


Figure 4: The logarithm plot of evaluation error $e_{pos}$ by fast algorithm with $\Lambda = 9$ and $\Lambda = 4$. Error is amplified as time step goes in both case.

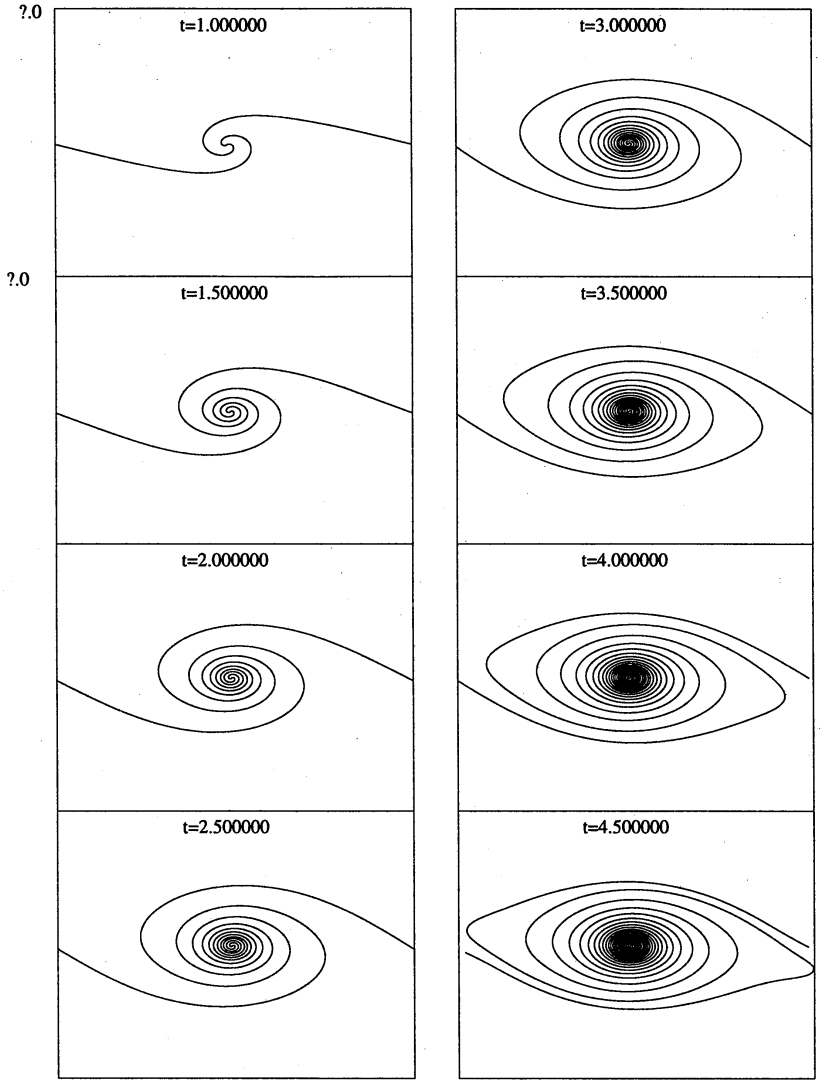


Figure 5: The time evolution of transformed vortex sheet by Draghicescu's fast method. Numerical parameters: $N = 8192$, $\Lambda = 10$, $\delta = 0.01$, and $\Delta t = 0.05$. The increase of vortex blobs make it possible for us to obtain a long time evolution of vortex sheet.

The present section is devoted to the analysis of Draghicescu's method with Fourier filter. Fourier filter works well only when the low frequency modes are dominant and high frequency modes decay sufficiently rapidly. In view of this, we consider the following initial condition:

$$w_n(0) = \exp\left(\frac{2\pi i n}{N}\right) + \epsilon \exp\left(\frac{4\pi i n}{N}\right) \qquad (n = 1, 2, \cdots, N),$$

where $\epsilon = 0.01$ as before.

