An efficient front-tracking solver for thermocapillary migration simulations

Zhaohua Yin and Qiaohong Li
National Microgravity Laboratory, Institute of Mechanics, Chinese Academy of Sciences, No.15 Beisihuanxilu, Beijing 100190, P.R. China

1 Email: zhaohua.yin@qq.com

Abstract. An efficient numerical scheme with the front-tracking strategy is presented to investigate the thermocapillary migration of drops. The speed bottleneck of current simulations, especially when the Marangoni number (Ma) is large, is caused by the non-separable elliptic partial differential equations (PDE). In this paper, Fast Fourier Transform is adopted to solve the standard Poisson equation (Tri-FFT), and the non-separable elliptic PDE is solved by the iterative usage of Tri-FFT with high efficiency. In general, the computational cost of the whole system is very low with a hybrid package of Tri-FFT and the Successive Over-Relaxation methods. Finally, the impacts of Ma and Reynolds numbers on thermocapillary migration simulations when Ma>100 are studied.

1. Introduction

The motion of drops is quite common in daily life as well as in industrial production. On the earth, drops are mostly driven by buoyancy. However, in space, buoyancy effects can be ignored, so other kinds of forces should be introduced to drive drops. In particular, interfacial tension normally decreases with increasing temperature. When drops are placed in another kind of liquid (bulk fluid) with temperature gradient, the bulk fluid near the interface is driven toward the cold region, while the counterforce pushes drops to the warm region. The drop migration caused by temperature gradient on the interface is called the thermocapillary migration. Thermocapillary migration is one of the fundamental multi-fluid flows in microgravity fluid dynamics and shows practical potential in many industrial applications, e.g., material production, crystal growth and chemical pharmacy. Because of its theoretical and practical value, thermocapillary migration has attracted much interest and there have been lots of achievements obtained with experimental, theoretical and numerical methods (see the review book [1]). The original thermocapillary work by Young et al. neglected the inertial convection and thermal convection (the so-called YGB Model [2]). The derived steady migration velocity from YGB Model is:

\[ V_{YGB} = \frac{2U}{(2+3\mu_2/\mu_1)(2+k_2/k_1)} \]

where, \( k \) is the thermal conductivity, \( \mu \) the kinematic viscosity and \( U \) the reference velocity defined by the balance of thermocapillary force and viscosity force on the drop: \( U = |\sigma_\tau|/\nabla T_{\infty} |R/\mu_1. \)

Here, \( \sigma_\tau \) is the changing rate of the interfacial tension with temperature, \( R \) the drop radius and \( \nabla T_{\infty} \) the temperature gradient imposed on the bulk liquid. Throughout this paper, symbols with subscript 2
are the parameters of the drop, and those with the subscript 1 are the values of the bulk liquid. Marangoni number (Ma) and Reynolds number (Re) are two of the most important non-dimensional numbers in thermocapillary migration. Their definitions are: \( Ma=UR/\kappa_1, Re=Q_1 UR/\mu_1 \). Here, \( \kappa=k/\rho C_p \) is the thermal diffusivity, \( \rho \) the density and \( C_p \) the specific heat. The remaining four non-dimensional numbers are \( \alpha=\mu_2/\mu_1, \beta=\kappa_2/\kappa_1, \gamma=C_{p2}/C_{p1}, \xi=Q_2/Q_1 \).

For small Ma numbers (e.g. \( Ma<100 \)), different studies both in experiments and numerical simulations reach a good agreement: the final steady migration velocity of the drop \( (U_F) \) decreases with the increasing Ma number. However, when Ma number is large \( (Ma>100) \), the \( U_F \) predictions from asymptotic analysis and numerical simulations disagree with space experiments: the asymptotic analysis [3] and numerical simulations [4] showed that the \( U_F \) increases with the increasing \( Ma \), while space experiments showed that the \( U_F \) decreases with the increasing \( Ma \) [5-7]. This disagreement attracts researchers’ increasing interest in large Ma thermocapillary migration. Since the behavior of large Ma drop migration is very complicated, it takes a long time for drops to reach steady migration velocity, and the cost of the full drop migration simulation for large Ma is huge even in moderate resolution (50 points per Radius) in our previous work [8]. The influence of non-dimensional numbers on drop migration can only be discussed for low Ma or before the drop reaches steady migration velocity. Hence, the speed of numerical simulation is a bottle-neck for further numerical studies of large Ma thermocapillary migration.

