Eigenvalues and Eigenvectors

1 Stability

To this point we've studied both explicit and implicit discretizations of our model equation, the simple linear for of the heat equation with constant diffusivity. We observed experimentally that our choice of explicit discretization technique – the so-called Forward Time Center Space (FTCS) method – was only stable for the choice of

$$C \equiv \frac{\alpha \Delta t}{\Delta x^2} \leq \frac{1}{2n}$$

where $n = 1, 2, 3$ is the dimensionality of the problem. We saw the effect of instability when the solution "blew up" for values just beyond this threshold. For our implicit methods – backward Euler and Crank Nicholson – there was apparently no limit on the size of C to yield a stable solution (accuracy is another issue, something which will be clarified shortly). Now we will study the theoretical basis for this phenomenon.

1.1 Ordinary Differential Equation

We begin our analysis by looking at a simple system – a simple linear first order ordinary differential equation (one independent variable) with constant coefficients:

$$\frac{du}{dt} = -\alpha u \tag{1}$$

where α is a positive constant (as with the heat equation – negative diffusivity is non-physical) and $u = u(t)$. If we denote the initial condition is $u(0) \equiv u_0$, then it should be clear that an analytical solution exists in this case, specifically $\boxed{u(t) = u_0 e^{-\alpha t}.}$ Note that since α is positive the solution tends to zero as $t \to \infty$.

Now we will approximate the solution in the same manner as the full heat equation – by discretizing the system (here only in time since there is no spatial coordinate).

1.1.1 Forward Euler

We start with the forward Euler explicit technique, which since there is no spatial depedence can be written as:

$$\begin{aligned} \frac{u_{n+1} - u_n}{\Delta t} &= -\alpha u_n \\ u_{n+1} &= (1 - \alpha \Delta t) u_n \end{aligned} \tag{2}$$

It is clear that at each time step we multiply the estimated solution at the previous time step by the quantity $1 - \alpha \Delta t$, so that at some arbitrary time step k in the future, the following recursion relation yields the solution:

$$u_k = (1 - \alpha \Delta t)^k u_0 \tag{3}$$

We are always interested in two things: consistency and stability of the approximation. In this case, the stability condition is clear: $|1 - \alpha \Delta t| < 1$, or $\alpha \Delta t < 2$. Otherwise evaluating the k'th power would yield a solution that rapidly grew in time. We know from the analytical solution that the correct behavior is to tend to zero, and thus that a solution that blows up is a numerical artifact due to too large a choice of Δt.

1.1.2 Backward Euler

Now we discretize using an implicit method – the simplest one (though not the most accurate) is Backward Euler, where the derivative is evaluated at the future time step:

$$\begin{aligned} \frac{u_{n+1} - u_n}{\Delta t} &= -\alpha u_{n+1} \\ u_{n+1} + \alpha \Delta t u_{n+1} &= u_n \\ u_k = (1 + \alpha \Delta t)^{-k} u_0 &\to 0 \ \forall \ \alpha \Delta t \end{aligned} \tag{4}$$

It should be clear that this formulation is unconditionally stable. That is, no matter how large Δt, the solution tends to its true steady state value. This unconditional stability is a characteristic of implicit methods. In the case that we have a simple ODE, it is clear why this is the case. We have to then ask what the equivalent matrix property is in the case that the right hand side is multiplied by a matrix rather than a scalar.

1.2 1-D Heat Equation Forward Euler

Recall that for the 1-D heat equation the forward Euler solution is given by:

$$u_j^{n+1} = u_j^n + \frac{\alpha \Delta t}{\Delta x^2}[u_{j+1}^n - 2u_j^n + u_{j-1}^n]$$

When written in matrix form the u_j for an given time can be written as a column vector $\underline{u}$, and we can write the system for constant zero boundary conditions as:

$$\underline{u}^{n+1} = A\underline{u}^n$$

, where A is given as:

$$A = \begin{bmatrix} 1-2c & c & 0 & \dots & & \\ c & 1-2c & c & 0 & \dots & \\ 0 & c & 1-2c & c & 0 & \dots \\ \vdots & \dots & \ddots & & & \end{bmatrix} \tag{5}$$

and C is $\alpha \Delta t \Delta x^{-2}$, a constant. Thus, if we remove the underline for simplicty and move the time index from a superscript to a subscript (to avoid confusion with powers), it is clear that at any timestep k the solution is given as $\boxed{u_k = A^k u_0}$, analogous to the scalar ODE. Taking this analogy further, the question obviously arises: how can we determine the stability condition – ie the property of A such that its k'th power does not increase the values of $\underline{u}$? What does it mean for a matrix to be "less than one" so that the solution stays bounded? In other words, what determines if the k applications of the matrix to the solution drives the solution to the correct steady state or forces it to diverge (go to infinity)[1]. To understand this requires a more general discussion of the concept of eigenvalues and eigenvectors of a matrix. After we understand eigenvalues and eigenvectors, in next week's lecture we will use them to analyze the stability of our heat equation example.