We first study the effectiveness of the direct summation and the Fourier filter applied to the transformed equation (4). What we want to cut off by the filter is round-off error. Figure 6 shows the numerical solution of the sheet at $t = 1.2$ and $t = 1.4$, with the various threshold value from $10^{-18}$ to $10^{-10}$ and without filtering. Numerical parameters are $N = 8192$, $\Delta t = 0.01$,and $\delta = 0.01$. When Fourier filter is not applied, numerical solution is at $t = 1.2$. When the threshold is less than $10^{-18}$, round-off error can't be cut off by the filter. As a result, round-off error is amplified and makes the numerical solution unstable. Therefore, the threshold must be at least greater than $10^{-18}$. On the other hand, when the threshold is equal to $10^{-10}$, numerical computation becomes unstable at $t = 1.4$. This is because the filter cuts off the genuine increase of the modes. Consequently, to make accurate computation with Fourier filter up to $t = 1.4$, we find that it is desirable to choose the threshold value from $10^{-16}$ to $10^{-12}$.

Next we apply Fourier filter to Draghicescu's fast method. In this case, in addition to round-off error, approximation error must be erased by the filter. However, approximation error is much greater than round-off error. Therefore, unless the threshold value is chosen properly, it is difficult to cut off both errors by the filter. In order to study the valid threshold, we give in Figure 7 logarithmic plots of Fourier coefficients of the numerical solution when we proceed by one time step. Approximation order of each graph from top to bottom in the Figure 7 is (a) $\Lambda = 10$, (b) $\Lambda = 12$, and (c) $\Lambda = \infty$ (direct method), respectively. Other parameters are $N = 8192$, $\Delta t = 0.01$,and $\delta = 0.01$. This figure indicates that if we set the threshold value to $10^{-12}$, the numerical errors of high frequency modes can't be eliminated in the case of $\Lambda = 10$ and $\Lambda = 12$.

Figure 8 is the time evolution of the sheet by Draghicescu's fast method with Fourier filter. Approximation order $\Lambda$ is 12 and the threshold is $10^{-12}$. At $t = 0.6$, numerical computation doesn't work well because of the failure of the filter. In order to cut off the errors surely the threshold must be greater than $10^{-12}$, for example $10^{-10}$, which is found to be unsuitable in the previous observation.

In order to use smaller threshold, approximation error must be reduced. From the results from section 3, there are three possibilities to improve accuracy of approximation;

- larger number of vortex blobs ($N$)
- higher order Taylor expansion ($\Lambda$)
- larger desingularization parameter ($\delta$)

In this paper, since the desingularized parameter $\delta$ is fixed to 0.01, we use larger $N$ and higher order approximation, so that the effect of Fourier filter is compatible with efficiency.

Figure 9 shows the time evolution of the sheet. Numerical parameters are $N = 16384$, $\Delta t = 0.01$, $\delta = 0.01$,and $\Lambda = 15$. The threshold value is $10^{-12}$. Efficiency is 42.3%. In this combination of parameters, Draghicescu's fast method with Fourier filter works well. Therefore, Fourier filter is effective only when approximation error is sufficiently small, in other words, a large number of vortex blobs are needed if we take efficiency into consideration. From the viewpoint of practical application, this restriction is not serious because the necessity of the fast method arises only when we want to use a large number of vortex blobs.

Acknowledgment.
The authors began the present study by the suggestion from Professor R. Krasny. We express our gratitude to Prof. R. Krasny, who kindly introduced to us Draghicescu's algorithm as well as some new references.

# References

[1] R.E. Caflisch and O. F. Orellana, Singular solutions and ill-posedness for the evolution of vortex sheets, SIAM J. Math. Anal., vol. 20(1989), pp. 293–307.

[2] C.I. Draghicescu, An efficient implementation of particle methods for the incompressible Euler equations, SIAM J. Numer. Anal., vol. 31 (1994), pp. 1090–1108.

[3] C. I. Draghicescu and M. Draghicescu, A fast algorithm for vortex blob interactions, J. Comput. Physics, vol. 116 (1995), p. 69-78.

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

[5] J. Hamilton and G. Majda, On the Rokhlin-Greengard method with vortex blobs for problems posed in all space or periodic in one direction, J. Comput. Phys., vol. 121 (1995), pp. 29-50.

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

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

[8] R. Krasny, Computation of vortex sheet roll-up, Springer Lecture Notes in Math., # 1360 (1988), Eds., C. Anderson and C. Greengard, pp. 9–22.

