13

Refining the Numerical Methods

The matrix method of numerical stability analysis can become cumbersome when working with large datasets or when very fine resolution is needed. The reason is that the processor time needed to find the eigenvalues of an $N \times N$ matrix increases rapidly with increasing N. Here, we'll explore a few more-refined methods aimed at improving resolution without increasing processing time.

13.1 Higher-Order Finite Differences

The most obvious refinement of our second-order matrix method is to increase the accuracy of the derivative matrices. In section 1.4.3, you derived the second-order, second-derivative matrix with Dirichlet boundary conditions $f_0 = f_{N+1} = 0$:

$$\mathsf{D}^{(2)} = \frac{1}{\Delta^2}\begin{bmatrix} -2 & 1 & 0 & 0 & \dots \\ 1 & -2 & 1 & 0 & \dots \\ & & \ddots & & \\ \dots & 0 & 1 & -2 & 1 \\ \dots & & & 1 & -2 \end{bmatrix} \tag{13.1}$$

If we repeat the derivation of this matrix, keeping two more terms in the Taylor series approximations (e.g., 1.7), the result is accurate to fourth order in Δ:

$$\mathsf{D}^{(2)} = \frac{1}{12\Delta^2}\begin{bmatrix} -15 & -4 & 14 & -6 & 1 & 0 & \dots \\ 16 & -30 & 16 & -1 & 0 & 0 & \dots \\ -1 & 16 & -30 & 16 & -1 & 0 & \dots \\ & & & \ddots & & & \dots \\ \dots & 0 & -1 & 16 & -30 & 16 & -1 \\ \dots & 0 & 0 & -1 & 16 & -30 & 16 \\ \dots & 0 & 1 & -6 & 14 & -4 & -15 \end{bmatrix} \tag{13.2}$$

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

In principle, one can continue from here to calculate finite difference approximations of arbitrarily high order, but the resulting formulae become very complicated and results can be contaminated by roundoff error.

It is important to recognize that higher-order approximations are designed to be more accurate *in the limit as* $\Delta \to 0$. In practice, we make Δ as small as we can with the available resources, but we often find ourselves using the method when Δ is not as small as we would like (i.e., compared with the flow features we are interested in). In that case, *there is no guarantee that a higher-order method will be more accurate.*

How do these methods compare when used for stability analysis? Consider the Rayleigh equation (3.16, 3.17), reproduced here for convenience:

$$\sigma \nabla^2 \hat{w} \;=\; -\imath k U \nabla^2 \hat{w} \;+\; \imath k \frac{d^2 U}{dz^2}\, \hat{w}\,; \qquad \nabla^2 = \frac{d^2}{dz^2} - \tilde{k}^2. \tag{13.3}$$

We choose $U = \tanh z$ and impose impermeable boundary conditions at $z_1 = -4$ and $z_2 = +4$. The fastest-growing instability is two-dimensional, with wavenumber $k = \tilde{k} = 0.47$, and has (real) growth rate $\sigma = 0.1768316166$. (This very precise value was determined using the shooting method to be described later in section 13.4.) The solution method is exactly as described in section 3.5, except that the second-derivative matrix is now the fourth-order version, (13.2).

In some applications, our aim is to approximate the solution of (13.3) as accurately as possible, while in others we try to attain acceptable accuracy while minimizing processing time and memory. Therefore, we will compare methods based on two criteria:

- How quickly does the error go to zero as $N \to \infty$?
- What is the smallest N needed to achieve 1 percent accuracy?

Figure 13.1 shows the relative error in the growth rate for the second- and fourth-order finite difference methods. The number of grid points ranges from 9 to 513 (illustrated in Figure 13.1a). Not surprisingly, both methods become more accurate as N is increased. At large N, the error in the second-order method (blue) decreases like N^{-2}, while the error in the fourth-order method is proportional to N^{-4}. At low N, however, the order makes much less difference.