In the numerical studies of thermocapillary migration, it is inevitable to deal with the moving-interface problem. The Front-tracking scheme [9-12], the Level-set scheme [13] and the Volume of fluid scheme (VOF) [14] are methods proved to be useful for moving-interface problem. The advantage of the Front-tracking scheme is the more precise determination of the interface position, which is normally 1/100~1/1000 of the mesh size. In the case of the thermocapillary migration, the Front-tracking results are served as the benchmark solution in this field [15]. Since fluid properties are different between drops and bulk liquid, it is inevitable to come across non-separable elliptic partial differential equation (PDE) in numerical simulations. Successive Over-Relaxation (SOR) is the most popular numerical algorithms to solve non-separable elliptic PDE, but it converges very slowly when initial conditions are not optimal. Multi-Grid technique (e.g., see [16]) is proved to have the quickest converging rate, but it is relatively difficult to be programmed and paralleled. Another semi-direct method can be found in [17]. The basic concept of this semi-direct solver for the non-separable elliptic PDE is the iterative application of a standard Poisson equation [18]. So far as we know, there have been no effort to adopt the semi-direct solver in the field of multi-phase flows, and it will be the main focus of this work. In this paper, the Fast Fourier Transform is adopted to solve the standard Poisson equation (the so-called Tri-FFT scheme) [19], and the related semi-direct method will be termed as ITri-FFT.

The paper is arranged as follows: section 2 describes the governing equations for thermocapillary migration; section 3 describes numerical methods such as projection method and ITri-FFT; section 4 shows the performance of ITri-FFT and its combination with other iteration schemes; section 5 discusses the influences of Re and Ma on drop thermocapillary migration when heat convection is strong.

2. Governing equations

In this study, the front-tracking method is adopted in our simulations. Both three dimensional (3D) and axisymmetric models are adopted (figure 1). The drop with radius R is surrounded by the bulk fluid. The direction of the temperature gradient is along the z-axis. In practice, it is convenient to introduce the nondimensional quantities in numerical simulations:
\[ \mathbf{u}^* = \mathbf{u}/U, \quad \mathbf{x}^* = \mathbf{x}/R, \quad t^* = t(U/R), \quad \mathbf{F}_\sigma = \mathbf{F}/(q_1 U^2), \quad T^* = T((\nabla T_{\infty})/R), \quad p^* = p/(q_1 U^2), \quad \phi^* = \phi/q_1, \quad \mu^* = \mu/\mu_1, \quad k^* = k/k_1, \quad C_p^* = C_p/C_p^*. \]

Here, \( \mathbf{u} \) is the velocity vector, \( p \) the pressure, and \( T \) the temperature. \( \mathbf{F}_\sigma \) is the body force term produced by the interfacial tension (see equation (28-35) in reference [9] for details). In 3D model:

![Figure 1](image)

(a) 3D model  
(b) Axisymmetric model

**Figure 1.** The schematic diagram of the 3D and axisymmetric model for Marangoni migration.

\[
\mathbf{F} = \int_B \delta(\mathbf{x} - \mathbf{x}_f) \left( \frac{1}{ReCa} - \frac{1}{Re(T - T'_0)} \right) \kappa \mathbf{n} dA,
\]

while in axisymmetric model:

\[
\mathbf{F} = \int_B \delta(\mathbf{x} - \mathbf{x}_f) \left( \frac{1}{ReCa} - \frac{1}{Re(T - T'_0)} \right) \kappa \mathbf{n} - \frac{1}{Re} \frac{\partial T}{\partial s} \mathbf{T} ds,
\]