[9] P.G. Saffman, Vortex Dynamics, Cambridge Univ. Press, (1992).

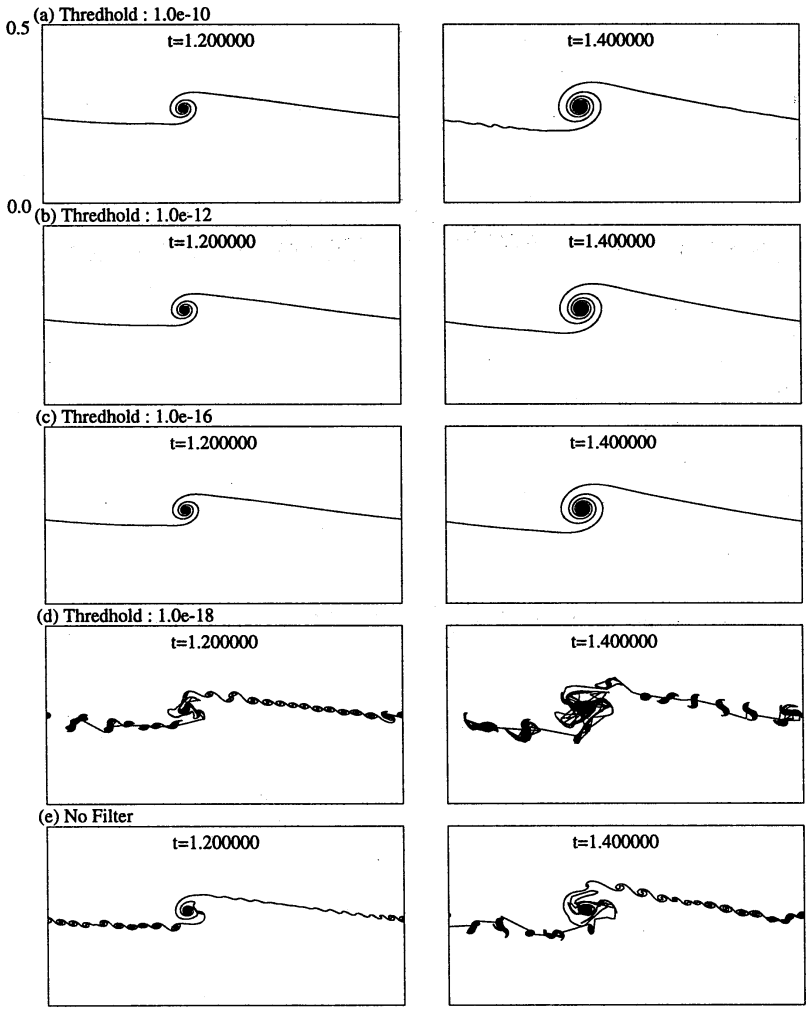


Figure 6: The numerical solution of transformed equation at $t = 1.2$ and $t = 1.4$ calcurated by direct method with Fourier filter. Each row corresponds to the case of threshold value: (a) $10^{-10}$, (b) $10^{-12}$, (c)$10^{-16}$, (d)$10^{-18}$,and (e)No Filter