To achieve 1 percent accuracy in the growth rate requires $N = 23$ with the second-order method and only $N = 18$ with the fourth-order method. This may not seem like much of a difference, but the time needed to compute the eigenvalues is reduced by 40 percent.[1] In the analysis of large geophysical datasets, the computation must be repeated thousands of times, taking several weeks (on a

[1] The time needed to calculate eigenvalues in Matlab is approximately proportional to N^2, and $(18/23)^2 = 0.6$.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

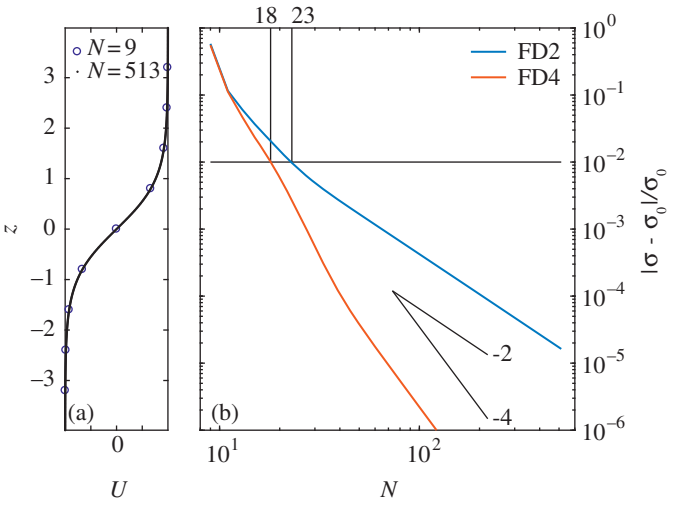

Figure 13.1 (a) Velocity profile $U = \tanh z$ spanned by 9 and 513 grid points. (b) Relative error in the growth rate for second- (blue) and fourth-order (red) finite difference approximations. The problem is specified by the Rayleigh equation (3.19), $U = \tanh z$, $k = 0.47$, and $\hat{w}(-4) = \hat{w}(4) = 0$.

2016-vintage workstation). Therefore, the minor task of upgrading the derivative matrices to fourth order is well justified.

13.2 Finite Differences on an Adaptive Grid

It stands to reason that accuracy could be improved efficiently by focusing resolution where it is needed most, i.e., by letting the grid spacing Δ vary with z such that the spacing is fine in regions where the solution varies most rapidly. In a shear layer, for example, one might make Δ smaller at the center of the layer and larger outside the layer. To do this requires two things: derivative matrices that allow for non-uniform Δ, and a plan for distributing the grid points efficiently.

The first requirement is straightforward. The derivation in section 1.4.2 is easily generalized to allow for variable Δ, though the algebra is a bit more complicated. The second-derivative becomes

$$\tilde{f}_i'' = \frac{2f_{i-1}}{\Delta_{i-1}(\Delta_{i-1}+\Delta_i)} - \frac{2f_i}{\Delta_{i-1}\Delta_i} + \frac{2f_{i+1}}{\Delta_i(\Delta_{i-1}+\Delta_i)}, \tag{13.4}$$

where

$$\Delta_i = z_{i+1} - z_i. \tag{13.5}$$

Check that, if the Δ_i are all the same, this reduces to the familiar approximation for the second-derivative that you derived in homework problem 2.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

Incorporation of the boundary conditions is straightforward. In the case $f_0 = f_{N+1} = 0$, the result is

$$\tilde{f}_1'' = -\frac{2f_1}{\Delta_0\Delta_1} + \frac{2f_2}{\Delta_1(\Delta_0+\Delta_1)}; \quad \tilde{f}_N'' = \frac{2f_{N-1}}{\Delta_{N-1}(\Delta_{N-1}+\Delta_N)} - \frac{2f_N}{\Delta_{N-1}\Delta_N}. \tag{13.6}$$

Now, how shall we distribute the grid points? There is no "right" answer to this; we are required to guess in advance where the solution will vary most rapidly. Let's suppose that this will happen in regions where the shear is strongest.

We begin with a set of points spaced evenly between the boundaries z_B and z_T:

$$z_i = z_B + i\Delta, \quad \text{where } i = 0, 1, 2, \ldots, N+1$$

and

$$\Delta = \frac{z_T - z_B}{N-1} > 0.$$

Note this list *includes* the top and bottom points: $z_0 = z_B$ and $z_{N+1} = z_T$.