2 Eigenvalues and Eigenvectors

Consider a square matrix A and a vector $\vec{x}$. Think of A as an operator $A\colon \mathbb{R}^n \to \mathbb{R}^n$, where $\mathbb{R}^n$ is a Euclidean vector space. The matrix A is a discrete operator which changes vectors in the vector space. Suppose we find a very special vector such that for some value of $\lambda \neq 0$

$$A\vec{x} = \lambda \vec{x} \tag{6}$$

[1]Note that we can derive an analytical solution in the 2d case as well and show that the solution in steady state should go to zero for homogeneous boundary conditions

where $\lambda \in \mathbb{R}$ (i.e. is a scalar). The set of these such vectors are the eigenvectors of A. The scalars which are associated with them are their corresponding eigenvalues. Geometrically, the eigenvalues represent the amount which the vector is stretched or contracted by A (but $\vec{x}$ is not rotated, by definition!). For a square $n \times n$ matrix, there will be n eigenvalues and eigenvectors (though not necessarily distinct). Note that eigenvalues can be real or imaginary, but this is a distinction we will not worry about until later.

2.1 Determining the eigenvalues

Analytically

For small matrices it is possible to compute the eigenvalues/vectors analytically (for more realistic sized systems we will develop numerical techniques, but studying the analytical method gives insight into eigenvector properties). Note that Equation 6 can be rearranged to give

$$(A - I\lambda)\vec{x} = 0$$

This is equivalent to saying that the matrix $A - I\lambda$ is singular, or $\det(A - I\lambda) = 0$, since the transformation takes a nonzero vector into the null space, and information is "lost", i.e. there is no way to invert the transformation. (See chapter 4 in Strang). We can solve the nth order polynomial from the determinant relation to find the eigenvalues.

Example: Find the eigenvalues and eigenvectors of A

$$A = \begin{bmatrix} 4 & -5 \\ 2 & -3 \end{bmatrix}$$

Using the identity above,

$$A - I\lambda = \begin{bmatrix} 4 - \lambda & -5 \\ 2 & -3 - \lambda \end{bmatrix}$$

$\det(A - I\lambda) = \lambda^2 - \lambda - 2 = 0$ The roots of this equation are $\lambda = -1, 2$. We substitute the eigenvalues back into the equation separately to find their eigenvectors. We get the vectors $\vec{x} = (1, 1)$ and (5 ,2), respectively.

Some properties of eigenvalues

- $\sum \lambda_i$ = trace(A), where the trace of A is the sum of the diagonal elements
- $\lambda_1 \times \ldots \lambda_i = \det(\mathrm{A})$
- The eigenvalues of a triangular matrix are the diagonal elements

2.2 Eigendecomposition

The eigendecomposition of a matrix reveals how the largest eigenvalue of A^{-1} dictates the stability a linear system. of Let A be an $n \times n$ non-defective matrix (i.e. with n distinct eigenvectors). Define S as the special matrix whose columns are the n eigenvectors of A.

$$S = \begin{bmatrix} \vec{x_1} & \vec{x_2} & \dots & \vec{x_n} \\ \vdots & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{7}$$

What is AS?

$$AS = \begin{bmatrix} A\vec{x_1} & A\vec{x_2} & \dots & A\vec{x_n} \\ \vdots & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{8}$$

Since $\vec{x_i}$ are eigenvectors with corresponding eigenvectors λ_i,

$$AS = \begin{bmatrix} \vec{x_1} & \vec{x_2} & \dots & \vec{x_n} \\ \vdots & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \dots & \vdots \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 & \dots \\ 0 & \lambda_2 & 0 & \dots \\ 0 & \vdots & \ddots & \dots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} = S\Lambda \tag{9}$$

Or alternatively, $A = S\Lambda S^{-1}$. This is incredibly useful for computing powers of A. For example, consider

$$A^2 = S\Lambda S^{-1} S\Lambda S^{-1}$$

Since matrix multiplication is associative and a matrix times its inverse is the identity matrix, we have

$$A^k = S\Lambda^k S^{-1} \tag{10}$$

Returning to our original problem, we now can compute u_k without having to explicitly compute the kth power of A. That is, we have

$$u_k = S\Lambda^k S^{-1} u_0$$

It can be shown that the stability condition is related to the spectral radius, defined as

$$\rho(A) = \max(|\lambda_i|) \; . \tag{11}$$

As we will see, a value of $p(A) < 1$ implies stability.