where \( \mathbf{x} \) is the space vector, \( \mathbf{x}_f \) the position of the cell \( f \) on the interface \( B \), \( \delta \) a delta function, \( T'_0 \) reference temperature at \( z=0 \), \( \kappa \) the sum of two principal curvatures of the interface, and \( s \) the natural coordinate along the interface. \( \mathbf{n} \) and \( \mathbf{T} \) denote the normal and tangential unit vectors of the interface, respectively. \( Ca = U_1/\sigma_0 \) is Capillary number, where \( \sigma_0 \) is the interfacial tension when temperature is \( T'_0 \). In the following, symbols without stars are adopted to indicate nondimensional values (except in section 3, where \( \mathbf{u}^* \) denotes the provisional velocity). The computing domain is a cube \( (0, x_1) \times (0, y_1) \times (z_0, z_1) \) or a cylinder \( (0, r_1) \times (z_0, z_1) \). The non-dimensional equations for the entire domain can be written as:

\[
\nabla \cdot \mathbf{u} = 0,
\]

\[
\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = \frac{1}{Re} \nabla \cdot (\mu (\nabla \mathbf{u} + \nabla \mathbf{u}^T)) - \nabla p + \mathbf{F}_\sigma,
\]

\[
\rho C_p \left( \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T \right) = \frac{1}{Ma} \nabla \cdot (k \nabla T).
\]

Boundary conditions on the solid walls (figure 1 (a)) for 3D model are:

\[
\mathbf{u}|_{x=0, x=x_1} = \mathbf{u}|_{y=0, y=y_1} = \mathbf{u}|_{z=z_0, z_1} = 0, \quad T|_{x=x_1, y=0} = T|_{y=y_1, 0} = T|_{z=z_0, 1} = 0 + z.
\]

For the axisymmetric model, boundary conditions on the solid walls (figure 1 (b)) are:
\[ u_{r=r_0} = u_{z=0} = 0, \quad T_{r=r_0} = T_{z=0} = T_0 + z, \] (8)

and boundary conditions on the symmetry axis are:
\[ u_{r=0} = 0, \quad \frac{\partial w}{\partial r}_{r=0} = 0, \quad \frac{\partial T}{\partial r}_{r=0} = 0, \] (9)

where \( u \) is velocity in \( r \) direction and \( w \) velocity in \( z \) direction.

Initial conditions for both models are: \( u_{r=0} = 0, \quad T_{r=0} = T_0 + z. \) (10)

3. Numerical methods

3.1. Projection method

The simplest projection scheme is originally proposed by Chorin and Temam \([20, 21]\). With the implicit Euler time stepping, the algorithm is:

\[
\begin{align*}
\frac{q^{n+1}u^{*} - q^n u^n}{\Delta t} &= \frac{1}{Re} \nabla \cdot \left( \mu^n (\nabla u^n + \nabla T u^n) - \nabla \cdot (q^n u^n u^n) + F_{\sigma}^n \right), \\
\frac{q^{n+1}u^{n+1} - q^{n+1}u^*}{\Delta t} &= -\nabla p^{n+1}, \\
\nabla \left( \frac{1}{q^{n+1}} \nabla p^{n+1} \right) &= \nabla \cdot u^*,
\end{align*}
\]

(11) (12) (13)

where \( u^* \) is the provisional velocity and \( n \) the time step. The second-order projection method [22] applied here is:

\[
\begin{align*}
\frac{3q^{n+1}u^{*} - 4q^n u^n + q^{n-1} u^{n-1}}{2\Delta t} &= \frac{1}{Re} \nabla \cdot \left( \mu^n (\nabla u^{*} + \nabla T u^{*}) - \nabla \cdot (q^n u^n u^n) + 2F_{\sigma}^n - F_{\sigma}^{n-1} \right), \\
-2(\nabla \cdot (q^n u^n u^n)) &= \nabla \cdot (q^n u^{n-1} u^{n-1} u^n + 1), \\
\frac{3q^{n+1}u^{n+1} - 3q^{n+1}u^*}{2\Delta t} &= -\nabla \varphi, \\
\nabla \left( \frac{1}{q^{n+1}} \nabla \varphi \right) &= -\frac{3}{2\Delta t} \nabla \cdot u^*,
\end{align*}
\]

(14) (15) (16)

where \( \varphi = p^{n+1} - p^n \).