We now define the absolute shear

$$s_i = \frac{|U_{i+1} - U_i|}{z_{i+1} - z_i}, \quad \text{for } i = 0, 1, 2, \ldots, N \tag{13.7}$$

and a transformed version

$$\zeta_i = \frac{\ln(s_i/s_{min})}{\ln(s_{max}/s_{min})}. \tag{13.8}$$

The latter variable approaches 0 and 1 when the absolute shear is at its minimum and maximum values, s_{min} and s_{max}, respectively.[2] We now define the new grid increment

$$\delta_i = \frac{z_T - z_B}{\sum_{j=0}^{N}(1 - a\zeta_j)}(1 - a\zeta_i), \quad i = 0, 1, 2, \ldots, N,$$

where a is a constant such that $0 \le a < 1$. If $a = 0$, $\delta_i = \Delta$, i.e., the original uniform grid is recovered (cf. 13.2). As $a \to 1$, the stretching becomes extreme.

The new grid points are

$$\xi_0 = z_B; \quad \xi_j = z_B + \sum_{i=0}^{j-1}\delta_i; \quad j = 1, 2, \ldots, N+1.$$

Results for the hyperbolic tangent shear layer are shown in Figure 13.2. With $a = 0$, the second-order result from Figure 13.1 is reproduced (blue curve). With

[2] When coding this, it is advisable to increment s_{min} with a tiny value, e.g., 10^{-16}, to avoid the possibility of dividing by zero.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

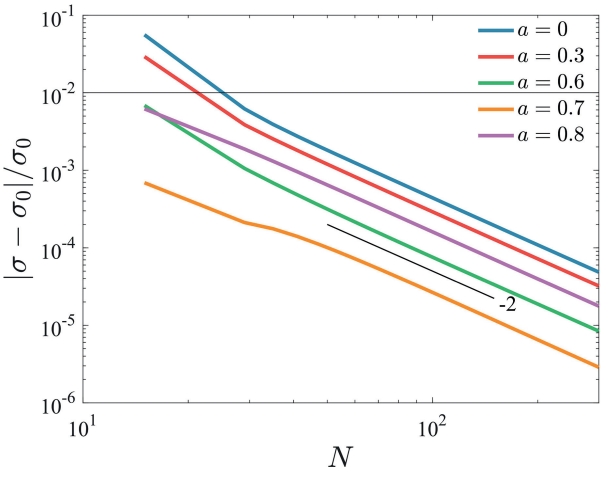

Figure 13.2 Relative error in the growth rate for second-order finite difference approximations using various degrees of grid stretching. The problem is specified by the Rayleigh equation (3.19), $U^\star = \tanh z^\star$, $k^\star = 0.47$, and $\hat{w}(-4) = \hat{w}(4) = 0$. Calculation courtesy of Qiang Lian.

nonzero a, the error decreases, the most accurate choice being $a = 0.7$ (yellow curve). Increasing a further leads to reduced accuracy as resolution becomes inadequate at the edges of the shear layer. As $N \to \infty$, the error decreases as N^{-2} in all cases.[3]

Note that our choice of the grid adaptation algorithm (13.7, 13.8) is based on the assumption that shear is the primary factor determining the resolution requirement at a given location. In more general situations where properties other than shear become important, some creative trial-and-error may be called for.

13.3 Galerkin Methods

Finite difference methods are *local*, in the sense that the derivative at each point is approximated using nearby points only. In a *global* method, derivatives are approximated using functions that span the entire domain. Global methods are intrinsically more complicated, but the extra effort is often repaid in fast, accurate results. An example is the Galerkin methods, two of which are described in the next two subsections.

In the Galerkin technique, we expand the solution in terms of a set of $\mathcal{N}$ orthogonal basis functions, i.e.,

[3] This result is somewhat surprising because (13.4) is formally accurate only to first-order. See Ferziger and Peric (1999) for further discussion.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

$$\hat{w}(z) = \sum_{n=1}^{\mathcal{N}} w_n F_n(z). \tag{13.9}$$

If the basis functions F_n are chosen properly,[4] the result becomes exact in the limit $\mathcal{N} \to \infty$. A common example is the Fourier sine series:

$$F_n(z) = \sin \frac{n\pi}{H} z\,; \quad n = 1, 2, \cdots \mathcal{N}. \tag{13.10}$$

In practice, we choose the largest value of $\mathcal{N}$ that our resources allow and hope that the result will be accurate enough for the purpose. The examples given below show that this approach can be spectacularly successful if $\mathcal{N}$ is sufficiently large.

In general the basis functions $F_n(z)$ must satisfy three criteria.

(i) Each basis function must obey the boundary conditions. For example, the set (13.10) obeys the conditions $\hat{w} = 0$ at $z = 0$ and $z = H$.

