Chapter 4. MAC Scheme

§4.1. MAC Scheme and the Staggered Grid.

The MAC scheme is a numerical method used to solve the incompressible Navier-Stokes equation in the velocity-pressure formulation:

$$\begin{cases} \partial_t \boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p = \nu\Delta\boldsymbol{u}\,, \\ \nabla\cdot\boldsymbol{u} = 0 \end{cases} \tag{4.1.1}$$

where $\boldsymbol{u} = (u, v)$. A special feature of the MAC scheme is the use of a staggered grid. One such grid is displayed in Figure 3. The pressure variable p is defined at the "$*$" points, and the first and second components of the velocity, u and v, are defined at the "$\triangle$" and the "$\bigcirc$" points, respectively. Define

$$\tilde{D}_x u(x,y) = \frac{u(x+\Delta x, y) - u(x-\Delta x, y)}{2\Delta x}\,, \qquad D_x u(x,y) = \frac{u(x+\Delta x/2, y) - u(x-\Delta x/2, y)}{\Delta x}\,,$$

$$\tilde{E}_x u(x,y) = \frac{u(x+\Delta x, y) + u(x-\Delta x, y)}{2}\,, \qquad E_x u(x,y) = \frac{u(x+\Delta x/2, y) + u(x-\Delta x/2, y)}{2}\,,$$

and similarly for $\tilde{D}_y u$, $\tilde{E}_y u$, $D_y u$, $E_y u$. With this notation we have

$$\Delta_h u = (D_x^2 + D_y^2) u\,.$$

We can then write the MAC scheme as follows:

$$\begin{cases} \dfrac{du}{dt} + u\,\tilde{D}_x u + E_x E_y v\,\tilde{D}_y u + D_x p = \nu\Delta_h u\,, \qquad \text{at "}\triangle\text{" points,} \\[2ex] \dfrac{dv}{dt} + E_x E_y u\,\tilde{D}_x v + v\tilde{D}_y v + D_y p = \nu\Delta_h v\,, \qquad \text{at "}\bigcirc\text{" points,} \\[2ex] D_x u + D_y v = 0\,, \qquad \text{at "}*\text{" points.} \end{cases} \tag{4.1.2}$$

The simplest way of treating the boundary is to use the reflection technique. On the segment of Γ_x (see Figure 3), the boundary condition $v = 0$ is imposed exactly at the "$\bigcirc$" points: $v_{i-1/2,0} = 0$. The boundary condition $u = 0$ is imposed approximately at the "$\bullet$" points by letting

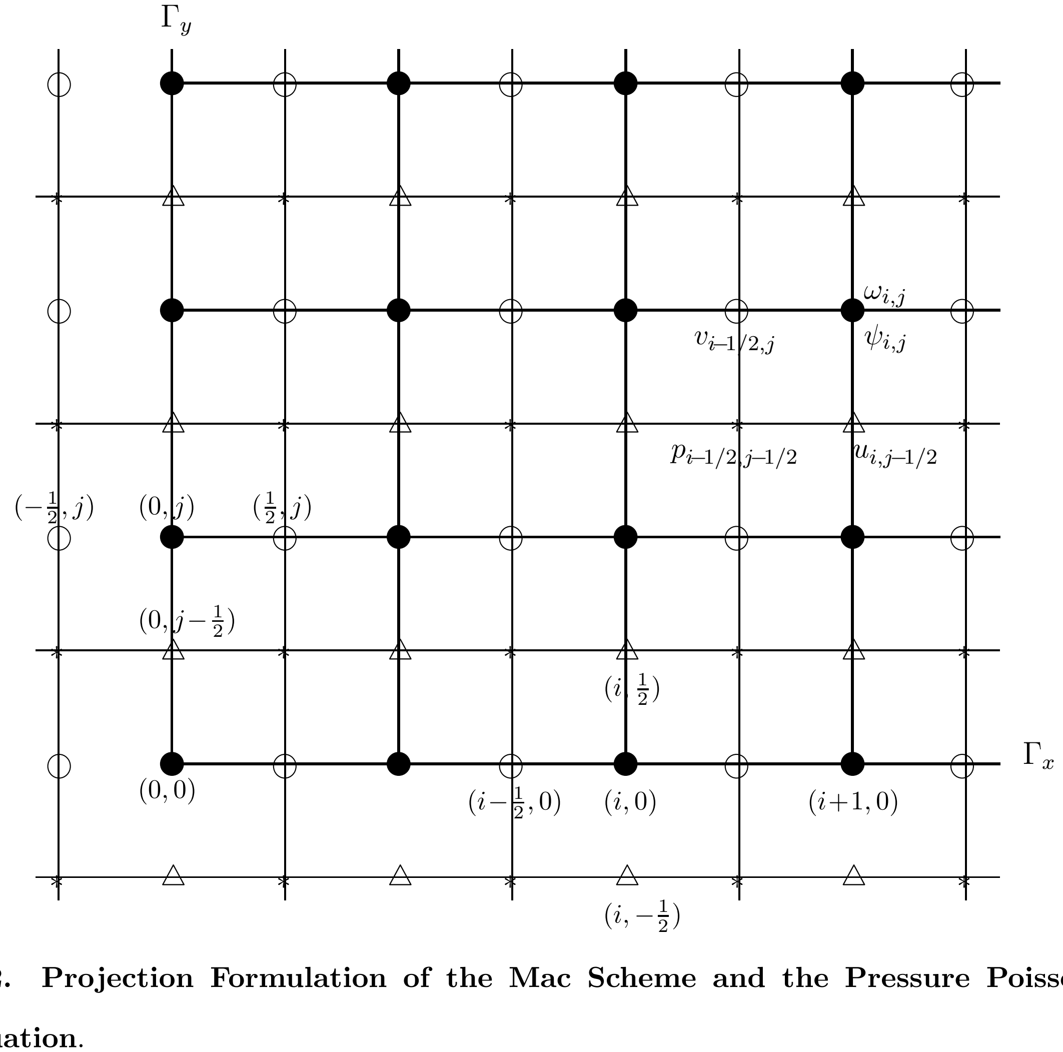
$$u_{i,-1/2} = -u_{i,1/2}. \tag{4.1.3}$$

Similarly for Γ_y, we have

$$v_{-1/2,j} = -v_{1/2,j}\,, \qquad u_{0,j-1/2} = 0.$$

The fully discrete MAC scheme,i.e. including the time discretization, with the viscous term treated explicitly,is given by:

$$(4.1.4)\quad \begin{cases} \dfrac{u^{n+1}-u^n}{\Delta t} + u^n\,\tilde{D}_x u^n + E_xE_y v^n\,\tilde{D}_y u^n + D_x p^n = \nu\Delta_h u^n\,, & \text{at "}\triangle\text{" points,} \\ \dfrac{v^{n+1}-v^n}{\Delta t} + E_xE_y u^n\,\tilde{D}_x v^n + v^n\tilde{D}_y v^n + D_y p^n = \nu\Delta_h v^n\,, & \text{at "}\bigcirc\text{" points,} \\ D_x u^{n+1} + D_y v^{n+1} = 0\,, \qquad \text{at "}*\text{" points.} & \end{cases}$$



§4.2. Projection Formulation of the Mac Scheme and the Pressure Poisson Equation.

In actual computations, we decompose the MAC scheme into 2 steps. First we introduce two intermediate values (u^*, v^*),

$$
\text{(4.2.1)} \qquad \begin{cases} \dfrac{u^* - u^n}{\Delta t} + u^n\,\tilde{D}_x u^n + E_x E_y v^n\,\tilde{D}_y u^n = \nu \Delta_h u^n\,, & \text{at "}\triangle\text{" points,} \\[2ex] \dfrac{v^* - v^n}{\Delta t} + E_x E_y u^n\,\tilde{D}_x v^n + v^n \tilde{D}_y v^n = \nu \Delta_h v^n\,, & \text{at "}\bigcirc\text{" points,} \end{cases}
$$

subtract (4.2.1) form (4.1.4) to get

$$
\text{(4.2.2)} \qquad \begin{cases} \dfrac{u^{n+1} - u^*}{\Delta t} + D_x p^n = 0 & \text{at "}\triangle\text{" points,} \\[2ex] \dfrac{v^{n+1} - v^*}{\Delta t} + D_y p^n = 0 & \text{at "}\bigcirc\text{" points,} \\[2ex] D_x u^{n+1} + D_y v^{n+1} = 0\,, & \text{at "}*\text{" points.} \end{cases}
$$

We can then rewrite (4.2.2) in a projection form:

$$
\text{(4.2.3)} \qquad \begin{cases} u^* = u^{n+1} + \Delta t D_x p^n & \text{at "}\triangle\text{" points,} \\ v^* = v^{n+1} + \Delta t D_y p^n & \text{at "}\bigcirc\text{" points,} \\ D_x u^{n+1} + D_y v^{n+1} = 0\,, & \text{at "}*\text{" points.} \end{cases}
$$

This formulation motivated Chorin to develop the projection method in late 60's. We will discuss the projection method in the next Chapter.

Now, back to the MAC scheme. In the first step, we can directly evaluate (u^*, v^*) at all the interior grid points using (4.2.1).

