On stability of numerical schemes

How to discretize time for some problems

18.303 Linear Partial Differential Equations: Analysis and Numerics

Behavior of time-dependent problems

Let us consider the differential equation

$$\alpha \frac{\partial^2 u(t, \mathsf{x})}{\partial t^2} + \mu \frac{\partial u(t, \mathsf{x})}{\partial t} = \mathcal{L} u(t, \mathsf{x})$$

with some boundary conditions that we will not specify at this point. Here we assume $\alpha, \mu \geq 0$.

Behavior of time-dependent problems

Let us consider the differential equation

$$\alpha \frac{\partial^2 u(t, \mathsf{x})}{\partial t^2} + \mu \frac{\partial u(t, \mathsf{x})}{\partial t} = \mathcal{L} u(t, \mathsf{x})$$

with some boundary conditions that we will not specify at this point. Here we assume $\alpha, \mu \geq 0$.

We assume that $\mathcal{L}$ has a complete eigenbasis $\{\phi_{\mathsf{n}}\}$ with

$$\mathcal{L}\phi_{\mathsf{n}}(\mathsf{x}) = \lambda_{\mathsf{n}}\phi_{\mathsf{n}}(\mathsf{x})$$

and

$$u(t, \mathsf{x}) = \sum_{\mathsf{n}} \hat{u}_{\mathsf{n}}(t)\phi_{\mathsf{n}}(\mathsf{x}).$$

Plugging the expansion of u in the differential equation and using the *linear independence* of ϕ_{n} gives

$$\alpha \frac{\partial^2 \hat{u}_{\mathsf{n}}(t)}{\partial t^2} + \mu \frac{\partial \hat{u}_{\mathsf{n}}(t)}{\partial t} = \lambda_{\mathsf{n}} \hat{u}_{\mathsf{n}}(t).$$

This equation is solved by an exponential ansatz

$$\hat{u}_{\mathsf{n}} = \exp(a_{\mathsf{n}} t).$$

Plugging the expansion of u in the differential equation and using the *linear independence* of ϕ_n gives

$$\alpha \frac{\partial^2 \hat{u}_n(t)}{\partial t^2} + \mu \frac{\partial \hat{u}_n(t)}{\partial t} = \lambda_n \hat{u}_n(t).$$

This equation is solved by an exponential ansatz

$$\hat{u}_n = \exp(a_n t).$$

Plugging this in gives the characteristic equation

$$\left(\alpha a_n^2 + \mu a_n - \lambda_n\right) \exp(a_n t) = 0,$$

which can be solved giving

$$a_n = \frac{-\mu \pm \sqrt{\mu^2 + 4\alpha\lambda_n}}{2\alpha}$$

if $\alpha \neq 0$ and

$$a_n = \frac{\lambda_n}{\mu}$$

otherwise.

Case i: $\alpha = 0$

This describes first order dynamic. The solution of the $\mathsf{n}th$ coefficient is proportional to

$$\exp\left(\frac{\lambda_{\mathsf{n}}}{\mu}t\right).$$

Case i: $\alpha = 0$

This describes first order dynamic. The solution of the $\mathsf{n}th$ coefficient is proportional to

$$\exp\left(\frac{\lambda_{\mathsf{n}}}{\mu} t\right).$$

Depending on the sign of λ_{n} this solution is either growing or decreasing exponentially.

Case i: $\alpha = 0$

This describes first order dynamic. The solution of the n*th* coefficient is proportional to

$$\exp\left(\frac{\lambda_n}{\mu}t\right).$$

Depending on the sign of λ_n this solution is either growing or decreasing exponentially.

Note: remember that the eigenvalues of the Laplacian are negative (semi)definite. The modes with the zero eigenvalue are constant in time and the rest of them decrease. This is the case for the heat equation. See how important it is that the eigenvalues are negative?

Case ii: $\mu = 0$

This gives a kind of wave equation. Now the solution is

$$\exp\left(\pm\sqrt{\frac{\lambda_n}{\alpha}}t\right).$$

Again, if *all* the eigenvalues are negative, the solution will be oscillating. Otherwise there will be a solution (for each n) that is decaying and a solution that grows exponentially. We also have the special case $\lambda_n = 0$ for modes that do not change in time.