(ii) The basis functions must obey an *orthogonality relation*, meaning

$$\int_{z_1}^{z_2} W(z) F_m(z) F_n(z) dz = \delta_{mn}, \tag{13.11}$$

where $z = z_1$ and $z = z_2$ are the boundaries (which may be at infinity) and $W(z)$ is a weighting function. In the Fourier example (13.10), $z_1 = 0$, $z_2 = H$, and the weighting function is a constant, $W = 2/H$.

(iii) The basis functions must form a *complete set* with respect to the chosen boundary conditions, meaning that any smooth function satisfying the boundary conditions can be approximated to arbitrary precision by making $\mathcal{N}$ sufficiently large.

There are many alternative choices, e.g., cosine series, Hermite polynomials, Chebyshev functions, ..., which can be found in most applied math references (e.g., Spiegel, 1968). One may also invent basis functions for a particular problem. An example is the *hydrostatic normal modes* for a particular buoyancy profile (Gill, 1982; McWilliams, 2006).

Having chosen our basis functions, we substitute (13.9) into the equation we want to solve. Finally, we multiply through by $W(z)F_m(z)$ and integrate over the domain. If we've done everything right, the result is an algebraic eigenvalue problem for the coefficients w_m.

[4] Specifically, the basis functions must form a *complete set* over the space of functions satisfying the boundary conditions.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

As an example, we'll solve the Rayleigh equation (13.3) subject to $\hat{w}(0) = \hat{w}(H) = 0$ using the Fourier basis functions (13.10). This is called the Fourier-Galerkin (or just Fourier) method. Substituting (13.9) into (13.3) gives

$$\sigma \sum_{n=1}^{\mathcal{N}} w_n \nabla^2 F_n \;=\; -\imath k U \sum_{n=1}^{\mathcal{N}} w_n \nabla^2 F_n \;+\; \imath k U'' \sum_{n=1}^{\mathcal{N}} w_n F_n. \tag{13.12}$$

where primes indicate differentiation with respect to z. For the Fourier basis functions (13.10), we can simplify using the fact that $F_n'' = -(n\pi/H)^2 F_n$, and therefore the Laplacian $\nabla^2 F_n$ becomes $D_n F_n$, where

$$D_n = -\left(\frac{n\pi}{H}\right)^2 - \tilde{k}^2. \tag{13.13}$$

We now multiply through by $(2/H)F_m$ and integrate, remembering (13.11):

$$\sigma \sum_{n=1}^{\mathcal{N}} w_n D_n \underbrace{\frac{2}{H}\int_0^H F_m F_n dz}_{=\delta_{mn}} = -\imath k \sum_{n=1}^{\mathcal{N}} w_n D_n \frac{2}{H}\int_0^H F_m U F_n dz + \imath k \sum_{n=1}^{\mathcal{N}} w_n \frac{2}{H}\int_0^H F_m U'' F_n dz.$$

The summation on the left-hand side can be done explicitly, resulting in $w_m D_m$. Dividing through by D_m then gives the algebraic eigenvalue problem:

$$\sigma w_m = A_{mn} w_n, \tag{13.14}$$

where

$$A_{mn} \;=\; -\imath k \frac{D_n}{D_m}\int_0^H F_m U F_n dz \;+\; \imath k \frac{1}{D_m}\int_0^H F_m U'' F_n dz. \tag{13.15}$$

As in previous methods, the eigenvalue is the growth rate σ, but the eigenvector is now composed of the coefficients w_n in the expansion (13.9). When coding the Galerkin method, note that the two integrals appearing in (13.15) are independent of the wavenumber. When looping over many values of the wavenumber, those integrals only have to be computed once.

We have some freedom in choosing the number of basis functions, $\mathcal{N}$. In the Fourier case, the smallest wavelength resolvable with grid spacing Δ is 2Δ (the Nyquist wavelength, e.g., Harris and Stocker, 1998), so $\mathcal{N}$ should be at least $N/2$. Here, we choose $\mathcal{N} = N$.