To recover p^n, we need first to derive the pressure Poisson equation using (4.2.2). Apply D_x to the first equation of (4.2.2) and D_y to the the second equation of (4.2.2), and then add them together. This will give

$$
\text{(4.2.4)} \qquad \Delta_h p^n = -\frac{1}{\Delta t}(D_x u^* + D_y v^*)
$$

In the above we have also used the third equation of (4.2.2).

At the $(1/2, j)$ points , we have

$$
\text{(4.2.5)} \qquad \delta_h p^n_{1/2,j} = -\frac{1}{\Delta t}\left(\frac{u^*_{1,j} - u^*_{0,j}}{h} + \frac{v^*_{1/2,j+1/2} - u^*_{1/2,j-1/2}}{h} \right)
$$

To evaluate (4.2.5) we need the value of $u^*_{0,j}$ at the $(0, j)$ grid point. However, as we will see below, this unknown quantity can be arbitrarily prescribed.

From the first equation of (4.2.2) and the no-slip boundary condition for u^{n+1}, we have along the boundary

$$(D_x p^n)_{0,j} = \frac{u^*_{0,j}}{\Delta t} \tag{4.2.6}$$

But moving this boundary condition (4.2.6) to the right hand side of (4.2.5),

$$\delta_h p^n_{1/2,j} \to \delta_h p^n_{1/2,j} - \frac{2}{h}(D_x p^n)_{0,j}$$

or

$$\delta_h p^n_{1/2,j} \to -\frac{1}{\Delta t}\left(\frac{u^*_{1,j} - u^*_{0,j}}{h} + \frac{v^*_{1/2,j+1/2} - u^*_{1/2,j-1/2}}{h}\right) - \frac{2}{h}\frac{u^*_{0,j}}{\Delta t} \tag{4.2.7}$$

we see that the arbitrary constant that we prescribed above, cancels. Since this is the case, it is easiest to just take

$$u^*_{0,j} = 0 \tag{4.2.8}$$

and the boundary condition for p becomes

$$(D_x p^n)_{0,j} = 0. \tag{4.2.9}$$

This is agree with the fact that there is no boundary condition for p in NSE.

§4.2. Proper Time-Stepping for the Mac scheme.

The original MAC scheme used Forward-Euler for the time discretization. Later Forward-Euler was replaced with the Crank-Nicholson scheme. However, there is a severe cell Reynolds number constraint associated with the convection term. This was discussed in detail in §2.1.

The way of over coming these difficulties is by using a 3rd order, or higher, RK method for the time discretization to eliminate the cell Reynolds number constraint. For simple geometries this indeed result in a quite efficient 2nd order method.

Chapter 2. Basic Numerics

In this chapter we will discuss some basic numerical issues in the computation of incompressible flow. These include such topics as stability, consistency, boundary conditions, efficient time stepping, etc. We will focus on each of these issues individually, identifying the main difficulties associated with each, which will be illustrated through the use of simple model problems. This approach will result in a thorough and clean exposition of the derivation, analysis, and implementation of numerical methods for the given model problem. Once this is accomplished, hopefully more realistic problems can be tackled by systematically applying the lessons we have learned. This will be exactly our approach in later chapters when we examine the current state of the art methods for computing incompressible flows. Also provided in each subsection is a set of exercises which are meant to illustrate more technical aspects of the material, along with programming problems for one to numerically examine the individual issues raised.

§2.1. Time Marching and Numerical Methods for ODE

The time marching in the fluids computations follows the simple techniques developed for the following ODE:

$$u' = f(u) \tag{2.1.1}$$

More precisely, we shall explain that the issues raised in the development of numerical methods to approximate the solutions of ODEs actually are rooted in numerical approximations of PDEs which describe some basic physical phenomena such as convection, diffusion, and reactive sources. The main issue of stiffness associated with an ODE, and the stability region analysis of a given numerical method, really comes from numerical PDEs. Indeed, for the ODE (2.1.1), any consistent numerical discretization will converge as long as the function $f(u)$ is locally Lipschitz continuous, and the solution u is sufficiently smooth. Moreover, if $f(u)$ is smooth, any consistent method will give full accuracy[1]. However, when we

[1] See exercise 2.1.1.

develop numerical methods for an ODE, we usually require that the numerical method is stable for the following scalar linear ODE:

$$u' = \lambda u \tag{2.1.2}$$

For this simple equation the idea of stiffness translates into a λ with both a large negative real part and large imaginary part, negative or positive. The real part of λ represents the dissipative effect, which is always negative, and usually falls in the region $\text{Re}\lambda \in [-\nu/h^2, 0]$ where ν is the diffusion coefficient and h is spatial grid size. The imaginary part of λ represents the convective effect and is usually bounded by ah, where a is the wave speed for the convection. This can be explained more clearly by the following convection diffusion equation:

$$u_t = au_x + \nu u_{xx} \tag{2.1.3}$$

with a periodic boundary condition $u(t, 0) = u(t, 2\pi)$. The solution can be expanded in a Fourier series, $u(x, t) = \sum_k \hat{u}_k(t) \exp(ikx)$. For each mode $\hat{u}_k(t)$ we have

$$\partial_t \hat{u}_k(t) = (iak - \nu k^2)\hat{u}_k(t)$$

with $\lambda = iak - \nu k^2$. The same is true for the discrete system:

$$\partial_t u_j = a\frac{u_{j+1} - u_{j-1}}{2h} + \nu\frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} \tag{2.1.4}$$

where $u_j(t) \sim u(x_j, t)$, $x_j = jh$, for $j = 1, \cdots, n$, and $h = 2\pi/n$ is the grid size. Expand the solution of (2.1.4) using a discrete Fourier series

$$u_j(t) = \sum_{k=0}^{n} \hat{u}_k(t) \exp(ikx_j).$$

For each mode k, we have

$$\partial_t \hat{u}_k(t) = (i\frac{a}{h}\sin(kh) - 4\frac{\nu}{h^2}\sin^2(kh/2))\hat{u}_k(t) \tag{2.1.5}$$

where

$$\lambda = i\frac{a}{h}\sin(kh) - 4\frac{\nu}{h^2}\sin^2(kh/2), \quad \text{for } k = 1, \cdots, n \tag{2.1.6}$$

There are basically two types of discretizations: implicit or explicit. Obviously, an explicit scheme is (much) more efficient than an implicit one. The main purpose for using an implicit scheme is to insure stability and the use of a large time step. Methods for time dependent problems may be very different from methods used to solve steady state problems. Sometimes in fluid computations, the convection term may be treated explicitly, while the diffusion term is treated implicitly. This is the case in the projection method[2].

Let us first review explicit schemes.

First order explicit scheme, – forward Euler

$$u^{n+1} = u^n + \Delta t f(u^n) \tag{2.1.7}$$

To study the stability, we apply the above scheme to the linear equation (2.1.2) and write it as a recursive formula

$$u^{n+1} = (1 + \lambda \Delta t) u^n \tag{2.1.8}$$

The stability is clearly given by

$$|1 + \lambda \Delta t| < 1 \tag{2.1.9}$$

The set of all λ that satisfy the above equation defines the stability region of the method. Applying this to (2.1.6), we have

$$\left(\frac{a\Delta t}{h}\right)^2 \cos^2(kh/2) + \left(\frac{2\nu\Delta t}{h^2}\right)^2 \sin^2(kh/2) \le \frac{2\nu\Delta t}{h^2} \tag{2.1.10}$$

This basically requires that[3]

$$\frac{a\Delta t}{h} \le \frac{2\nu\Delta t}{h^2} \le 1 \tag{2.1.11}$$

The first inequality gives

$$\mathrm{Re}_c \equiv \frac{ah}{\nu} \le 2 \tag{2.1.12}$$

[2]see exercise 2.1.2

[3]A better explanation is to use energy estimate or maximum principle, see exercise 2.1.3 and 2.2.2.

Re_c is known as the *cell Reynolds number* or *numerical Peclet number*. The inequality (2.1.12) is known as the cell Reynolds number constraint and it imposes a very strong restriction on the spatial grid size. For the convection equation itself($\nu = 0$), the forward Euler scheme with central differencing is un-conditionally unstable. Thus we see that the diffusion term plays the role of stabilizing the scheme. We will show later that the cell Reynolds number constraint is an unnecessary one. It can be avoided by simply using a high order explicit time discretization. The second inequality in (2.1.11) is the stability condition coming from the diffusion term, which must always be enforced for an explicit scheme.