General case

$$\hat{u}_{\mathsf{n}}(t) \propto \exp\left(\frac{-\mu \pm \sqrt{\mu^2 + 4\alpha\lambda_{\mathsf{n}}}}{2\alpha} t\right)$$

General case

$$\hat{u}_{\mathsf{n}}(t) \propto \exp\left(\frac{-\mu \pm \sqrt{\mu^2 + 4\alpha\lambda_{\mathsf{n}}}}{2\alpha} t\right)$$

If we assume that $\alpha \neq 0$ we can write this as

$$\hat{u}_{\mathsf{n}}(t) \propto \exp\left(\frac{-\mu \pm \sqrt{\mu^2 + 4\lambda_{\mathsf{n}}}}{2} t\right) = \exp\left(\frac{-\mu t}{2}\right) \exp\left(\pm\frac{\sqrt{\mu^2 + 4\lambda_{\mathsf{n}}}}{2} t\right)$$

with the rescaling $\mu \to \mu/\alpha$ and $\lambda_{\mathsf{n}} \to \lambda_{\mathsf{n}}/\alpha$ (if we were careful here, we would use different symbols for these new coefficients but let's allow us to be a bit lazy).

$$\hat{u}_n(t) \propto \exp\left(\frac{-\mu t}{2}\right) \exp\left(\pm \frac{\sqrt{\mu^2 + 4\lambda_n}}{2} t\right)$$

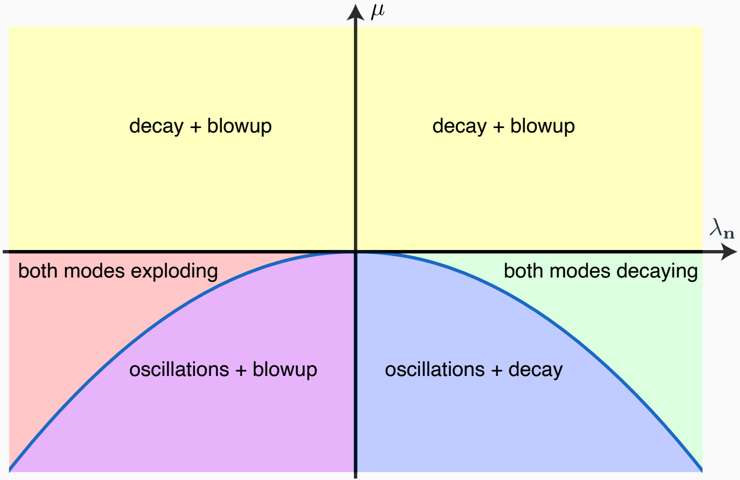

Diagram of different behavior

Stability

- Stability of time-dependent numerical solutions can mean many things

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)
- Many times we use scalar quantities like the energy to see if the numerical solution agrees with known behavior of the equation

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)
- Many times we use scalar quantities like the energy to see if the numerical solution agrees with known behavior of the equation
- For example:

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)
- Many times we use scalar quantities like the energy to see if the numerical solution agrees with known behavior of the equation
- For example:
    - Wave equations are *energy conserving*

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)
- Many times we use scalar quantities like the energy to see if the numerical solution agrees with known behavior of the equation
- For example:
  - Wave equations are *energy conserving*
  - For heat equations energies are non-increasing

Stability

- Stability of time-dependent numerical solutions can mean many things
- Something that is *always* required is that the solutions do not blow up unphysically (e.g. the solution to the heat equation doesn't blow up)
- Many times we use scalar quantities like the energy to see if the numerical solution agrees with known behavior of the equation
- For example:
    - Wave equations are *energy conserving*
    - For heat equations energies are non-increasing
- We will get back to energies later; for now we'll concentrate on stability in the sense that the solutions don't blow up

Let us look at a heat equation with

$$\frac{\partial u(t, \mathsf{x})}{\partial t} = \mathcal{L} u(t, \mathsf{x})$$

with some sufficient boundary conditions.

Let us look at a heat equation with

$$\frac{\partial u(t, \mathsf{x})}{\partial t} = \mathcal{L}u(t, \mathsf{x})$$

with some sufficient boundary conditions.

Assume that we discretize the function in time (but not in space, for now). We introduce a time step s and function values

$$u^{(k)}(\mathsf{x}) = u(ks, \mathsf{x}).$$

Let us look at a heat equation with

$$\frac{\partial u(t, \mathsf{x})}{\partial t} = \mathcal{L} u(t, \mathsf{x})$$

with some sufficient boundary conditions.