In Figure 13.3, we compare the Fourier-Galerkin technique with the second- and fourth-order finite difference techniques for two idealized velocity profiles. The first case shown is the hyperbolic tangent shear layer (Figure 13.3a). The finite

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

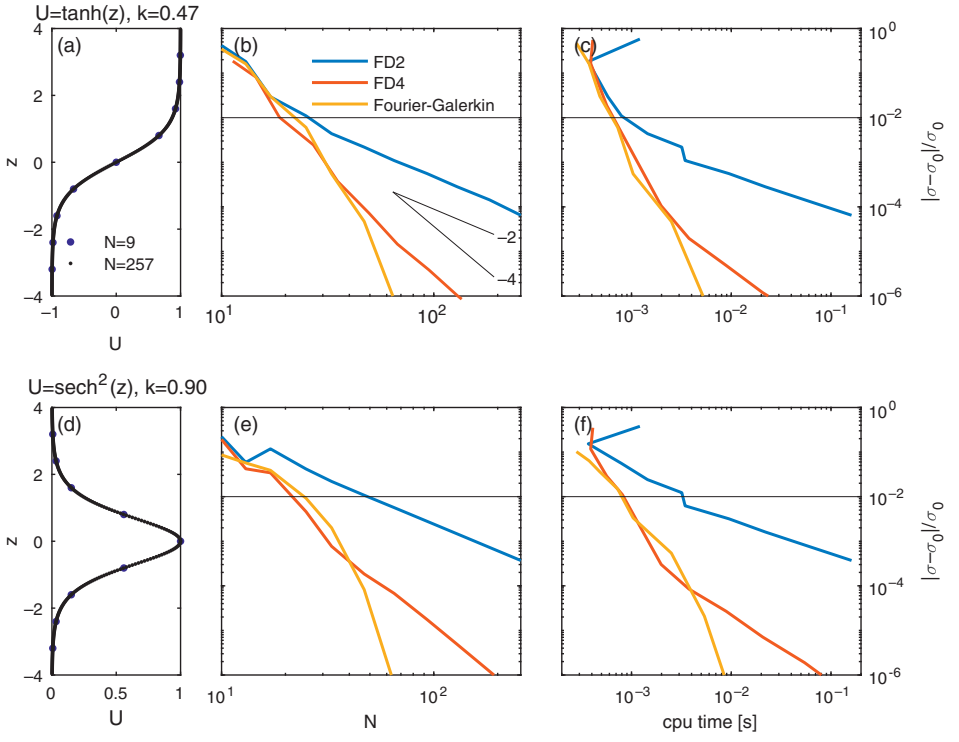

Figure 13.3 Comparisons of Fourier-Galerkin and finite difference solutions of the Rayleigh equation for the hyperbolic tangent shear layer with $k = 0.47$ (a,b) and the Bickley jet with $k = 0.9$ (c,d). Left panels show the velocity profile with the minimum and maximum grid spacings tested. Middle panels show the relative error in the growth rate versus the number of grid points. The right panels show relative error versus processing time (numerical values are specific to the computer used, but relative values should be general). The "exact" growth rate σ_0 is computed using a high-precision shooting method (section 13.4).

difference results (blue and red curves on Figure 13.3b) reproduce Figure 13.1. The Fourier-Galerkin result is shown in yellow.

For N greater than about 40, the Fourier method converges much more rapidly than the finite difference methods. Based on this, the Fourier method is preferred if we seek the most accurate results possible and computation time is not a consideration. If we only need 1 percent accuracy, the required number of grid points is intermediate between the two finite difference methods. The second case tested is the Bickley jet (Figure 13.3d,e,f). The resolution requirement is somewhat more stringent than in the tanh case because the profile is more sharply curved. The 1 percent error tolerance requires $N = 46$ for the second-order method, about $N = 20$ for either the fourth-order or the Fourier method. Either of the more sophisticated methods reduces the computation time by about a factor of four. In both cases,

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

the extra coding required to implement the Fourier method is justified only if one requires relative errors less than about 10^{-4}.

13.4 The Shooting Method

The shooting method is fundamentally different from the matrix methods considered so far. Its main advantage is precision: it can deliver a much more accurate result for a given amount of processing time. The main disadvantage is reliability: there is no guarantee that all modes, or even the fastest-growing mode, will be found. Considerable ingenuity is needed to manage this problem, and the resulting codes tend to be quite complicated. The latter issue is the reason we focus on matrix methods in this introductory-level book.

Aptly named, the shooting method is analogous to an aiming method used by artillery gunners to hit a distant target. They begin with an initial "test" shot and note how much they miss by. With this information, they adjust their aim and try again, repeating the process until the target is hit.