3.2. ITri-FFT in solving non-separable elliptical PDEs

The difference between second-order projection method and first-order projection method is that the first-order one can get provisional velocity explicitly by equation (11) if flow state is known at \( n \), while the second-order one leads to a relative complicated equation (equation (14)). The provisional velocity equation equation (14) as well as the pressure Poisson equation equation (13) and equation (16) are non-separable PDEs to be solved. In thermocapillary migrating simulations, the scaled velocity is very small and maximum Re numbers involved are below 100. Hence, the equation (14) is always a non-elliptic PDE \([23, 24]\). On the other hand, the density difference in our simulation is never large, so the pressure Poisson equation equation (13) and equation (16) are elliptic PDEs. As will be shown later, the different types of PDEs lead to different efficiencies when different numerical schemes are applied. As we mentioned before, ITri-FFT is the iterative application of a standard Poisson equation solved by Tri-FFT. The 3D Tri-FFT can be easily implemented with available subroutine packages (e.g., the intel MKL library), and the axisymmetric implementation can be found in [8]. Several iterative schemes can be adopted in ITri-FFT to solve pressure Poisson equations equation (16):
\[ \Delta \phi^{m+1} = -q \nabla \left( \frac{1}{q} \nabla \phi^{m+1} - \frac{3q}{2\Delta t} \nabla \cdot \mathbf{u} \right), \]  
(17)

\[ \Delta (\phi^{m+1}) = \phi^{m+1} \Delta x + \frac{3}{2\Delta t} \nabla \cdot \mathbf{u}, \]  
(18)

\[ \Delta (\phi^{m+1} - \phi) = \nabla \cdot (\phi \nabla \left( \frac{1}{q} \right) + \frac{3q}{2\Delta t} \nabla \cdot \mathbf{u}). \]  
(19)

where \( s = (1/q)^{1/2} \) and \( m \) is the iterative step. Among them, equation (17) has been proved to be useful for complex boundary and more efficient than Multi-Grid method [25, 26], and it will be adopted as our iterative scheme for pressure Poisson equation. However, equation (16) can’t be solved by the direct discretized format of equation (17) because the discretized format of equation (16) is not exactly the same as that of equation (17). The difference can be illustrated by the following one-dimensional expression. The central difference scheme for \( \frac{\partial}{\partial x} \left( \frac{1}{q} \frac{\partial \phi}{\partial x} \right) \) is:

\[ D \left[ \frac{\partial}{\partial x} \left( \frac{1}{q} \frac{\partial \phi}{\partial x} \right) \right] = \frac{1}{\Delta x} \left( \frac{\phi_{i+1} - \phi_{i}}{q_{i+1} + q_{i}} \right) + \frac{2}{\Delta x} \frac{\phi_{i} - \phi_{i-1}}{q_{i}}, \]  
(20)

where \( D[A] \) represents the central difference scheme for expression A. On the other hand,

\[ D \left[ \frac{1}{q} \frac{\partial^2 \phi}{\partial x^2} \right] = \frac{1}{\Delta x^2} \left[ \frac{\phi_{i+1} - 2\phi_{i} + \phi_{i-1}}{q_{i}} \right] + \frac{2}{\Delta x} \frac{\phi_{i+1} - \phi_{i-1}}{q_{i}} \frac{\phi_{i+1} - \phi_{i-1}}{q_{i}}. \]  
(21)

The coefficients of \( \phi_{i+1} \), \( \phi_{i} \) and \( \phi_{i-1} \) are:

\[ \phi_{i+1} = \frac{\phi_{i-1}(3q_{i+1}q_{i} - 4q_{i}q_{i+1}^2 + q_{i+1}q_{i} + q_{i} + q_{i+1})}{4q_{i+1}q_{i}^2}, \]  
(22)

\[ \phi_{i} = \frac{2(q_{i+1}q_{i-1}^2 - q_{i})}{q_{i}(q_{i} + q_{i-1})^2}, \]  
(23)

\[ \phi_{i-1} = \frac{2(q_{i+1}q_{i}^2 - q_{i})}{4q_{i+1}q_{i}^2 + q_{i} + q_{i-1}q_{i}}, \]  
(24)