Second order explicit schemes

Now we consider second order schemes for the time discretization. These corresponding to either the mid-point rule or trapezoidal rule(which is better known as Heun's method). The *mid-point rule* gives:

(2.1.13)
$$\begin{cases} u^* = u^n + \frac{1}{2}\Delta t f(u^n), \\ u^{n+1} = u^n + \Delta t f(u^*) \end{cases}$$

and *Heun's method* gives:

(2.1.4)
$$\begin{cases} u^* = u^n + \Delta t f(u^n), \\ u^{n+1} = u^n + \Delta t (f(u^*) + f(u^n))/2 \end{cases}$$

For linear equation (2.1.2) both schemes reduce to

$$u^{n+1} = (1 + \beta + (\beta)^2/2)u^n$$

where $\beta = \lambda \Delta t$. The above formula comes from the Taylor expansion of $\exp(\beta)$ up to second order accuracy. The stability region is given by

(2.1.15)
$$|1 + \beta + (\beta)^2/2| \leq 1$$

The Runge-Kutta(RK) methods One possible third order RK method is given by

Heun's:

$$\begin{cases} u_1 = u^n + \frac{1}{3}\Delta t f(u^n), \\ u_2 = u^n + \frac{2}{3}\Delta t f(u_1), \\ u^{n+1} = \frac{1}{4}(u^n + 3u_1) + \frac{3}{4}\Delta f(u_2) \end{cases} \tag{2.1.16}$$

For the linear ODE (2.1.2), the above scheme reduces to

$$u^{n+1} = (1 + \beta + \frac{1}{2!}\beta^2 + \frac{1}{3!}\beta^3)u^n$$

(Just look at the Taylor expansion of $\exp(\beta)$!). The stability region is given by

$$|1 + \beta + \frac{1}{2!}\beta^2 + \frac{1}{3!}\beta^3| \leq 1 \tag{2.1.17}$$

Jameson proposed the following three stage RK method for the linear equation (2.1.2),

$$\begin{cases} u_1 = u^n + \frac{1}{3}\Delta t f(u^n) \\ u_2 = u^n + \frac{1}{2}\Delta t f(u_1) \\ u^{n+1} = u^n + \Delta t f(u_2) \end{cases} \tag{2.1.18}$$

This is a second order scheme for the nonlinear problem[4]. For the linear problem (2.1.16) and (2.1.18) have identical stability regions. Nevertheless, (2.1.18) is easier to program (all steps are forward euler) and requires less memory than (2.1.16). This scheme is especially useful for steady state computations.

To minimize the stability restriction coming from the convection term, we seek a method whose stability region encompasses as large a portion of the imaginary axis as possible[5]. If one if willing to sacrifice accuracy, among all possible second order three-stage RK methods the following scheme optimizes the stability region due to the convection term:

$$\begin{cases} u_1 = u^n + \frac{1}{2}\Delta t f(u^n) \\ u_2 = u^n + \frac{1}{2}\Delta t f(u_1) \\ u^{n+1} = u^n + \Delta t f(u_2) \end{cases} \tag{2.1.19}$$

[4]See exercise 2.1.4.

[5]See exercise 2.1.5

The stability region is given by

$$|1+\beta+\frac{1}{2}\beta^2+\frac{1}{4}\beta^3| \le 1 \tag{2.1.20}$$

One important fact is that the stability region in (2.1.20) covers $i[-2,2]$ along the imaginary axis. Plugging into (2.1.6) for λ, this tells us that the convection term will be stable under the standard CFL constraint $a\Delta t \le 2h$. Thus, no help is required from the diffusion term. This is remarkable! It tells us that we can compute large Reynolds flow without having any cell Reynolds number constraint[6].

The fourth order classic Runge-Kutta method is given by,

$$\begin{cases} u_1 = u^n + \frac{1}{2}\Delta t f(u^n), \\ u_2 = u^n + \frac{1}{2}\Delta t f(u_1), \\ u_3 = u^n + \Delta t f(u_2), \\ u^{n+1} = \frac{1}{3}(-u^n + y_1 + 2y_2 + y_3) + \frac{1}{6}\Delta t f(u_3) \end{cases} \tag{2.1.21}$$

for the linear ODE (2.1.2), it reduces to

$$u^{n+1} = (1+\beta+\frac{1}{2!}\beta^2+\frac{1}{3!}\beta^3+\frac{1}{4!}\beta^4)u^n$$

The stability region is given by

$$|1+\beta+\frac{1}{2!}\beta^2+\frac{1}{3!}\beta^3+\frac{1}{4!}\beta^4| \le 1 \tag{2.1.22}$$

The stability region covers $i[-2.7, 2.7]$ in the imaginary axis.

We shall also remark that the basic structure in all the above schemes is forward euler. We can easily modify a code once have the forward euler version.

The multi-step method

There is much more to be said concerning optimizing the stability regions restrictions coming from the convection term if one is willing to investigate multi-step methods. This is very important for the unsteady fluids computation. We will only describe two ways to design multi-step methods: Adams-Bashforth(A-B) approach and the backward differential approach.

[6]See exercise 2.1.5, 2.1.6.

In the Adams-Bashforth approach, we reformulate the ODE (2.1.1) in the following integral form:

$$u^{n+1} = u^n + \int_{t^n}^{t^{n+1}} f(u(t))\,dt \tag{2.1.23}$$

and extrapolate (f(u))(t) from the points $t^n, t^{n-1}, \cdots$. The second order A-B scheme is given by:

$$u^{n+1} = u^n + \Delta t(\frac{3}{2}f(u^n) - \frac{1}{2}f(u^{n-1})) \tag{2.1.24}$$

For the linear equation (2.1.2) the above equation reduces to

$$u^{n+1} = u^n + \beta(\frac{3}{2}u^n - \frac{1}{2}u^{n-1}).$$

Letting $u^n = z^n$ gives the following characteristic equation:

$$z^2 - z = \beta(\frac{3}{2}z - \frac{1}{2}) \tag{2.1.25}$$

and the stability region is given by

$$\beta = \frac{2(z^2 - z)}{3z - 1}, \quad |z| \le 1. \tag{2.1.26}$$

For the general explicit multi-step methods, the characteristic equation becomes

$$\sigma(z) = \beta\rho(z) \tag{2.1.27}$$

whose stability region is given by

$$\beta = \frac{\sigma(z)}{\rho(z)}, \quad |z| \le 1.$$

where $\rho(z)$ describes the approximation for the right side of (2.3), and $\sigma(z)$ describes the approximation of the left side of (2.3). In the Adams-Bashforth approach, $\sigma(z) = z^n - z^{n-1}$ where n is the order of the method. In the backward differential approach, we use $u^{n+1}, u^n, \cdots$, to obtain a high order approximation of $\partial_t u$ in (2.1). The stability analysis is effectively the same as that just outlined above for the A-B methods. See exercise 2.1.8.

Implicit Schemes and Predictor-Corrector methods

An easy way to solve the stability problems associated with fluids computations is to blindly use implicit schemes. Unfortunately, this approach is widely used in the field. The implicit treatment of the convection term results in a very large sparse nonlinear system that we must solve. This is usually not too much more expensive to solve than solving a very large sparse linear system by using Newton's method. The second order convergence of Newton's method can usually be achieved since we have a very good initial guess, namely, u^n. Nevertheless, one should only use this implicit approach when the cost of the time stepping is the main issue. This is the case for steady state or low Reynolds number flow computations.

The standard first and second order implicit schemes are backward Euler and the Crank-Nicholson(C-N) scheme, respectively. They are as follows.

$$u^{n+1} = u^n + \Delta t f(u^{n+1}) \tag{2.1.28}$$

$$u^{n+1} = u^n + \frac{1}{2}\Delta t(f(u^{n+1}) + f(u^n)) \tag{2.1.29}$$

The stability regions for the above two schemes are given by

$$|1-\beta| \geq 1, \qquad |2-\beta| \geq |2+\beta| \tag{2.1.30}$$

respectively. Notice that both of them cover all the left half plane $Re\lambda \leq 0$. A scheme with this property is called A-stable scheme. Dahlquist showed that any A-stable scheme is at most second order accuracy[7].

A general way to obtain other implicit multi-step methods is through the Adams-Moulton approach which uses the points $t^{n+1}, t^n, \cdots$ to interpolate $f(u)(t)$ at the right hand side of the integral (2.1.23). C-N is the second order A-M scheme, and the third order scheme is given by

$$u^{n+1} = u^n + \frac{1}{12}\Delta t(5f(u^{n+1}) + 8f(u^n) - f(u^{n-1})) \tag{2.1.31}$$

We can sometimes solve the equation arising from the use of an implicit scheme by iteration. An effective way of doing this is the so called predictor-corrector methods. We

[7]See exercise 2.1.7

first use an explicit scheme, say A-B method, to obtain a good initial guess to $\tilde{u}^{n+1}$. We then use it to evaluate $f(\tilde{u}^{n+1})$ in the implicit scheme, say an A-M method. You can gain more stability by repeated iteration[8]. This kind of approach is known as Adams-Moulton predictor-corrector methods. More detail discussion can be found in Gears book.

Exercise:

1. Suppose the function $f(u)$ in ODE (2.1) is locally Lipschitz continuous. Show that any consistent numerical method will converge when the solution is smooth. Furthermore, if $f(u)$ is smooth, then the numerical method will have full accuracy, that is you should be able to obtain an error estimate up to the order of the truncation error.

2. Perform the stability analysis for (2.4) with the convection term discretized with the second order A-B method, and the diffusion term discretized with the C-N method. This treatment is used in the second order projection methods for NSE.

3. Using energy estimates, or the maximum principle, give a stability and error estimate for the forward Euler discretization of (2.4) which satisfies the stability condition (2.11). Could you get a better stability condition? Can you do the same analysis for the mid-point rule?

4. Show that the three stage RK methods (2.18) and (2.19) are only second order accurate. Perform an accuracy check and compare the results with a true third order scheme (2.16).