To solve a differential eigenvalue problem via the shooting method, you begin with an initial guess for the eigenvalue. You then use the boundary conditions to specify the solution at one boundary (call it the near boundary), and integrate the differential equation from there to the other (far) boundary. In general, the boundary condition at the far boundary will not be satisfied, but the size of the mismatch tells you how to adjust your estimate of the eigenvalue. You then try again, repeating the procedure until the far boundary condition is satisfied.

13.4.1 A Simple Illustration

Consider the Rayleigh equation for 2D modes:

$$\hat{w}_{zz} = \left(\frac{U_{zz}}{U - c} + k^2 \right) \hat{w}, \tag{13.16}$$

where the subscript z indicates differentiation. We choose $U = \tanh z$, $k = 0.47$, and impermeable boundary conditions at $z_1 = -4$ and $z_2 = +4$. It remains only to find the value (or values) of c that permit satisfaction of both boundary conditions. The results shown in Figure 13.4 were obtained via the following sequence of steps.

(i) Start with an initial guess for c, which we'll call c_1. Cheating shamelessly, we use results from section 3.9.1 to "assume" that c is purely imaginary, and choose $c = 0.44\iota$.

(ii) Begin the integration at the near boundary, $z_1 = -4$. Because the differential equation is second order, we must specify both $\hat{w}(z_1)$ and $\hat{w}_z(z_1)$, in accordance with the Dirichlet boundary conditions, $\hat{w}(z_1) = 0$.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

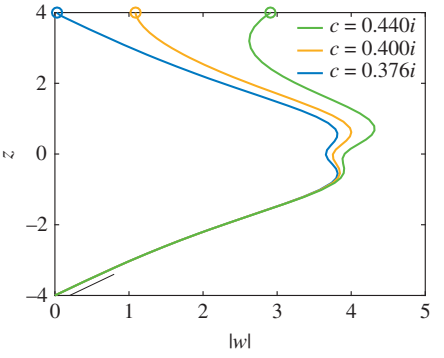

Figure 13.4 Three example integrations of (13.16), with $U = \tanh z$ and $k = 0.47$ and different estimates of c. For simplicity we assume that c is purely imaginary. Integration begins at the lower boundary with $\hat{w} = 0$ and $\hat{w}_z = 1$ (indicated by short black line). Successive attempts converge to a solution that satisfies the upper boundary condition.

(iii) How do we specify $\hat{w}_z(z_1)$? We know that it can't be zero, or the integration would yield the null solution $\hat{w}(z) = 0$. What value should we use? It actually doesn't matter. Because the equation is homogeneous, the solution is defined only up to a multiplicative constant. Therefore, set $\hat{w}_z(z_1)$ to some arbitrary (nonzero) value, e.g., $\hat{w}_z(z_1) = 1$.

(iv) Integrate the equation from z_1 to z_2 using an initial value solver such as Matlab's ode45. The result is the green curve on Figure 13.4.

(v) Unless we're very lucky, the value of $\hat{w}$ at the far boundary z_2 will not be zero. We'll call that value

$$M(c_1) = \hat{w}(z_2). \tag{13.17}$$

In this case $M(c_1) = 2.9$ (green circle).

(vi) Now try again with a different value of c, say c_2, integrate again, and calculate $M(c_2)$. Here we choose $c_2 = 0.40\imath$, giving us the yellow curve and $M(c_2) = 1.1$. We're headed in the right direction!

(vii) Based on this information, continue refining the guess until $M = 0$ to within some predefined tolerance. For the example shown, trial and error leads us to the final choice $c = 0.376\imath$, which gives a value of M very close to zero (blue curve).

13.4.2 The General Method

For the example above, we allowed ourselves the huge advantage of assuming in advance that c is purely imaginary. In general, c is complex, and so is $M(c)$. A