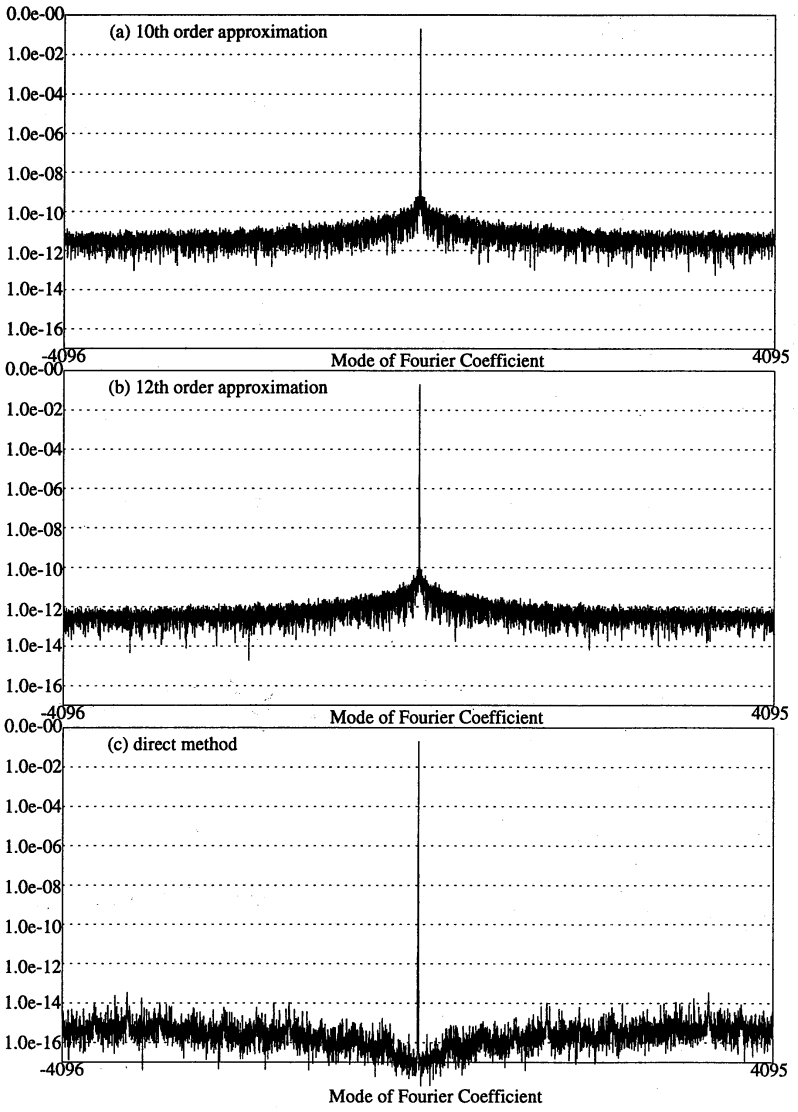


Figure 7: The logarithmic plot of Fourier coefficient of the numerical solution when we proceed one time step. Numerical parameters; $N = 8192$, $\delta = 0.01$,and $\Delta t = 0.01$. Approximation order in (a) and (b) are $\Lambda = 10$ and $\Lambda = 12$, respectively. In (c), we calcurated by direct method.