5. Show that (2.19) gives the largest stability region along the imaginary axis among all second order three stage RK methods. Show, by numerical computation, that the scheme is stable under the standard CFL condition for the convection diffusion problem (2.4), when the diffusion coefficient is zero. Compare it with the first and second order explicit schemes.

[8]See exercise 2.1.9.

6. Show the second order accuracy, both in time and space, for the three stage RK (2.9) for the convection diffusion problem (2.4) under the standard CFL condition $|a|\Delta \leq h$, and diffusion stability condition $2\nu\Delta \leq h^2$. Could you extend this result for case of $\nu = 0$ and the case of the variable convection speed $a(x)$?

7. Prove the Dahlquist theorem: An A-stable scheme is at most second order accuracy.

8. Derive the class of multi-step schemes using the backward differential approach and find their stability regions.

9. Show, numerically, that the Adams-Moulton predictor-corrector methods have larger stability regions compared with those of the A-B methods. Do you gain more stability if you perform more iterations in the corrector step?

§2.2. Issues concerning resolution, central or upwind discretization, Reynolds Number, boundary Layer, shear layer and shock layer

Back to the convection diffusion equation:

$$u_t = au_x + \nu u_{xx}. \tag{2.2.1}$$

We analyzed in section (2.1) the stability restrictions that arise from both the spatial and time discretizations of (2.2.1) in the case of periodic boundary conditions, $u(t,0) = u(t,2\pi)$. A cell Reynolds constraint results when we use a low order explicit time discretization along with centered spatial differencing. This cell Reynolds constraint can easily be overcome by using a higher order explicit time discretization, or more costly implicit time discretization.

The solution of (2.1.3) with periodic boundary conditions will be smooth if the initial data is smooth. In this case we were able to easily analyze the stability region of the numerical scheme. Similar conclusions can be reached in the case of variable coefficients, and even a nonlinear problem, as long the solution is smooth. In the case of resolved computational solutions of practical incompressible flows, the analysis in the previous section can be viewed as a basic guideline in examine the stability of the time discretization. By resolved we mean that at least 8 to 10 points are present per wave structure in the true solution. We should elaborate this point. Imagine a flow along a wall. Very near to the wall the flow is considerably different in character from the flow away from the wall. This is due to the effects of friction, whose magnitude you can think of as represented by, more or less, $1/Re$. The flow structure near the wall is referred to as the boundary layer, for obvious reasons. It is physical! It's thickness is typically on the order of $1/\sqrt{Re}$, and one should have at least 8 to 10 computational points in this region if one is to say that we have computed a reliable solution. So typically we must take $h \leq 1/10\sqrt{Re}$. This is considerably less restrictive then the cell Reynolds number constraint.

However, suppose we impose a fixed boundary condition for (2.1.3), say,

$$u(0,t) = 1, \quad u(1,t) = 0. \tag{2.2.2}$$

The solution will reach the steady solution represented by the solution to

$$au_x + \nu u_{xx} = 0, \quad u(0) = 1, u(1) = 0. \tag{2.2.3}$$

This will occur on a time scale proportional to the typical length divided by the speed(a). This is referred to as the convective time scale. We can easily solve the steady equation (2.2.3) to obtain the solution

$$u(x) = (1 - e^{(1-x)a/\nu})/(1 - e^{a/\nu}) \tag{2.2.4}$$

There is a boundary layer at right boundary if $a < 0$, a boundary layer at left boundary if $a > 0$. The width of this boundary layer is of $O(\nu/a)$. This is **not** the same boundary layer referred to above in conjunction with the typical incompressible flow.

Using central spatial differencing for the steady equation (2.2.3), we have

$$u_{j+1} - u_{j-1} + \frac{2\nu}{ah}(u_{j+1} - 2u_j + u_{j-1}) = 0,$$

or

$$u_j = \left(\frac{1}{2} + \frac{ah}{4\nu}\right) u_{j+1} + \left(\frac{1}{2} - \frac{ah}{4\nu}\right) u_{j-1}. \tag{2.2.5}$$

When

$$\mathrm{Re}_c = |a|h/\nu \le 2 \tag{2.2.6}$$

both of the coefficients in (2.2.5) are positive, which using the maximum principle, shows that the solution is monotone[9]. This cell Reynolds constraint is essential. Without it oscillations, and even instabilities, will occur in the numerical solution. Indeed, if we evaluate the solution at grid point x_{n-1}, next to the right boundary, we have

$$u(x_{n-1}) = (1 - e^{-\mathrm{Re}_c})/(1 - e^{-\mathrm{Re}_c/h}) \sim 0.86, \quad \text{if } \mathrm{Re}_c = 2 \tag{2.2.7}$$

for the case $a < 0$. The same would occur at the left boundary if $a > 0$. Notice that in this example

$$u(x_{n-1}) = \frac{u(x_{n-1}) - u(x_n)}{u(x_0) - u(x_n)}.$$

[9]See exercise 2.2.1, 2.2.2.

Hence, approximately 86% of the boundary layer transition is represented by one point. In this case the boundary layer is clearly under-resolved. But we have satisfied the cell Reynolds number constraint. This shows that stability and resolution are two different issues.

Instead of using central differencing in space, one can use upwind differencing to improve the stability. But the upwind differencing can not improve the accuracy, i.e., the resolution. Indeed, in the upwind approach some numerical viscosity is added to the scheme so that the effective cell Reynolds number is limited to 2 as we will explain below. The state of the art modern shock capturing schemes rely on the fact that one can robustly and effectively add numerical viscosity to a scheme so that a sharp monotone transition will be preserved in the computed shock layer. Usually only a few computational points are present in the shock transition. As we have seen above, this basically says that the effective cell Reynolds number is less than, but close to 2[10]. To explain this point more clearly, let us look at the upwind scheme:

$$u_j - u_{j-1} + \frac{\nu}{ah}(u_{j+1} - 2u_j + u_{j-1}) = 0 \tag{2.2.8}$$

when $a < 0$. This gives

$$(2 + |a|h/\nu)u_j = u_{j+1} + (1 + |a|h/\nu)u_{j-1}$$

and we know that the profile is always monotone[11]. However, lets look at the modified equation for (2.2.8),

$$au_x + (\nu + |a|h/2)u_{xx} = O(h^2). \tag{2.2.9}$$

This comes from simple Taylor expansion of the upwinding scheme, upto $O(h^2)$. Our computed solution using the upwinding scheme better approximates the solution to (2.2.9) than (2.2.3). The effective viscosity in the modified equation is $\bar{\nu} \equiv \nu + |a|h$ and the effective cell Reynolds number is

$$\mathrm{Re}_c = \frac{|a|h}{\bar{\nu}} = 2\frac{|a|h}{\nu + |a|h} \sim 2, \quad \text{as } \nu \sim 0 \tag{2.2.10}$$

[10]See exercise 2.2.4.

[11]See exercise 2.2.3

Thus, the effective cell Reynolds number is always less than 2. So while we gained better stability, we have completely lost resolution.

Exercise:

1. Show numerically that the center differencing approximation (2.2.5) with boundary condition (2.2.2) converges, and gives a monotone profile in the boundary layer when the cell Reynolds constraint is satisfied. Also, show numerically that the scheme is unstable when the cell Reynolds number is larger than 2.

2. Show that the central differencing scheme, applied to (2.2.1), is indeed a monotone scheme when the diffusion stability condition $2\nu\Delta t \leq h^2$ and the cell Reynolds number constraint are satisfied!

3. Show that the upwinding scheme is always stable and gives monotone profiles in the boundary layer even when the diffusion coefficient ν is very small. Check the accuracy in the computed steady state solution with the exact solution (2.35) in the maximum norm. Compare it with the central differencing scheme at the same resolution when the diffusion coefficient is chosen so that the cell Reynolds number is close to 2.

4. Use MUSCL scheme or ENO scheme to replace the upwind scheme in the above exercise. Do you get the same result? Can you show that the effective cell Reynolds number in MUSCL scheme is always less than 2? This is a very good research project and is definitely publishable.

§2.3. Numerical methods for the Poisson equation

For simplicity, we will consider the Poisson equation

$$\Delta u = f \tag{2.3.1}$$

on a unit square $\Omega = [0,1]^2$, with Dirichlet boundary conditions:

$$u \,|_{\Gamma} = u_b\,, \tag{2.3.2}$$

or Neumann boundary condition:

$$\frac{\partial u}{\partial \boldsymbol{n}} = u_b\,, \qquad \text{on } \Gamma\,. \tag{2.3.3}$$

where Γ is the boundary of Ω. Assume that the boundary data u_b in (2.3.3) is consistent with (2.3.1), i.e.,

$$\int_\Omega f\,dx\,dy = \int_\Gamma u_b\,ds$$

The solution of (2.3.1)-(2.3.3) is unique up to to constant. For simplicity, we take the constant to be zero.

We will use the following uniform computational grid:

$$x_i = ih, \quad y_j = jh, \quad i,j = 0,1,\cdots,n, \quad h = 1/n \tag{2.3.4}$$

Denote the standard second order centered difference operators as

$$D_x^2 u = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \quad D_y^2 u = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} \tag{2.3.5}$$

Then the standard second order approximation of (2.3.1) is given by

$$\Delta_h u = f, \quad \Delta_h = D_x^2 + D_y^2. \tag{2.3.6}$$

We will discuss a 4th order compact difference in a later subsection.