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

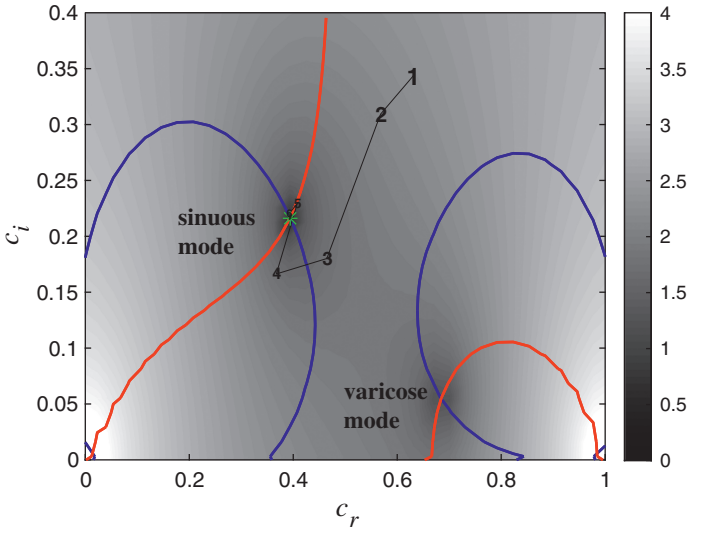

Figure 13.5 The function $M(c)$ for the Bickley jet profile $U = \text{sech}^2 z$ with $k = 0.7$ and $\hat{w} = 0$ at $z = \pm 5$. Shading indicates the absolute value $|M|$. The contours $M_r = 0$ and $M_i = 0$ are shown in blue and red, respectively. Numerals indicate a sequence of guesses for the eigenvalue c leading to $c = 0.3932 + 0.2164\iota$ (green asterisk).

solution is valid only if both the real and imaginary parts of M are zero. The most straightforward approach is to make a contour plot such as that shown in Figure 13.5. In this case we have used the Bickley jet profile $U = \text{sech}^2 z$ with impermeable boundaries at $z = \pm 5$. The wavenumber k is set to 0.7.

The complex function M is represented in two ways. Gray shading indicates $|M|$, with dark regions showing near-zero values. The real part of M is zero on the blue contour; the imaginary part is zero on the red contour. The requirement $M = 0$ is satisfied wherever the red and blue contours cross.

There are two such intersections, corresponding to the sinuous and varicose modes of the Bickley jet (see section 3.9.2). If you have time, you can simply read the values of c_r and c_i from the graph. It is usually preferable to automate that procedure, though. The simplest method is linear extrapolation:

(i) Make two initial guesses. In this case, we have chosen $c_1 = 0.6311 + 0.3437\iota$ and $c_2 = 0.5680 + 0.3094\iota$, shown on Figure 13.5 by the numerals 1 and 2. The first value was chosen at random; the second is 0.9 times the first.

(ii) Solve the equation for each of these values and denote the results M_1 and M_2.

(iii) Compute the coefficients of a linear function passing through (c_1, M_1) and (c_2, M_2), then find the value of c at which the linear function is zero:

$$c = c_1 - \frac{c_2 - c_1}{M_2 - M_1} M_1. \tag{13.18}$$

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

Use this as the next guess. The result is shown by the numeral 3 on Figure 13.5.

(iv) Iterate the procedure, each time using the most recent two results to define the linear function, until $|M|$ converges. In this case the convergence criterion was

$$\left| \frac{M_2 - M_1}{M_1} \right| < 10^{-6}. \tag{13.19}$$

This criterion was satisfied at the point shown by the green asterisk, which corresponds to the sinuous mode.[5]

(v) To find the varicose mode, we would repeat this process until a distinct, second mode was identified. (The varicose mode is visible on Figure 13.5 as the intersection of the red and blue curves near $c = 0.68 + 0.05\iota$.)

The shooting method is an example of *nonlinear root-finding*, a notoriously difficult class of problems. Everything depends on the initial guess, and there is usually no way to guarantee that all roots have been found. For the complicated background flows that one encounters in nature, the function $M(c)$ can be much more complicated than that shown in Figure 13.5. The absolute value $|M|$ can have numerous peaks with valleys meandering fractally around them. Occasionally, the bottom of one of those valleys will touch zero, and there lies an eigenvalue. In this case, success depends entirely on the accuracy of the initial guess. It may be necessary to try many initial guesses to be reasonably confident of finding all unstable modes, and even then we cannot be certain that every mode, or even the fastest-growing mode, has been found.[6]

A few fine points:

- It helps to put some thought into the initial guesses. In the example above, our two initial guesses were guided by Howard's semicircle theorem (section 4.8).

[5] To compute the highly accurate results used earlier to test the various matrix methods, this convergence value was reduced to 10^{-12}.