3 Going Backwards: Connecting discrete to continuous

Example: compound interest Let 6 % be the annual interest of a loan, P_0 the principal. How the interest accumulates depends on the period of compounding. Let the interest first compound once annually, for 5 years.

$$P_k = (1.06)^k P_0$$

$$P_5 = (1.06)^5(1000) = \$1338$$

Now let the interest compound monthly

$$P_{60} = (1.0 + \frac{.06}{12})^{60} 1000 = \$1349$$

Finally, let the interest compound daily

$$P_{5\times 365} = (1.0 + \frac{.06}{365})^{5\times 365} 1000 = \$1349.83$$

Observe the trend

$$\lim_{\Delta t \to 0} \frac{P_{k+1} - P_k}{\Delta t} = \frac{dP}{dt} = 0.06P \tag{12}$$

We already know how to solve this ODE; $P(t) = P_0 e^{0.06t}$. Now let's try this process with a matrix rather than a scalar.

Example: Fibonacci Sequence (Strang example)

$F = \{0, 1, 1, 2, 3, 5, 8 \ldots\}$ is the Fibonacci Sequence. What if we want to know F_{1000}? The general form of the sequence is $F_{k+2} = F_{k+1} + F_k$. In matrix form,

$$u_k = \begin{bmatrix} F_{k+l} \\ Fk \end{bmatrix} \tag{13}$$

$$u_{k+1} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} u_k \tag{14}$$

Now, try to use eigendecomposition to find large iterations.

4 Finding eigenvalues numerically

For most matrices it is not practical to calculate eigenvalues directly by solving for the roots of the characteristic equation. In such cases there are a number of different numerical approaches depending on the characterstics of the matrix and the exact information desired. One very popular and simple technique, called power iteration, is a numerical technique for finding the eigenvalue with the largest absolute value.

4.1 Power Iteration

Consider the problem of computing eigenvalues of an $n \times n$ matrix A. Assume that A has n eigenvalues such that

$$|\lambda_1| \geq |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$

. Also, assume that A is diagonalizable, which means that is has n linearly independent eigenvectors $\mathbf{x_1}, \cdots, \mathbf{x_n}$, such that $A\mathbf{x_i} = \lambda_\mathbf{i}\mathbf{x_i}$ for $i = 1, \cdots, n$.

Starting with an initial arbitrary vector $\mathbf{x^{(0)}}$, we continuously multiply A by the result of the previous iterate, obtaining a sequence of vectors $\mathbf{x^{(1)}}, \mathbf{x^{(2)}}, ...$ as:

$$\mathbf{x^{(k)}} = \mathbf{A}\mathbf{x^{(k-1)}} = \mathbf{A^k}\mathbf{x^{(0)}}$$

.

The sequence then converges to the dominant eigenvector of A. This is clear if we expand $\mathbf{x^{(0)}}$ into its eigenvectors:

$$\begin{aligned}
\mathbf{x^{(k)}} &= A^k\mathbf{x^{(0)}} && (15)\\
&= \sum_{i=1}^{n} c_i A^k \mathbf{x_i} && (16)\\
&= \sum_{i=1}^{n} c_i \lambda_i^k \mathbf{x_i} && (17)\\
&= \lambda_1^k \left[c_1\mathbf{x_1} + \sum_{\mathbf{i=2}}^{\mathbf{n}} \mathbf{c_i} \left(\frac{\lambda_\mathbf{i}}{\lambda_\mathbf{1}} \right)^\mathbf{k} \mathbf{x_i} \right] && (18)
\end{aligned}$$

Because $|\lambda_1| > |\lambda_i|$ for $i = 2, \cdots, n$ then the coefficents of $\mathbf{x_i}$ for $i = 2, \cdots, n$ tend to zero as k tends to infinity. Therefore, the direction of $\mathbf{x^{(k)}}$ converges to that of $\mathbf{x_1}$. This provides a simple technique to estimate the largest eigenvalue computationally.