Assume that we discretize the function in time (but not in space, for now). We introduce a time step s and function values

$$u^{(k)}(\mathsf{x}) = u(ks, \mathsf{x}).$$

Forward difference will give

$$\frac{u^{(k+1)} - u^{(k)}}{s} = \mathcal{L} u^{(k)}.$$

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathsf{n}}^{(k+1)} - \hat{u}_{\mathsf{n}}^{(k)}}{s} = -\lambda_{\mathsf{n}} \hat{u}_{\mathsf{n}}^{(k)}.$$

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_{\mathbf{n}} \hat{u}_{\mathbf{n}}^{(k)}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = (1 - s\lambda_{\mathbf{n}}) \hat{u}_{\mathbf{n}}^{(k)}.$$

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_{\mathbf{n}} \hat{u}_{\mathbf{n}}^{(k)}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = (1 - s\lambda_{\mathbf{n}}) \hat{u}_{\mathbf{n}}^{(k)}.$$

This recursion relation is easy to solve giving

$$\hat{u}_{\mathbf{n}}^{(k)} = (1 - s\lambda_{\mathbf{n}})^k \hat{u}_{\mathbf{n}}^{(0)}.$$

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_{\mathbf{n}} \hat{u}_{\mathbf{n}}^{(k)}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = (1 - s\lambda_{\mathbf{n}}) \hat{u}_{\mathbf{n}}^{(k)}.$$

This recursion relation is easy to solve giving

$$\hat{u}_{\mathbf{n}}^{(k)} = (1 - s\lambda_{\mathbf{n}})^{k} \hat{u}_{\mathbf{n}}^{(0)}.$$

We can easily see that this is bounded if $|1 - s\lambda_{\mathbf{n}}| < 1$.

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_{\mathbf{n}} \hat{u}_{\mathbf{n}}^{(k)}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = (1 - s\lambda_{\mathbf{n}}) \hat{u}_{\mathbf{n}}^{(k)}.$$

This recursion relation is easy to solve giving

$$\hat{u}_{\mathbf{n}}^{(k)} = (1 - s\lambda_{\mathbf{n}})^k \hat{u}_{\mathbf{n}}^{(0)}.$$

We can easily see that this is bounded if $|1 - s\lambda_{\mathbf{n}}| < 1$.

For physical systems, the eigenvalues are negative $(\lambda_{\mathbf{n}} > 0)$ and we only have to worry about the most negative eigenvalue in the system.

We write this in the eigenbasis of $\mathcal{L}$ as

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_{\mathbf{n}} \hat{u}_{\mathbf{n}}^{(k)}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = (1 - s\lambda_{\mathbf{n}}) \hat{u}_{\mathbf{n}}^{(k)}.$$

This recursion relation is easy to solve giving

$$\hat{u}_{\mathbf{n}}^{(k)} = (1 - s\lambda_{\mathbf{n}})^{k} \hat{u}_{\mathbf{n}}^{(0)}.$$

We can easily see that this is bounded if $|1 - s\lambda_{\mathbf{n}}| < 1$.

For physical systems, the eigenvalues are negative ($\lambda_{\mathbf{n}} > 0$) and we only have to worry about the most negative eigenvalue in the system.

Assume we have some *truncated* basis with N eigenvectors. We assume λ_N to be the largest eigenvalue (by absolute value). Now we get the condition

$$1 - s\lambda_N > -1 \Leftrightarrow s < 2/\lambda_N.$$

What if we also have spatial discretization?

What if we also have spatial discretization?

We will just have a matrix substitution $\mathcal{L} \to L$ with some matrix L. It will have its own eigenbasis and associated eigenvalues.

What if we also have spatial discretization?

We will just have a matrix substitution $\mathcal{L} \to L$ with some matrix L. It will have its own eigenbasis and associated eigenvalues.

Exercise: Assume $\mathcal{L} = \Delta$ (the Laplacian). Assume we have Dirichlet boundaries for the heat equation. Now

$$A = \frac{1}{\Delta x^2} \begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix}.$$

What is the stability condition for the time step?

The eigenvectors of A are $\phi_n(k) = \sin(nk\pi/(N+1))$ and the associated eigenvalues are

$$-2 \cdot \frac{1 - \cos(n\pi/(N+1))}{\Delta x^2}.$$

The eigenvectors of A are $\phi_n(k) = \sin(nk\pi/(N+1))$ and the associated eigenvalues are

$$-2 \cdot \frac{1 - \cos(n\pi/(N+1))}{\Delta x^2}.$$

The largest eigenvalue comes with $n = N$, which, for large N, we can approximate as

$$-2(1 - \cos(N\pi/(N+1))) \approx -2(1 - \cos(\pi)) = -4.$$

The eigenvectors of A are $\phi_n(k) = \sin(nk\pi/(N+1))$ and the associated eigenvalues are

$$-2 \cdot \frac{1-\cos(n\pi/(N+1))}{\Delta x^2}.$$

The largest eigenvalue comes with $n = N$, which, for large N, we can approximate as