Since the density \( (q) \) is not necessarily uniform in our study [8], equation (22), equation (23) and equation (24) may not be zero throughout the domain zone. Therefore, the discretized format of equation (16) is not the same as that of equation (17). To resolve this difference, a strictly equivalent iterative scheme is adopted:

\[ D[\Delta \phi^{k+1}] = D[-q \nabla \cdot \left( \frac{1}{q} \nabla \phi^{k} \right)] + D[\Delta \phi^{k}] + D\left[ \frac{3q}{2\Delta t} \nabla \cdot \mathbf{u} \right]. \]  
(25)

The provisional velocity equation is a little different from pressure Poisson equation. In order to apply ITri-FFT to equation (14), equation (14) should be rewritten as following:

\[ \frac{3\mathbf{u}^{*}}{2\Delta t} = \frac{1}{Re q_{n+1}} \nabla \cdot (\mu^{n+1} (\nabla \mathbf{u}^{*} + \nabla \mathbf{u}^{*} T)) + \frac{1}{q_{n+1}} (\frac{4q^{n} u^{n-1} u^{n-1} - q^{n-1} u^{n-2} + q^{n} u^{n-1} u^{n-1}}{2\Delta t} \nabla p^{n}) \]

\[ - \frac{1}{q^{n+1}} (2 \nabla \cdot (q^{n} u^{n} u^{n}) - \nabla \cdot (q^{n} u^{n-1} u^{n-1})) + \frac{1}{q^{n+1}} (2F_{e}^{n} - F_{e}^{n-1}). \]  
(26)

And the iterative scheme for equation (26) is:
\[
\frac{3(u^*)^{k+1}}{2\Delta t}C_S \Delta (u^*)^{k+1} = -C_S \Delta (u^*)^k + \frac{1}{Re_{n+1}} \nabla \left( \mu_n (\nabla (u^*)^k + \nabla (u^*)^k) \right) \\
+ \frac{1}{q_{n+1}} \left( \frac{4q^n u^n - q_{n-1} u^{n-1}}{2\Delta t} \nabla p^n \right) \\
- \frac{1}{q_{n+1}} \nabla \left( q^n u^n - q^n u^{n-1} \right) \nabla \left( q^{n-1} u^{n-1} - q^n u^{n-1} \right) + \frac{1}{q_{n+1}} (2F^n - F^{n-1}),
\]

where \( C_s \) is a stable parameter, and its range should be \( 1.0-1.5 \) according to our numerical experiments.

As comparison, the traditional SOR scheme is also employed to solve equation (13), equation (15) and equation (16). All axisymmetric model codes here are written in CUDA (Compute Unified Device Architecture) to make use of the fast and economical GPU (Graphics Processing Unit) accelerator. The GPU accelerator we employ in this work is Tesla C2075, and the parallel-type SOR (i.m. Red-Black SOR) instead of the traditional SOR is implemented here.

4. The performance of ITri-FFT

In this section, we will discuss the efficiency of the ITri-FFT for different models. The physical parameters in this section are set according to those in space experiments \cite{1}: \( \alpha=0.14, \beta=0.47, \xi=1.89, \gamma=0.69, Ma=175, Re=2.1, \) and \( Ca=0.04. \) The convergence restriction for the solution of the pressure is \( |p^{k+1} - p^k|_{\infty} < 10^{-6} \), and those restrictions of the provisional velocities are defined similarly.

| Table 1. Time steps for different resolutions: \( N_z \) is the number of grid in \( z \)-direction, and \( N_r \) the grid number in \( r \)-direction. |
| Resolution \( (N_z \times N_r) \) | 100x400 | 200x800 | 400x1600 | 800x3200 |
| Time step | 0.006 | 0.004 | 0.0015 | 0.0005 |

| Table 2. Time-consumption for one iterative step of ITri-FFT and SOR. |
| Resolution | 100x400 | 200x800 | 400x1600 | 800x3200 |
| Iterative method | SOR | ITri-FFT | SOR | ITri-FFT | SOR | ITri-FFT |
| Time \( (10^{-3} \) second) | 0.144 | 0.981 | 0.365 | 3.15 | 1.16 | 11.8 | 4.37 | 38.8 |