Fast Poisson Solver for Dirichlet Problem

In the case of the Dirichlet boundary condition (2.3.2), the computational point are all of the interior points, and the boundary values are given by the boundary data (2.3.2).

The system (2.3.6) can be solved very efficiently via FFT, as we will see below. We first move the boundary data to the right hand side of the system (2.3.6). For instance, along $x = 0$ we would have

$$f_{1,j} \to f_{1,j} - u_{0,j}/h^2 \tag{2.3.7}$$

at the left side boundary. Similarly for other three sides of the boundary Γ. The resulting system can be viewed as a system with homogeneous (zero) boundary data. Taking the Sine transformation of both $u_{i,j}$ and $f_{i,j}$, we have

$$u_{i,j} = \sum_{k,\ell=1}^{n-1} \hat{u}_{k,\ell} \sin(kx_i\pi)\sin(\ell y_j\pi) \qquad f_{i,j} = \sum_{k,\ell=1}^{n-1} \hat{f}_{k,\ell} \sin(kx_i\pi)\sin(\ell y_j\pi) \tag{2.3.8}$$

For each of the Fourier modes present above, one can easy verify that[12]

$$D_x^2 \sin(kx_i\pi) = \lambda_k \sin(kx_i\pi), \quad \text{for } i = 1, 2, \cdots, n-1 \tag{2.3.9}$$

where

$$\lambda_k = -\frac{4}{h^2}\sin^2\left(\frac{k\pi}{2n}\right), \quad \text{for } k = 1, 2, \cdots, n-1 \tag{2.3.10}$$

is the Fourier multiplier for the D_x^2 operator. (2.3.6) is thus equivalent to

$$(\lambda_k + \lambda_\ell)\hat{u}_{k,\ell} = \hat{f}_{k,\ell} \tag{2.3.11}$$

Note that the Fourier multiplier $\lambda_k + \lambda_\ell \neq 0$ for $k, \ell = 1, \cdots, n-1$.

The computation of the solution to (2.3.6) consists the following four steps:

1. Homogenizing the boundary condition as in (2.3.7)

2. Taking the Sine transformation of $f_{i,j}$ to obtain $\hat{f}_{k,\ell}$, for $k, \ell = 1, 2, \cdots, n-1$, see (2.3.8)

3. Obtaining $\hat{u}_{k,\ell}$ from $\hat{f}_{k,\ell}$ by dividing by the Fourier multipliers, for $k, \ell = 1, 2, \cdots, n-1$, see (2.3.11)

4. Taking the inverse Sine transformation of $\hat{u}_{k,\ell}$ to obtain $u_{i,j}$, for $i, j = 1, 2, \cdots, n-1$, see (2.3.8)

[12]See exercise 2.3.1

The boundary value of $u_{0,j}$ shall then be restored from the given boundary data along Γ.

Fast Poisson Solver for the Neumann Problem

In the case of the Neumann boundary condition, the computational grid consists of all the interior points, as well as the grid points along Γ. The outside points, say $x_{-1,j}$, next to the boundary will be used as the "ghost points". Their values can be determined by the Neumann boundary and implemented as

$$\frac{u_{-1,j} - u_{1,j}}{2h} = u_b(0, y_j), \quad j = 0, 1, \cdots, n \tag{2.3.12}$$

(out-normal direction divide difference!) for the left side boundary. Similarly for the other three sides of the boundary Γ. We use the Cosine transformation to solve the resulting system. Detailed steps are as follows:

1. Homogenizing the boundary condition by moving the boundary data to the right hand side,

$$f_{0,j} \to f_{0,j} - \frac{2}{h} u_b(y_j), \tag{2.3.13}$$

at the left side boundary. Similarly for the other three sides the of boundary.

2. Take the Cosine transformation of $f_{i,j}$ to obtain $\hat{f}_{k,\ell}$, for $k, \ell = 0, 1, 2, \cdots, n$

$$f_{i,j} = \sum_{k,\ell=0}^{n} \hat{f}_{k,\ell} \cos(k x_i \pi) \cos(\ell y_j \pi) \tag{2.3.14}$$

3. Obtain $\hat{u}_{k,\ell}$ from $\hat{f}_{k,\ell}$ by dividing by the Fourier multipliers, for $k, \ell = 0, 2, \cdots, n$.

$$\hat{u}_{k,\ell} = \hat{f}_{k,\ell} / (\lambda_k + \lambda_\ell) \tag{2.3.15}$$

where λ_k are same the as in (2.3.10). Notice that when $k = \ell = 0$, the Fourier multiplier is zero. That tells us that there is one degree of freedom (up to a constant). We fix this freedom by setting (zero constant)[13]

$$\hat{u}_{0,0} = 0 \tag{2.3.16}$$

[13]See exercise 2.3.2 for the issue of numerical consistence.

4. Take inverse Cosine transformation of $\hat{u}_{k,\ell}$ to obtain $u_{i,j}$, for $i,j = 0,1,\cdots,n$

$$u_{i,j} = \sum_{k,\ell=0}^{n} \hat{u}_{k,\ell} \sin(kx_i\pi)\sin(\ell y_j\pi) \tag{2.3.17}$$

Error estimate and accuracy checking

It is easy to use energy estimates or the Maximum principle to show that the scheme has full 2^{nd} order accuracy[14]

$$u_{i,j} = u(x_i,y_j) + h^2 v(x_i,y_j) + O(h^4) \tag{2.3.18}$$

for some function v. In the finite difference case with a uniform grid, we are able to obtain a sharp error estimate (equality) and perform a numerical accuracy check by computing solutions on set of different grid sizes h[15].

$$\|u_h - u\|/h^2 \sim \text{constant} \tag{2.3.19}$$

where the norm $\|\cdot\|$ is usually taken as L^1, L^2 and L^∞.

Fourth Order Compact Method for the Dirichlet Problem

Taylor expansion gives

$$\partial_x^2 = D_x^2 - \frac{h^2}{12}D_x^4 + O(h^4), \quad \partial_y^2 = D_y^2 - \frac{h^2}{12}D_y^4 + O(h^4). \tag{2.3.20}$$

Hence, we have

$$\Delta = D_x^2 + D_y^2 - \frac{h^2}{12}(D_x^4 + D_y^4) + O(h^4) = \Delta_h - \frac{h^2}{12}\Delta_h^2 + \frac{h^2}{6}D_x^2 D_y^2 + O(h^4).$$

We can decomposed the right hand side of the above equation into a product of two compact operators (9-point stencil),

$$\Delta = (\Delta_h + \frac{h^2}{6}D_x^2 D_y^2)(1 - \frac{h^2}{12}\Delta_h) + (O(h^4). \tag{2.3.21}$$

Using

$$1 - \frac{h^2}{12}\Delta_h = (1 + \frac{h^2}{12}\Delta_h)^{-1} + O(h^4)$$

[14]See exercise 2.3.3

[15]See exercise 2.3.4.

we obtain a 4th order compact operator

$$\Delta = (1 + \frac{h^2}{12}\Delta_h)^{-1}(\Delta_h + \frac{h^2}{6}D_x^2 D_y^2) + O(h^4) \tag{2.3.22}$$

and a 4th order compact approximation to the Poisson equation (2.3.1)

$$(\Delta_h + \frac{h^2}{6}D_x^2 D_y^2)u = (1 + \frac{h^2}{12}\Delta_h)f. \tag{2.3.23}$$

The computational grid points are still all the interior grid points, and the boundary value of u is determined by the given boundary data. In this approach, we need to know the boundary value of f. Just as in the 2^{nd} order case we can use the Sine transformation to obtain a fast solver. Using energy estimates and a high order expansion, one can obtain the following error estimate:

$$u_{i,j} = u(x_i, y_j) + h^4 v(x_i, y_j) + O(h^6) \tag{2.3.24}$$

for some function v. We can use the above estimate to check the accuracy[16].

Poisson equation on a Staggered grid, Fast Solver and Boundary Condition.

The staggered grid in a unit square is given by

$$x_{i+1/2} = (i + 1/2)h, \quad y_{j+1/2} = (j + 1/2)h, \quad i, j = -1, 0, 1, \cdots, n, \quad h = 1/n \tag{2.3.25}$$

The boundary is located at the midpoint between two grid points. For instance, the left boundary is

$$x_0 = \frac{1}{2}(x_{-1/2} + x_{1/2}). \tag{2.3.26}$$

The computational grid points will be all the interior points for both the Dirichlet problem (2.3.1)-(2.3.2) and the Neumann problem (2.3.1)-(2.3.3). For a second order scheme

$$\Delta_h u = f, \quad \text{on } (x_{i+1/2}, y_{j+1/2}), \quad i, j = 0, 1, \cdots, n-1 \tag{2.3.27}$$

one need “ghost points” $x_{-1/2}$ on the left boundary, and $x_{n+1/2}$ on the right side boundary. For the Dirichlet problem, the boundary condition is enforced as follows:

$$\frac{u_{-1/2,j} + u_{1/2,j}}{2} = u_b(y_j)\,, \tag{2.3.28}$$

[16]See exercise 2.3.5.

on the left side boundary. Similarly for other the other three sides of the boundary. This is usually referred as a reflection boundary condition.