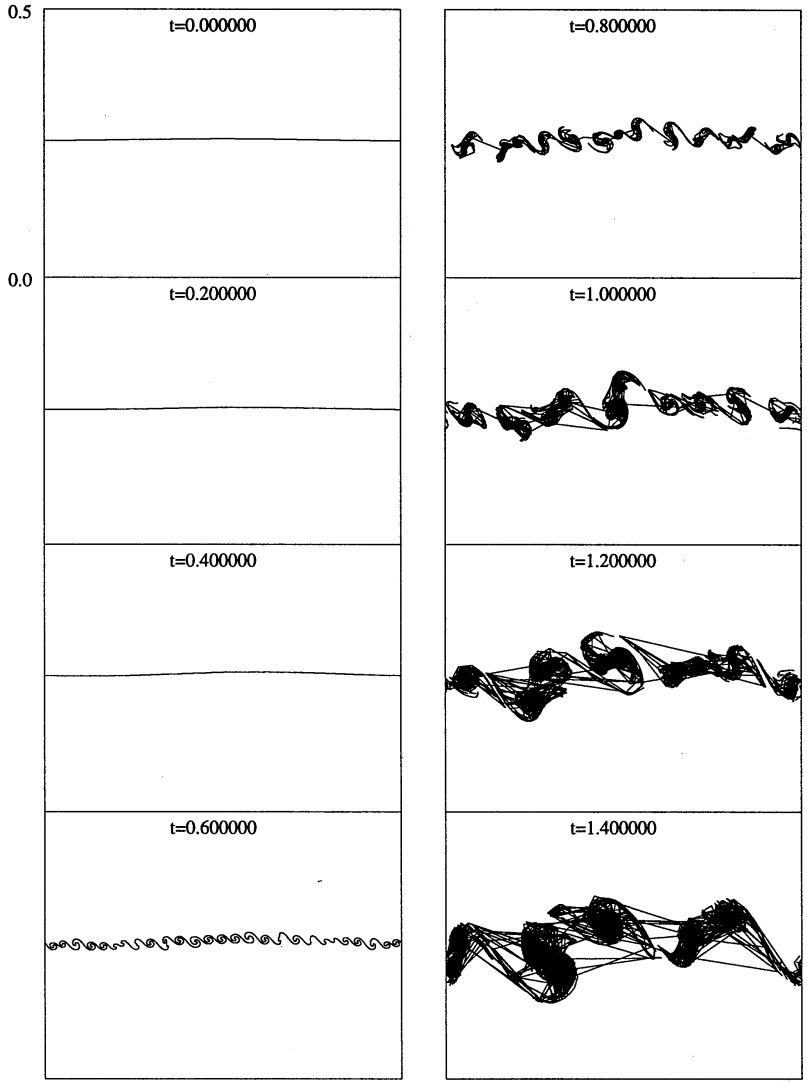


Figure 8: The time evolution of transformed vortex sheet using Draghicescu's fast algorithm with Fourier filter. Approximation order $\Lambda$ is 12 and the threshold value is $10^{-12}$. Numerical errors are not cut off by the filter.

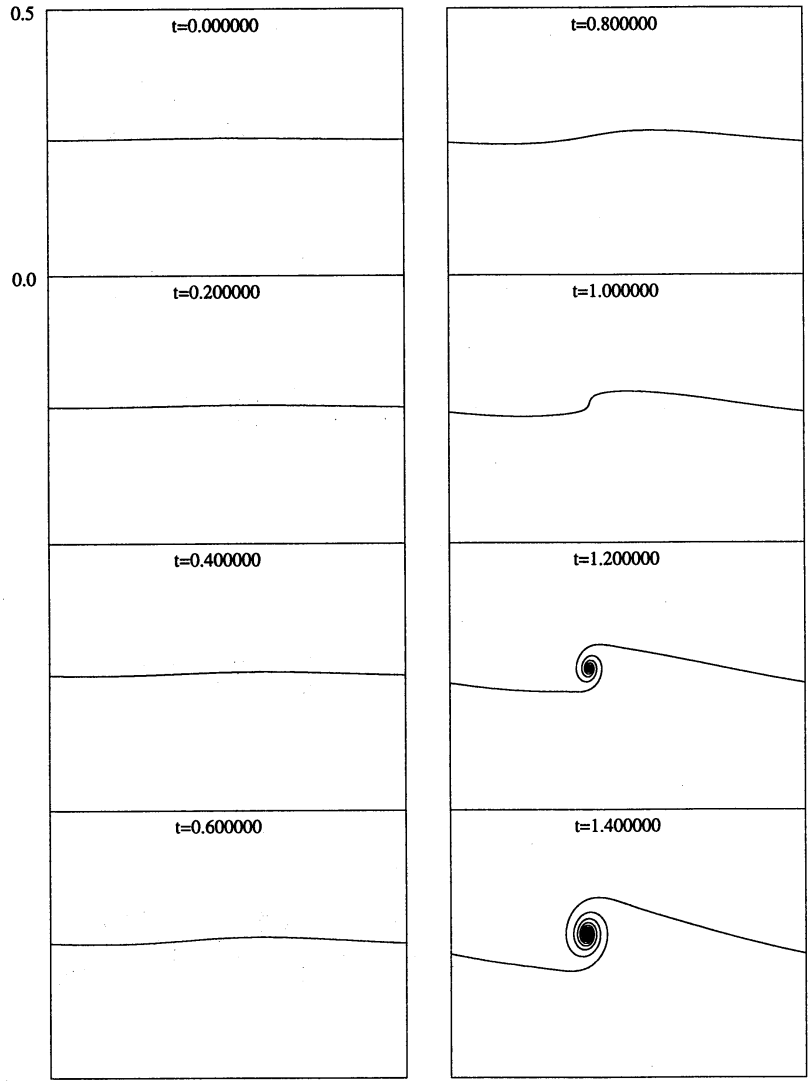


Figure 9: The time evolution of transformed vortex sheet using Draghicescu's fast algorithm with Fourier filter. $N = 16384$. Approximation order $\Lambda$ is 15 and the threshold value is $10^{-12}$.