4.1. ITri-FFT in solving pressure equation

4.1.1. The axisymmetric model. Throughout the paper, the uniformed grids are adopted in our simulations. For the axisymmetric model, the domain size is chosen to be \( 4R \times 16R \), and time steps for different resolutions are shown in table 1. Table 2 shows the time-consumption for one iterative step when equation (16) is solved. Since each ITri-FFT iterative step consists of \( 2N_z \) cosine/sine transforms and \( N_z \) triangle equation solvers, one ITri-FFT iterative step takes more time than one SOR iterative step. Fortunately, as shown in table 3, ITri-FFT takes much less iterative steps to converge than SOR, and this leads to high efficiency of ITri-FFT in solving pressure Poisson equation. As resolution increases, ITri-FFT becomes more efficient than SOR. When the resolution is \( 800 \times 3200 \), the simulation with ITri-FFT is 9.98 times faster than that with SOR. The main reason is that the total iterative step for SOR to converge increases rapidly with the increasing resolution while that of ITri-FFT rises slightly. Moreover, ITri-FFT becomes more efficient than SOR when the initial guess is not optimal. Table 4 shows the iterative steps and time-consumption for the first time step. As shown in table 4, it takes much more iterating steps for SOR to converge than ITri-FFT in the first time step.
because pressure field is set to be uniform at the beginning of our simulation. This bad guess leads to increasing iterative steps for SOR but brings little influence to ITri-FFT.

Table 3. Number of iteration and time-consumption for ITri-FFT and SOR in solving pressure equation.

| Resolution | Iterative method | 100x400 | 200x800 | 400x1600 | 800x3200 |
|------------|------------------|---------|---------|----------|---------|
|            | SOR              | N_{10000} | T_{10000} | N_{10000} | T_{10000} | N_{10000} | T_{10000} |
|            |                  | 500-700  | 18.75   | 900-1300 | 87.33   | 1300-2300 | 385.17   | 1800-2400 | 1756.6   |
|            | ITri-FFT         | 8-9     | 4.34    | 8-9      | 12.5    | 8-9      | 40.6     | 9-10      | 175.92   |
| Speed-up   |                  | 4.32    | 8.9     | 6.99     | 9.49    | 9.98     |

1. N_{10000} is the number of iteration for each time step to converge in the first 10000 time steps.  
2. T_{10000} is the whole time consumed for the first 10000 time steps (Unit: minute).

Table 4. Number of iteration and time-consumption for the first time step.

| Resolution | Iterative method | 100x400 | 200x800 | 400x1600 | 800x3200 |
|------------|------------------|---------|---------|----------|---------|
|            | SOR              | N_{1}   | T_{1}   | N_{1}    | T_{1}   | N_{1}    | T_{1}    |
|            |                  | 26667   | 3.84    | 48426    | 17.70   | 72044    | 83.89    | 109904   | 480.44   |
|            | ITri-FFT         | 12      | 0.0117  | 12       | 0.0378  | 13       | 0.154    | 14       | 0.543    |
| Speed-up   |                  | 326.73  | 467.99  | 542.99   | 884.24  |

1. N_{1} is the number of iteration for the first time step to converge.  
2. T_{1} is the whole time consumed for the first time step (Unit: second).

4.1.2. The 3D model. The advantage of solving pressure Poisson equation by ITri-FFT can also be observed in 3D simulation. Table 5 shows the number of iterations and time-consumption for ITri-FFT and SOR in solving 3D pressure Poisson equation. The domain is 6R×6R×24R with resolution of 61×61×241, and time step is 0.004. As shown in table 5, ITri-FFT is about 2.43 times faster than SOR for the first 10000 time-steps, and almost 56 times faster for the first time step when the initial guess for pressure is not optimal. In general, we can conclude that (a) ITri-FFT converges faster than SOR in solving pressure equation, (b) ITri-FFT can converge quickly even when initial conditions are not optimal, and (c) the number of iteration for ITri-FFT to converge increases only slightly when resolution rises. These advantages make ITri-FFT more efficient than SOR in solving pressure equation. From our numerical experiment, ITri-FFT is still more efficient than SOR when the density ratios are between 0.05 and 20. This range of density ratios is large enough to simulate different kinds of drops in the space experiments.