For the Neumann problem, the boundary condition is enforced as follows:

$$\frac{u_{-1/2,j} - u_{1/2,j}}{h} = u_b(y_j)\,, \tag{2.3.29}$$

(outward-normal direction) on the left side of boundary. Similarly for the other three sides of the boundary.

The linear systems can be solved quite efficient by quarter-wave FFT transformation. We leave it as an exercise[17].

The accuracy of the reflection boundary condition is rather confusing. Let us explain this with a simple example:

$$u'' = f, \quad u(0) = (1) = 0$$

approximated with the standard second order centered difference and reflection boundary conditions as in (2.3.27) and (2.3.28). A simple truncation error analysis at $x_{\frac{1}{2}}$ gives

$$D_x^2 u(x_{\frac{1}{2}}) = \frac{3}{4} u''(x_{\frac{1}{2}}) + \frac{h}{8} u'''(0) + O(h^2)\,.$$

suggesting that the operator D_x^2 is not consistent with the Laplacian near the boundary. A more sophisticated error analysis reveals that the overall scheme still has second order accuracy as we see below. Let

$$\bar{u}_{i+\frac{1}{2}} = u(x_{i+\frac{1}{2}}) - \frac{h^2}{8}[u''(0) + (u''(1) - u''(0)) x_{i+1/2}]\,, \qquad \text{for } i = 0, 1, \cdots n-1$$

and

$$\bar{u}_{-\frac{1}{2}} = -\bar{u}_{\frac{1}{2}}\,, \qquad \bar{u}_{n+\frac{1}{2}} = -\bar{u}_{n-\frac{1}{2}}\,.$$

Clearly

$$D_x^2 \bar{u}_{i+\frac{1}{2}} = u''(x_{i+\frac{1}{2}}) + O(h^2)\,, \qquad \text{for } i = 1, \ldots, n-2\,.$$

[17]See exercise 2.3.6.

At $x_{\frac{1}{2}}$,

$$\begin{aligned} D_x^2 \bar{u}_{\frac{1}{2}} &= \frac{\bar{u}_{\frac{3}{2}} - 3\bar{u}_{\frac{3}{2}}}{h^2} = \frac{u(x_{\frac{3}{2}}) - 3u(x_{\frac{3}{2}})}{h^2} + \frac{1}{4}u''(0) \\ &= \tfrac{3}{4}u''(x_{\frac{1}{2}}) + \tfrac{h}{8}u'''(0) - \tfrac{1}{4}u''(0) + O(h^2) = u''(x_{\frac{1}{2}}) + O(h^2)\,. \end{aligned}$$

Similarly, we have

$$D_x^2 \bar{u}_{n-\frac{1}{2}} = \frac{u(x_{n-\frac{3}{2}}) - 3u(x_{n-\frac{3}{2}})}{h^2} + \frac{1}{4}u''(1) = u''(x_{n-\frac{1}{2}}) + O(h^2)\,.$$

This gives

$$u(x_{i+\frac{1}{2}}) - u_{i+\frac{1}{2}} = O(h^2), \quad \text{for } i = 0, 1, \cdots, n-1\,. \tag{2.3.30}$$

Exercise:

1. Verify formula (2.3.9).

2. There are always issues concerning the numerical consistency for the Neumann problem even when the continuous problem is consistent. Show that the fast solver (2.3.13-17) eliminates such problems in the discrete case even if inconsistencies are present in the approximation.

3. Using energy estimates or the Maximum principle to show (2.3.18).

4. Write a code for both the Dirichlet problem and the Neumann problem. Check the accuracy by using (2.3.19) for a different set of grids, $n = 8, 16, 32, 64$.

5. Using energy estimates and high order expansions, show (2.3.24) and write a code to check the accuracy.

6. Derive a fast Poisson solver on staggered grid for both the Dirichlet problem and the Neumann problem with Quarter-wave FFT.

7. Give a complete proof of (2.3.30) for the 2-d problem.

§2.4. Numerical Methods for the Heat Equation

The standard second order approximation of the heat equation

$$\partial_t u = \nu \Delta u \,, \tag{2.4.1}$$

is given by

$$\partial_t u = \nu \Delta_h u \,. \tag{2.4.2}$$

As we discussed in §2.1, the time-stepping can be explicit or implicit. The choice usually will depend on the the size of ν. For the explicit scheme, the following stability condition must be satisfied:

$$4\nu \Delta t \leq h^2. \tag{2.4.3}$$

The most commonly used implicit scheme is the C-N scheme, which is unconditional stable. However, in this case one must solve a Poisson type equation at each time step.

There are typically two possible types of boundary conditions that one can prescribe for u. The Dirichlet boundary condition

$$u \mid_{\Gamma} = u_b \,, \tag{2.4.4}$$

or the Neumann boundary condition

$$\frac{\partial u}{\partial \boldsymbol{n}} = u_b \,, \qquad \text{on } \Gamma \,, \tag{2.4.5}$$

where u_b is a known function.

In the case of the Dirichlet boundary condition (2.4.4), the unknowns consists of all the interior grid points, since the boundary values are given by the boundary data u_b.

In the case of the Neumann boundary condition, the unknowns include all the interior grid points as well as the boundary grid points. In order to solve the problem numerically we must consider outside points, say $x_{-1,j}$, next to the left-hand side boundary. We call these the "ghost points". Their values can be determined from the Neumann boundary condition as shown below:

$$u_{-1,j} = u_{1,j} + 2h * u_b(x_0, y_j) \tag{2.4.6}$$

For either boundary condition the resulting numerical approximation is fully 2^nd order accurate in space.[18].

Fourth Order Approximation for the Dirichlet problem: Long Stencil

For simplicity, we only consider a 1-d problem here. The standard 4th order long-stencil approximations of ∂_x and ∂_x^2 are given by

$$\partial_x = \tilde{D}_x(1 - \frac{h^2}{6}D_x^2) + O(h^4) \tag{2.4.7}$$

and

$$\partial_x^2 = D_x^2(1 - \frac{h^2}{12}D_x^2) + O(h^4) \tag{2.4.8}$$

and using these, a fourth order spatial approximation of (2.4.1),(treating time continuously for now), is given by

$$\partial_t u = \nu(D_x^2 - \frac{h^2}{12}D_x^2 D_x^2)u\,. \tag{2.4.9}$$

Formally this approximation looks like the hyper-diffusion equation

$$\partial_t u = \nu(\partial_x^2 - \frac{h^2}{12}\partial_x^4)u.$$

Note that both the second and fourth order difference operators that appear above on the right-hand side of (2.4.9) are well-posed. This is important since we must resort to using one-sided approximations to $D_x^2 D_x^2$ near the boundary of our domain. As we will see later, the use of such one-sided approximations will not result in any stability concerns.

The unknown computational grid points consist only of the interior points since the boundary values are given by the boundary data u_b. However, to implement the numerical method, we also need to know the value of u_{-1} at the outside grid points, say $x_{i,-1}$, next to the left-side boundary. These grid points are known as “ghost point”. The main issue is prescribe appropriate values to these ”ghost points” such that the resulting method remains stable and achieves high order numerical accuracy.

[18]See exercise 2.4.1 and 2.4.2.

In deriving the value of u at the "ghost points", one-sided approximations are usually used. Shorter one-sided stencils usually result in better stability. Sometimes we can even use information from the PDE, or derivatives of the PDE, near the boundary to reduce the number of needed points in the stencil. As an illustration of this we will now derive a 4th order approximation for the "ghost points". The Taylor expansion of u_{-1} centered at the boundary point x_0, gives to $O(h^4)$:

$$u_{-1} = u_0 - (hu_0') + (\frac{h^2}{2!}u_0'') - (\frac{h^3}{3!}u_0^{(3)}) + (\frac{h^4}{4!}u_0^{(4)}) + O(h^5) \tag{2.4.10}$$

We first need to identify the known quantities on the right hand side of the above expansion. Clearly, u_0 is known from the given boundary data u_b. We can also exactly derive the value of u_0'' using the PDE itself by evaluating the PDE (2.4.1) at the boundary points x_0 to get

$$\partial_t u(x_0, t) = \nu \partial_x^2 u(x_0, t).$$

Hence,

$$\partial_x^2 u_0(t) = \frac{1}{\nu}\partial_t u_b(t) \tag{2.4.11}$$

In order to eliminate the other three unknown quantities on the right hand side of (2.4.10), we will generally need the value of u at u_1, u_2, and u_3. Expanding each of these in a Taylor series,again centered at x_0, gives to $O(h^5)$:

$$\begin{aligned} u_1 &= u_0 + \ (hu_0') + \ (\tfrac{h^2}{2!}u_0'') + \ (\tfrac{h^3}{3!}u_0^{(3)}) + \ (\tfrac{h^4}{4!}u_0^{(4)}) + O(h^5) \\ u_2 &= u_0 + 2(hu_0') + 2^2(\tfrac{h^2}{2!}u_0'') + 2^3(\tfrac{h^3}{3!}u_0^{(3)}) + 2^4(\tfrac{h^4}{4!}u_0^{(4)}) + O(h^5) \\ u_3 &= u_0 + 3(hu_0') + 3^2(\tfrac{h^2}{2!}u_0'') + 3^3(\tfrac{h^3}{3!}u_0^{(3)}) + 3^4(\tfrac{h^4}{4!}u_0^{(4)}) + O(h^5) \end{aligned} \tag{2.4.12}$$