[6] A blind man sets off to climb Mount Everest. "But how," a friend asks him, "will you find your way?" "Easy", he replies. "The summit of Mount Everest is the highest place on Earth, so I'll just keep walking uphill until I get there!"
The flaw in this plan, of course, is that walking continually uphill will get him to the top of the nearest hill, which is probably not Mount Everest. The shooting method works in the same way: you start with an initial guess for the eigenvalue, then use some algorithm to refine it until it attains some chosen level of accuracy. You might end up at the fastest-growing mode, but you might end up at some other mode, or even no mode at all – it all depends on the accuracy of your initial guess.
Now suppose that, instead of one blind man, 1,000 blind men set off to climb Mount Everest, each starting from some random location on Earth's surface. The odds of one of them reaching the goal would be improved (though still not very good). This is what we do with the shooting method. We make many initial guesses, refine each one, pick the ones most relevant to our problem (e.g., the largest growth rate), and hope for the best.

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

Because $U(z)$ varies between 0 and 1, unstable modes must have $0 < c_r < 1$ and $0 < c_i < 0.5$ or, more precisely, $(c_r - 1/2)^2 + c_i^2 < 1/4;\ c_i > 0$.

- The linear extrapolation described in step 3 can be improved by raising the order: fit a quadratic function to the most recent three estimates and find the nearest root (there will be two).
- A common problem is that several initial guesses will converge to the same root – a waste of computer time. This can be avoided by redefining the function M. Once a particular root has been found, say $c^{(1)}$, redefine $M(c)$ as $M(c)/|c-c^{(1)}|$. This effectively causes the search algorithm to avoid the known root. As further roots are found, M is redefined in the same way.
- Even a good initial value solver like Matlab's ode45 can accumulate roundoff error, which grows exponentially and renders the results meaningless. This can be avoided by means of *multiple shooting*. The integration is done in two parts: one starting from each boundary and meeting in the middle. We'll call that central point $z_m = (z_1+z_2)/2$. The criterion for a solution is that $\hat{w}$ and its derivative be continuous at z_m. We define the matching function

$$M(c) = \hat{w}(z_m^+)\hat{w}_z(z_m^-) - \hat{w}(z_m^-)\hat{w}_z(z_m^+), \tag{13.20}$$

  where z_m^+ and z_m^- indicate the middle point approached from above and below, respectively. Remembering that the solution is defined only up to a multiplicative constant, you can persuade yourself that continuity is achieved if and only if $M = 0$. The rest of the analysis proceeds as before.
- Asymptotic boundary conditions can be applied using the multiple shooting method. At the upper boundary the asymptotic condition is $\hat{w}_z = -k\hat{w}$. So we begin the integration with the conditions $\hat{w} = 1$, $\hat{w}_z = -k$, and similarly for the integration from the lower boundary. An advantage of the shooting method is that asymptotic boundary conditions can also be used in the analysis of stratified and/or viscous flows. In those cases, the matrix method doesn't work because the asymptotic boundary condition depends on the growth rate (or the phase speed), and the problem therefore does not reduce to an algebraic eigenvalue problem. Exercise: Demonstrate this by deriving asymptotic boundary conditions for the Taylor-Goldstein equation (4.18), approximated by second-order finite differences.

13.5 Generalizations

All methods discussed here have been illustrated using the simplest nontrivial problem: inviscid, homogeneous, parallel shear flows as described by Rayleigh's equation. It is straightforward (though not necessarily easy) to generalize any or all of these methods to more complicated situations, e.g., stratified flows as

https://doi.org/10.1017/9781108640084.014 Published online by Cambridge University Press

described by the Taylor-Goldstein equation (5.15) or viscous flows as described by the Orr-Sommerfeld equation (4.18).

13.6 Further Reading

- A thorough discussion of finite difference methods may be found in chapter 3 of Ferziger and Peric (1999).
- The failure of higher-order finite difference methods to improve accuracy has been seen in the analysis of observed alongshore currents (Putrevu and Svendsen, 1992).
- Hazel (1972) is a classic example of the shooting method applied to simple flows.
- Sun et al. (1998) and Smyth et al. (2013) describe the application of shooting methods and finite difference matrix methods in the analysis of observational data.
- Orszag (1971) is a classic demonstration of the Galerkin-Chebyshev method applied to the plane Poiseuille problem (section 5.8.2).
- Rees and Monahan (2014) describe a modern version of the shooting method applied to the Taylor-Goldstein equation.
- As mentioned several times by now, Spiegel (1968) is an extremely useful (and economical) reference for all facets of applied mathematics.