4.2. ITri-FFT in solving provisional velocity equations

Table 6 shows the iterative times to converge and time-consumption for ITri-FFT and SOR in solving provisional velocity. As shown in table 6, the simulation with SOR is much faster than that with ITri-FFT in solving provisional velocity. That is because ITri-FFT needs more iterative steps to converge than SOR when solving provisional velocity. In the following, we give a simple explanation on different convergent behaviors in this section. The governing equation for pressure belongs to the elliptic PDEs, and the pressure perturbation in one place can influence the whole domain because of the assumption of incompressibility. At the same time, ITri-FFT is also a global method, and can easily obtain high efficiency in solving the pressure equation. On the other hand, the momentum equations are not elliptic PDEs, and the perturbation in provisional velocity equation can only influence small region on certain direction. SOR is a local method, and its performance depends largely on initial guess. The velocity field from the last time step is an optimal guess, and can
converge quickly with SOR. In another word, SOR is a better iterative method in solving provisional velocity than ITri-FFT.

| Iterative method | The first time step | The first 10000 time steps |
|------------------|---------------------|----------------------------|
|                  | N₁      | T₁ (second) | N₁₀₀₀₀  | T₁₀₀₀₀ (minute) |
| SOR              | 17015   | 258.56     | 200-300 | 1339.88          |
| ITri-FFT         | 13      | 4.65       | 8-9     | 550.7            |
| Speed-up         | 55.62   | 2.43       |

Table 5. Number of iteration and time-consumption for ITri-FFT and SOR in solving pressure equation for 3D simulation.

Table 6. Number of iteration and time-consumption for ITri-FFT and SOR in solving provisional velocity (Time Unit: minute).

Table 7. Performance of different iterative-method packages.

| Resolution | 100x400 | 200x800 | 400x1600 | 800x3200 |
|------------|---------|---------|----------|----------|
| Iterative method | N₁₀₀₀₀  | T₁₀₀₀₀ | N₁₀₀₀₀  | T₁₀₀₀₀  |
| SOR        | 15-17   | 4.34    | 18-19    | 12.5     | 18-23   | 40.6     | 18-24    | 175.92   |
| ITri-FFT   | 30-40   | 19.35   | 53-68    | 95.22    | 54-80   | 377.13   | 56-87    | 1730.4   |
| Speed-up   | 4.46    | 7.61    | 9.29     | 9.84     |

To sum up, it is clear that the most efficient option is to solve the provisional velocity equation by SOR, and the pressure equation by ITri-FFT. The performances of different strategies are shown in table 7. The advantage of the hybrid method over others increases as the resolution rises. In fact, the Tri-FFT can also be used in updating the Material Properties (see, e.g., equation (27) in [9]). There is no need for iterative usage of Tri-FFT here, so it is very efficient in practice.

5. The influence of nondimensional numbers in thermocapillary migration

All simulations in this section are computed in a domain of 4R×16R with resolution of 400×1600. With the help of this hybrid iterative method, it is now possible to investigate the influence of nondimensional numbers on drop migration for large Ma.

5.1. The influence of Re for large Ma

The variation of the scaled drop migration velocity for different values of Re is shown in figure 2. It is clear that the scaled migration velocity increases slightly with growing Re when Ma is large. This is the same as that when Ma is small [15]. Although Re influences the steady migration velocity slightly, it plays an important role in the evolution of drop migration. We also study the time evolutions of drop
migration velocity for different values of Re with fixed values of Ma=100, 200 and 300. When Re is smaller, the initial migration velocity increases with a higher speed, resulting in a sharper overshot. When Re is large enough, such as Re=100 Ma=100, the overshot disappears. Though small Re leads to large overshoot in drop velocities at the beginning of migration, the overshoot lasts for a shorter time than large Re. Thus when Re is smaller, the drop reaches steady migration velocity faster.

![Figure 2. The Scaled drop migration velocity vs Re, Ma=100, 200 and 300, α=β=γ=ξ=1.0.](image)

5.2. The influence of large Ma
We also studies the time evolutions of drop migration velocity for different values of Ma with fixed values of Re=10, 40 and 100. When Re is small, the drop evolutions are similar regardless of different values of Ma because they all experience an increasing-decreasing-increasing process before reaching the steady migration velocity. This overshoot in drop velocity lasts longer when Ma is larger. The variation of the scaled drop migration velocity for different values of Ma is shown in figure 3. It is clear that the scaled migration velocity decreases roughly with increasing Ma.