We now take a linear combination of the equations in (2.4.12), say with coefficients $\alpha, \beta, and \gamma$. We would like this linear combination to agree to high order with the remaining unknown terms of (2.4.10), namely

$$-(hu_0') - (\frac{h^3}{3!}u_0^{(3)}) + (\frac{h^4}{4!}u_0^{(4)})$$

This requires that

$$\begin{aligned} \alpha + 2\beta + 3\gamma &= -1 \\ \alpha + 2^3\beta + 3^3\gamma &= -1 \\ \alpha + 2^4\beta + 3^4\gamma &= 1 \end{aligned} \tag{2.4.13}$$

Solving the above system (2.4.13), we obtain

$$\alpha = -6/11, \quad \beta = -4/11, \quad \gamma = 1/11 \tag{2.4.14}$$

Subtracting (2.4.10) from the linear combination gives

$$\alpha u_1 + \beta u_2 + \gamma u_3 - u_{-1} = (\alpha + \beta + \gamma - 1)u_0 + (\alpha + 4\beta + 9\gamma - 1)(\frac{h^2}{2!}u_0'') + O(h^5)$$

We now plug in our values for u_0, u_0'', and α, β, and γ that gives the desired one-sided approximation of u_{-1}:

$$u_{-1} = \frac{20}{11}u_0 - \frac{6}{11}u_1 - \frac{4}{11}u_2 + \frac{1}{11}u_3 + \frac{12h^2}{11\nu}\partial_t u_b + O(h^5)\,. \tag{2.4.15}$$

Now we show that the the use of (2.4.15) in the scheme is indeed stable. For simplicity, we assume $u_b = 0$. Using standard Energy estimates of (2.4.9) leads to

$$\partial_t \sum_{i=1}^{n-1} u_i^2 + 2\nu \sum_{i=1}^{n-1} (D_x^+ u_i)^2 + \frac{\nu h^2}{6}\left(\sum_{i=1}^{n-1} (D_x^2 u_i)^2 + u_1 D_x^2 u_0 + u_{n-1} D_x^2 u_n\right) = 0. \tag{2.4.16}$$

We can rewrite the one-sided "ghost point" boundary condition (2.4.15) as

$$D_x^2 u_0 = \frac{1}{h^2}\left(\frac{5}{11}u_1 - \frac{4}{11}u_2 + \frac{1}{11}u_3\right) \tag{2.4.17}$$

This is very much like the vorticity boundary condition as we will see in next section. We now rewrite (2.4.17) as

$$D_x^2 u_0 = -\frac{2}{11}D_x^2 u_1 + \frac{1}{11}D_x^2 u_2 \tag{2.4.18}$$

Using the Cauchy-Schwartz inequality gives the estimate

$$u_1 D_x^2 u_0 \geq -\frac{5}{4 \times 11^2}u_1^2 - (D_x^2 u_1)^2 - (D_x^2 u_2)^2. \tag{2.4.19}$$

The first term on the right-hand side above can be controlled by one of terms in the second summation appearing in (2.4.16). The next two terms on the right-hand side of (2.4.19) can be controlled by terms appearing in the third summation of (2.4.16). We can take care of the $u_{n-1}D_x^2 u_n$ term in (2.4.16) in a similar fashion. Using these estimates in (2.4.16) gives

$$\partial_t \sum_{i=1}^{n-1} u_i^2 + \nu \sum_{i=1}^{n-1} (D_x^+ u_i)^2 \leq 0 \tag{2.4.20}$$

Using standard error estimates, one can easily show that the scheme has full 4th order accuracy[19].

Fourth Order Approximation for the Neumann problem: Long Stencil

The main difference in the case of the Neumann problem is that the computational grid points consist of all the interior points, as well as the boundary points. We will now need 2 "ghost points", namely u_{-1} and u_{-2}, for the 4th order approximations in (2.4.9). To obtain a high order one-sided approximation of these "ghost" values, we expand both in a Taylor expansion centered at x_0:

$$u_{-1} = u_0 - (hu_0') + (\frac{h^2}{2!}u_0'') - (\frac{h^3}{3!}u_0^{(3)}) + (\frac{h^4}{4!}u_0^{(4)}) + O(h^5) \tag{2.4.21}$$

and

$$u_{-2} = u_0 - 2(hu_0') + 2^2(\frac{h^2}{2!}u_0'') - 2^3(\frac{h^3}{3!}u_0^{(3)}) + 2^4(\frac{h^4}{4!}u_0^{(4)}) + O(h^5) \tag{2.4.22}$$

Again, we first need to identify the known quantities on the right-hand side of each of the above expansions. Clearly, u_0' is known form the boundary data. We can derive the value of u_0''' using the PDE in a similar fashion as was used above in the Dirichlet case. Take a derivative of (2.4.1) and evaluate it at the boundary points to get

$$\partial_t u_x(x_0, t) = \nu \partial_x^3 u(x_0, t).$$

Hence,

$$u_0'''(t) = \frac{1}{\nu}\partial_t u_b(t) \tag{2.4.23}$$

[19]See exercise 2.4.3 and 2.4.4

To further eliminate the other unknown quantities, we will need the Taylor expansions of u, centered at x_0, at the two interior grid points x_1 and x_2

$$u_1 = u_0 + \ (hu_0') + \ (\frac{h^2}{2!}u_0'') + \ (\frac{h^3}{3!}u_0^{(3)}) + \ (\frac{h^4}{4!}u_0^{(4)}) + O(h^5), \tag{2.4.24}$$

$$u_2 = u_0 + 2(hu_0') + 2^2(\frac{h^2}{2!}u_0'') + 2^3(\frac{h^3}{3!}u_0^{(3)}) + 2^4(\frac{h^4}{4!}u_0^{(4)}) + O(h^5). \tag{2.4.25}$$

If we subtract (2.4.24) from (2.4.21), and use (2.4.23) we get:

$$u_{-1} = u_1 - 2hu_b - \frac{h^3}{3\nu}u_b' \tag{2.4.26}$$

Similarly, subtracting (2.4.25) form (2.4.22) gives

$$u_{-2} = u_1 - 4hu_b - \frac{8h^3}{3\nu}u_b'. \tag{2.4.27}$$

Compact Fourth Order Approximation: An Alternative

Using a one-sided formula such as (2.4.15) at the "ghost" point for the long stencil approach can be avoided if one uses a compact approach to discretizing the spatial operators.. The main disadvantage(and it is an important one!) is that one must now solve a Poisson-like equation at each time step. Applying the 4th order compact approximation (2.3.22) to (2.4.1), we have

$$\begin{cases} \partial_t \bar{u} = \nu(\Delta_h + \frac{h^2}{6}D_x^2 D_y^2)u \\ (1 + \frac{h^2}{12}\Delta_h)u = \bar{u} \end{cases} \tag{2.4.28}$$

Using the energy estimates, it is easy to show that the above scheme is indeed stable and obtains full 4^{th} order accuracy[20]

The above system (2.4.28) can be solved quite efficiently if an explicit time stepping dicretization is used for (2.4.1). In this case we don't need to know the boundary values of $\bar{u}$! This makes the scheme quite efficient.

Exercise:

[20] See exercise 2.4.7

1. Use energy estimates to show that the second order scheme (2.4.2) has full 2^{nd} order accuracy for both the Dirichlet and Neumann problems.

2. Construct an exact smooth solution for (2.4.1) by adding a force term to the right side (2.4.1). Write a code for the second order scheme, and use your constructed exact smooth solution to check the accuracy.

3. As in the previous exercise, write a code for the 4^{th} order long stencil scheme (2.4.11) and (2.4.15) and check the accuracy.

4. Derive a 3^{rd} order one-sided formula in a similar way that (2.4.15) was derived. Then replace (2.4.15) in the 4th order long stencil scheme (2.4.11) by your new 3^{rd} order approximation. Numerically check the accuracy of this new scheme. Does this new scheme maintain 4^th order accuracy.

5. Give an error estimate for the 4th order scheme (2.4.11)-(2.4.15).

6. Perform a stability analysis for the 4th order scheme (2.4.11) using the boundary conditions (2.4.26) and (2.4.27) for the Neumann problem.