$$-2(1-\cos(N\pi/(N+1))) \approx -2(1-\cos(\pi)) = -4.$$

This gives $\lambda_N \approx 4/\Delta x^2$. Now the condition becomes

$$s < 2/\lambda_N = \Delta x^2/2.$$

The eigenvectors of A are $\phi_n(k) = \sin(nk\pi/(N+1))$ and the associated eigenvalues are

$$-2 \cdot \frac{1 - \cos(n\pi/(N+1))}{\Delta x^2}.$$

The largest eigenvalue comes with $n = N$, which, for large N, we can approximate as

$$-2(1 - \cos(N\pi/(N+1))) \approx -2(1 - \cos(\pi)) = -4.$$

This gives $\lambda_N \approx 4/\Delta x^2$. Now the condition becomes

$$s < 2/\lambda_N = \Delta x^2/2.$$

This is called the Courant–Friedrichs–Lewy (CFL) condition. For those of you who did Pset 1, you probably noticed that the time step had to be very small. This explains why.

Let's try something else. Let's discretize the time as

$$\frac{u^{(k+1)} - u^{(k)}}{s} = \frac{\mathcal{L}u^{(k)} + \mathcal{L}u^{(k+1)}}{2}.$$

Let's try something else. Let's discretize the time as

$$\frac{u^{(k+1)} - u^{(k)}}{s} = \frac{\mathcal{L}u^{(k)} + \mathcal{L}u^{(k+1)}}{2}.$$

This is called the Crank-Nicolson method. It's equivalent to using central difference for time at time $k + 1/2$. For the coefficients we get

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_n \frac{\hat{u}_{\mathbf{n}}^{(k)} + \hat{u}_{\mathbf{n}}^{(k+1)}}{2}.$$

Let's try something else. Let's discretize the time as

$$\frac{u^{(k+1)} - u^{(k)}}{s} = \frac{\mathcal{L}u^{(k)} + \mathcal{L}u^{(k+1)}}{2}.$$

This is called the Crank-Nicolson method. It's equivalent to using central difference for time at time $k + 1/2$. For the coefficients we get

$$\frac{\hat{u}_{\mathbf{n}}^{(k+1)} - \hat{u}_{\mathbf{n}}^{(k)}}{s} = -\lambda_n \frac{\hat{u}_{\mathbf{n}}^{(k)} + \hat{u}_{\mathbf{n}}^{(k+1)}}{2}.$$

Solving for $\hat{u}_{\mathbf{n}}^{(k+1)}$ gives

$$\hat{u}_{\mathbf{n}}^{(k+1)} = \left(\frac{1 - \lambda_n s/2}{1 + \lambda_n s/2} \right) \hat{u}_{\mathbf{n}}^{(k)}$$

Assume $\lambda_N > 0$ (negative definite operator).

Assume $\lambda_N > 0$ (negative definite operator).

Solving for the condition gives

$$\left|\frac{1-\lambda_n s/2}{1+\lambda_n s/2}\right| < 1,$$

which we can write as

$$-1 < \frac{1-\lambda_n s/2}{1+\lambda_n s/2} < 1.$$

Assume $\lambda_N > 0$ (negative definite operator).

Solving for the condition gives

$$\left|\frac{1-\lambda_n s/2}{1+\lambda_n s/2}\right| < 1,$$

which we can write as

$$-1 < \frac{1-\lambda_n s/2}{1+\lambda_n s/2} < 1.$$

The lower bound gives $-1 < 1$ while the upper bound gives $-\lambda_n < \lambda_n$, which is always true if $\lambda_n > 0$.

Assume $\lambda_N > 0$ (negative definite operator).

Solving for the condition gives

$$\left|\frac{1-\lambda_n s/2}{1+\lambda_n s/2}\right| < 1,$$

which we can write as

$$-1 < \frac{1-\lambda_n s/2}{1+\lambda_n s/2} < 1.$$

The lower bound gives $-1 < 1$ while the upper bound gives $-\lambda_n < \lambda_n$, which is always true if $\lambda_n > 0$.

We see that this method is unconditionally stable i.e. the stability doesn't depend on s if we would do the spatial discretization.

Assume $\lambda_N > 0$ (negative definite operator).

Solving for the condition gives

$$\left|\frac{1-\lambda_n s/2}{1+\lambda_n s/2}\right| < 1,$$

which we can write as

$$-1 < \frac{1-\lambda_n s/2}{1+\lambda_n s/2} < 1.$$

The lower bound gives $-1 < 1$ while the upper bound gives $-\lambda_n < \lambda_n$, which is always true if $\lambda_n > 0$.

We see that this method is unconditionally stable i.e. the stability doesn't depend on s if we would do the spatial discretization.

Word of warning: stability just means here that the discretized system doesn't blow up. We should still make s sufficiently small if we want to approximate the time evolution well.