![Figure 3. The Scaled drop migration velocity vs Ma, Re=10, 40 and 100, α=β=γ=ξ=1.0 (see the low Ma results in figure 19 of reference [15]).](image)

6. Conclusions
The ITri-FFT scheme shows high efficiency in solving pressure equation, but it is slightly slower than the SOR scheme when the provisional velocity equation is solved. The hybrid iterative method with ITri-FFT to solve pressure and with SOR to solve provisional velocity performs better than the pure method either with ITri-FFT or with SOR. This hybrid iterative method can increase the performance of the Front-tracking code when the thermocapillary migrating problem is solved. The drop migration in a longer simulating time and larger range of fluid property can be accomplished within reasonable short time. It shows that small Re leads to large oscillation but short oscillation period at the beginning.
of migration, and finally to small steady migration velocity. Large Ma leads to long oscillation and small steady migration velocity.

Acknowledgments
This project is supported by the NSF of China (Contract No. G11472283).

References
[1] Subramanian R and Balasubramaniam R 2001 The Motion of Bubbles and Drops in Reduced Gravity (Cambridge University Press, Cambridge)
[2] Young N, Goldstein J and Block M 1959 Journal of Fluid Mechanics 11 350
[3] Balasubramaniam R and Subramanian R 2000 Physics of Fluids 12 733
[4] Ma X, Balasubramaniam R and Subramanian R 1999 Numerical Heat Transfer: Part A Applications 35(3) 291
[5] Balasubramaniam R, Lacy C, Woniak G and Subramanian R 1996 Physics of Fluids 8 872
[6] Hadland P, Balasubramaniam R, Wozniak G and Subramanian R 1999 Experiments in Fluids 26 240
[7] Xie J, Lin H, Zhang P, Liu F and Hu W 2005 Journal of Colloid and Interface Science 285 737
[8] Yin Z, Chang L, Hu W, Li Q and Wang H 2012 Physics of Fluids 24 092101-092101-18
[9] Tryggvason G, Bunner B, Esmaeili A, Juric D, Al-Rawahi N, Tauber W, Han J, Nas S and Y 2001 Journal of Computational Physics 169 708
[10] Glimm J, Grove J, Li X, Shyue K, Zhang Q and Zeng Y 1998 SIAM Journal on Scientific Computing 19 703
[11] Glimm J, Grove J, Li X and Tan D 2000 SIAM Journal on Scientific Computing 21 2240
[12] Hua J and Lou J 2007 Journal of Computational Physics 222 769
[13] Osher S and Sethian J 1988 Journal of Computational Physics 79 12
[14] Hirt C and Nichols B 1981 Journal of Computational Physics 39 201
[15] Yin Z, Gao P, Hu W and Chang L 2008 Physics of Fluids 20 082101
[16] Adams J and Smolarkiewicz P 2001 Applied mathematics and computation 121 301
[17] P Concus and G H Golub 1973 Siam Journal on Numerical Analysis 10 1103
[18] Wang H, Jiang W and Wang Y 2009 Computer Physics Communications 180 1305
[19] Hockney R 1965 Journal of the ACM 12 95
[20] Chorin A 1967 Journal of Computational Physics 2 12
[21] Temam R 1969 Archive for Rational Mechanics and Analysis 33 377
[22] Guermond J, Minev P and Shen J 2006 Computer Methods in Applied Mechanics and Engineering 195 6011
[23] Fletcher C 1991 Computational Techniques for Fluid Dynamics (Springer, Berlin) 2
[24] Wang J, Liu Y, Li T and Ye Z 2005 AIAA/ASME/SAE/ASEE Joint Propulsion Conference Tucson, AIAA 41 2005-4213
[25] Pares-Sierra A and Vallis G 1989 Journal of Computational Physics 82 398
[26] Zhao S and Yedlin M 1994 Journal of Computational Physics 113 215