7. Use energy estimates to show that the 4^{th} order compact scheme (2.4.28) is indeed a stable scheme.

§2.5. Convection Diffusion Equation in an Incompressible Velocity Field

The time evolution of many physical quantities, such as density or temperature(represented by θ), can be described by a convection diffusion equation

$$\partial_t\theta + (\boldsymbol{u}\cdot\nabla)\theta = \nu\Delta\theta\,, \tag{2.5.1}$$

where $\boldsymbol{u}$ is the velocity field. We will assume that $\boldsymbol{u}$ is smooth, and satisfies the incompressibility condition

$$\text{div } \boldsymbol{u} = 0, \tag{2.5.2}$$

as well as the no-slip boundary condition

$$\boldsymbol{u} \mid_\Gamma = 0. \tag{2.5.3}$$

Coupling between θ and $\boldsymbol{u}$ usually appears in momentum equation. For instance, this coupling may occur in the form of a gravity effect. But from a computational point of view, we can treat θ as a passive scalar since the coupling in the momentum equation can easily be dealt with using explicit time discretizations for the terms involving θ. As usual, two types of boundary conditions may be prescribed for θ. The Dirichlet boundary condition:

$$\theta \mid_\Gamma = \theta_b\,, \tag{2.5.4}$$

where θ_b is a known function, or the Neumann boundary condition:

$$\frac{\partial\theta}{\partial\boldsymbol{n}} = 0\,, \qquad \text{on } \Gamma\,. \tag{2.5.5}$$

The numerical approximation of (2.5.1) is very similar to that of the heat equation, which was discussed in §2.4. In the case of a second order spatial approximation, we can use standard centered difference methods to approximate the convection term.

For the case of the Dirichlet boundary condition, all of the unknowns occur at the interior grid points since the boundary values are equal to the given boundary data θ_b. The 4th order long-stencil approximation gives,

$$\partial_t\theta + u\tilde{D}_x(1-\frac{h^2}{6}D_x^2)\theta + v\tilde{D}_y(1-\frac{h^2}{6}D_y^2)\theta = \nu\Big(D_x^2(1-\frac{h^2}{12}D_x^2) + D_y^2(1-\frac{h^2}{12}D_y^2)\Big)\theta\,. \tag{2.5.6}$$

We need a "ghost value"for θ_{-1}. The derivation is very similar to that for the heat equation. When the non-slip boundary condition (2.5.3) for the velocity $\boldsymbol{u}$ holds, the resulting one-sided formula is indeed exactly same as in (2.4.15),

$$\theta_{i,-1} = \frac{20}{11}\theta_{i,0} - \frac{6}{11}\theta_{i,1} - \frac{4}{11}\theta_{i,2} + \frac{1}{11}\theta_{i,3} + \frac{12}{11}h^2\left(\frac{1}{\nu}\partial_t\theta_b - \partial_x^2\theta_b\right) + O(h^5)\,. \tag{2.5.7}$$

The stability analysis is also similar to that of heat equation, but the convection term will make the analysis more complicated[21].

In the case of Neumann boundary condition, the unknowns occur not only at the interior grid points, but also at the boundary grid points. If we use a second order scheme to discretize (2.5.1), we need one "ghost" point, whose value can be obtained from

$$\frac{\theta_{-1,j} - \theta_{1,j}}{2h} = u_b(y_j) \tag{2.5.8}$$

(outward-normal direction).

For the 4^{th} order long stencil approximation, we will need approximations for the value of θ at two "ghost points", θ_{-1} and θ_{-2}. In the case of the heat equation we were able to use information about u_0''' directly form the PDe to reduce the number of points needed for our one-sided approximations. In the case of the convection-diffusion equation we can perform a similar trick, however it will require a bit more explanation. For now, Let proceed naively.

We now derive a 3^{rd} order one-sided formula. Just as was done in §2.4 for u, we first expand θ_{-1} in a Taylor series centered at x_0:

$$\theta_{-1} = \theta_0 - (h\theta_0') + (\frac{h^2}{2!}\theta_0'') - (\frac{h^3}{3!}\theta_0''') + O(h^4)$$

Since the quantity θ_0' is known from the boundary data, we will need the value of θ at three interior points to eliminate the other three unknowns. We omit the details. The resulting one-sided formula is given by:

$$\theta_{-1} = \frac{1}{11}(6\theta_1 + 8\theta_2 - 3\theta_3 - 24hu_b). \tag{2.5.9}$$

[21]See exercise 2.4.3 and 2.4.4

Similarly, for the other "ghost point", we have

$$\theta_{-2} = \frac{1}{11}(-40\theta_1 + 75\theta_2 - 24\theta_3 - 60hu_b). \tag{2.5.10}$$

We will leave the derivation of the 4th order Neumann boundary condition as an exercise[22].

Now we will briefly outline a different derivation for the values of θ at the "ghost points". Recall that in the case of the heat equation (2.4.1), we differentiated the PDE to obtain a value for $u'''(x_0)$. See (2.4.23). For the convection-diffusion equation (2.5.1) we proceed in a similar fashion. However, we will need a minor modification. First, take the derivative in x of (2.5.1) and evaluate it at the boundary $x = 0$. (Note: $u = v = 0$ along the boundary due to the no-slip condition). Since $\theta_x = \theta_b$ on the boundary, we have

$$\partial_t(\theta_b) + u_x\theta_b + v_x\partial_y\theta = \nu(\partial_x^3\theta + \partial_y^2(\theta_b)) \tag{2.5.10a}$$

The only term that presents a possible problem is the evaluation of $\partial_y\theta$ along the boundary $x = 0$. But we can simply approximate $\partial_y\theta$ by using a 4th order centered difference approximation along the $x = 0$ boundary. We leave the details to you!

Compact Fourth Order Approximation.

We can use a compact approach to avoid one-sided formulas derived using exterior "ghost" point. The disadvantage of compact schemes is now one has to solve a Poisson-like equation at each time step. This approach is not recommended for the passive scalar problem since the long stencil method is adequate. However, in the active scalar equation, such as one involving a vorticity transport equation, the compact approach is highly recommenced as we will see in next section.

What is new here compared to the heat equation is the compact treatment of the convection term. Let us first rewrite (2.5.1) in the form

$$\partial_t\theta = \nu\Delta\theta - f, \qquad f = \nabla\cdot(\boldsymbol{u}\theta). \tag{2.5.11}$$

Applying the 4th order compact approximation (2.3.22), we have

$$\partial_t\bar{\theta} = \nu(\Delta_h + \frac{h^2}{6}D_x^2D_y^2)\theta - (1 + \frac{h^2}{12}\Delta_h)f, \tag{2.5.12}$$

[22]see exercise 2.4.5

where

$$\bar{\theta} = (1 + \frac{h^2}{12}\Delta_h)\theta. \tag{2.5.13}$$

Now we derive an essentially compact approximation for the convection term, which is the last term in (2.5.12). Using a 4th order long stencil to approximate both derivatives in $\nabla \cdot (\boldsymbol{u}\theta)$, the last term of (2.5.12), which we denote by I, becomes

$$I = (1 + \frac{h^2}{12}\Delta_h)\left(\tilde{D}_x(1 - \frac{h^2}{6}D_x^2)(u\theta) + \tilde{D}_y(1 - \frac{h^2}{6}D_y^2)(v\theta)\right) + O(h^4). \tag{2.5.14}$$

Simple algebra leads to

$$I = \tilde{D}_x(1 + \frac{h^2}{6}D_y^2)(u\theta) + \tilde{D}_y(1 + \frac{h^2}{6}D_x^2)(v\theta)) - \frac{h^2}{12}\Delta_h(u\tilde{D}_x\theta + v\tilde{D}_y\theta) + O(h^4) \tag{2.5.15}$$

Notice that $u\tilde{D}_x\theta + v\tilde{D}_y\theta$ vanishes at the boundary due to the no-slip boundary condition. Hence we don't need any additional numerical boundary conditions for this term. We can summarize the above as

$$\begin{cases} \partial_t\bar{\theta} = \nu(\Delta_h + \frac{h^2}{6}D_x^2D_y^2)\theta - \tilde{D}_x(1 + \frac{h^2}{6}D_y^2)(u\theta) - \tilde{D}_y(1 + \frac{h^2}{6}D_x^2)(v\theta)) \\ \qquad\qquad + \frac{h^2}{12}\Delta_h(u\tilde{D}_x\theta + v\tilde{D}_y\theta) \\ (1 + \frac{h^2}{12}\Delta_h)\theta = \bar{\theta} \end{cases} \tag{2.5.16}$$

Using energy estimates, it is easy to show that the above scheme is indeed stable, and achieves full 4th order accuracy[23]

The above system (2.5.16) can be solved quite efficiently when one uses any of the high order explicit time stepping schemes discussed in §2.1. This is the case as long as we know the boundary values for θ. Notice that we don't need to know the boundary values of $\bar{\theta}$!

Exercise:

1. Assume the velocity field $\boldsymbol{u}$ in (2.5.1) is smooth. Use energy estimates to show that the 2nd scheme applied to (2.5.1) achieves full accuracy.

[23]See exercise 2.4.6

2. Construct a smooth solution for (2.5.1) by adding a force term to the right side (2.5.1). Write a code that implements the 2^{nd} order scheme. Use your constructed smooth solution to check the accuracy.

3. As in the previous exercise, write a code for the 4^{th} order long stencil scheme (2.5.11) and (2.5.17) and check the accuracy.

4. Derive a 3^{rd} order one-sided formula in a similar way that (2.5.17) was derived. Then replace (2.5.17) in the 4th order long stencil scheme (2.5.11) by your new 3^{rd} order approximation. Numerically check the accuracy of this new scheme. Does this new scheme maintain 4^th order accuracy.

5. Use energy estimates to show that (2.5.23) is indeed a stable scheme.