PHY407: Computational Physics Fall, 2016

Instructor: Paul Kushner

TAs: Miriam Diamond and Heather Fong

Marker: Chad Gu

Lecture 9: Partial differential equations, Part 2

Summary & Status

☑ Weeks 1-3: Programming basics, numerical errors, numerical integration and differntiation.

☑ Weeks 4-5: Solving linear & nonlinear systems and Fourier transforms.

☑ Week 6: ODEs Part 1: RK4, Leapfrog, Verlet, adaptive time stepping; customizing python output

☑ Week 7: ODEs Part 2: Bulirsch-Stoer, Boundary Value Problems/shooting,

☑ Week 8: PDEs Part 1: Elliptic equation solvers, leapfrog time stepping, FTCS

❑ Week 9: PDEs Parts 2

  – Crank-Nicholson and Spectral Methods

❑ Weeks 10-11: Random numbers & Monte Carlo methods

PHY407: Computational Physics
Fall, 2016

Instructor: Paul Kushner

TAs: Miriam Diamond and Heather Fong

Marker: Chad Gu

Lecture 9: Partial Differential Equations, Part 2

- Stability of FTCS, implicit methods, and the Crank Nicholson Method
- Spectral Method

Recall: FTCS Method for the Heat Equation

For the 1-D heat equation:

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2},$$

The FTCS method is:

$$T_j^{n+1} = T_j^n + \frac{\kappa h}{a^2}\left(T_{j+1}^n - 2T_j^n + T_{j-1}^n\right)$$

where n is timestep, h is timestep width, j is grid point, a is grid spacing.

Consider the stability of a single Fourier mode: $\hat{T}_k^n \exp\left(ikx_j\right) = \hat{T}_k^n \exp\left(iajk\right)$:

$$\left|\frac{\hat{T}_k^{n+1}}{\hat{T}_k^n}\right| = 1 + \frac{\kappa h}{a^2}\left(e^{ika} + e^{-ika} - 2\right) = 1 - \frac{4h\kappa}{a^2}\sin^2\left(\tfrac{1}{2}ka\right)$$

For stability:

$$h \le \frac{a^2}{2\kappa}$$

Recall: FTCS for the Wave Equation

$$\frac{\partial^2 \phi}{\partial t^2} = c^2 \frac{\partial^2 \phi}{\partial x^2},$$

(Subject to suitable boundary and initial conditions.)

FTCS for the Wave Equation

$$\frac{\partial^2 \phi}{\partial t^2} = c^2 \frac{\partial^2 \phi}{\partial x^2},$$

(Subject to suitable boundary and initial conditions.)

$$\frac{d^2 \phi_j(t)}{dt^2} \approx \frac{c^2}{a^2}\left(\phi_{j+1}(t) - 2\phi_j(t) + \phi_{j-1}(t)\right),\ j = 1, \ldots, J-1$$

FTCS for the Wave Equation

$$\frac{\partial^2 \phi}{\partial t^2} = c^2 \frac{\partial^2 \phi}{\partial x^2},$$

(Subject to suitable boundary and initial conditions.)

$$\frac{d^2 \phi_j(t)}{dt^2} \approx \frac{c^2}{a^2}\left(\phi_{j+1}(t) - 2\phi_j(t) + \phi_{j-1}(t)\right),\ j = 1,\ldots,J-1$$

Now transform to pairs of first order ODEs:

$$\frac{d\phi_j}{dt} = \psi_j$$

$$\frac{d\psi_j}{dt} = \frac{c^2}{a^2}\left(\phi_{j+1} - 2\phi_j + \phi_{j-1}\right)$$

and discretize using Euler-Forward (so there about 2J ODEs).

FTCS for the Wave Equation

$$\begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} = \begin{pmatrix} 1 & h \\ \dfrac{-2hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^n \\ \psi_j^n \end{pmatrix} + \begin{pmatrix} 0 \\ \dfrac{c^2 h}{a^2}\left(\phi_{j+1}^n + \phi_{j-1}^n\right) \end{pmatrix}$$

Now consider a single Fourier component with wavenumber k as before: substitute $\hat{\phi}_k^n \exp(ikja)$ and $\hat{\psi}_k^n \exp(ikja)$ into above, and use $e^{ika} + e^{-ika} = 2\cos ka$ and the half-angle formula.

We get

$$\begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \mathbf{A} \begin{pmatrix} \hat{\phi}_k^n \\ \hat{\psi}_k^n \end{pmatrix}, \mathbf{A} = \begin{pmatrix} 1 & h \\ -hr^2 & 1 \end{pmatrix}, r = \frac{2c}{a}\sin\left(\frac{ka}{2}\right)$$

which depends on k.

Stability for this method.

$$\begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \mathbf{A} \begin{pmatrix} \hat{\phi}_k^{n} \\ \hat{\psi}_k^{n} \end{pmatrix}, \mathbf{A} = \begin{pmatrix} 1 & h \\ -hr^2 & 1 \end{pmatrix}, r = \frac{2c}{a} \sin\left(\frac{ka}{2}\right)$$

Now the eigenvalues of $\mathbf{A}$ are $\lambda_\pm = 1 \pm ihr$ and $|\lambda_\pm|^2 = 1 + h^2 r^2 \geq 1$ with corresponding eigenvectors $\mathbf{v}_+, \mathbf{v}_-$.

Suppose initial condition is $\alpha_+ \mathbf{v}_+ + \alpha_- \mathbf{v}_-$. After m timesteps, this becomes $\alpha_+ \lambda_+^m \mathbf{v}_+ + \alpha_- \lambda_-^m \mathbf{v}_-$, which will grow without bound!

So, FTCS is never stable for the wave equation!

The Implicit Method

- We have other choices on how to discretize in time the set of ODES

$$\frac{d\phi_j}{dt} = \psi_j, \quad \frac{d\psi_j}{dt} = \frac{c^2}{a^2}\left(\phi_{j+1} - 2\phi_j + \phi_{j-1}\right)$$

- What if we evaluate the RHS at time t+h instead of t?

The Implicit Method

$$\begin{pmatrix} \phi_j^n \\ \psi_j^n \end{pmatrix} = \begin{pmatrix} 1 & -h \\ \dfrac{2hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} - \begin{pmatrix} 0 \\ \dfrac{c^2h}{a^2}\left(\phi_{j+1}^{n+1} + \phi_{j-1}^{n+1}\right) \end{pmatrix}$$

The RHS is a matrix expression that we can invert to get the LHS.

If we do the Von Neumann substitution, $\hat{\phi}_k^n \exp(ikja)$ and $\hat{\psi}_k^n \exp(ikja)$ we get

$$\mathbf{B}\begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \begin{pmatrix} \hat{\phi}_k^n \\ \hat{\psi}_k^n \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 1 & -h \\ hr^2 & 1 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \mathbf{B}^{-1} \begin{pmatrix} \hat{\phi}_k^n \\ \hat{\psi}_k^n \end{pmatrix}.$$

$\mathbf{B}^{-1}$ has eigenvalues $\lambda_{\pm} = {(1 \pm ihr)}\big/{\left(1 + h^2r^2\right)}, \left|\lambda_{\pm}\right| \leq 1$

The Implicit Method

$$\begin{pmatrix} \phi_j^n \\ \psi_j^n \end{pmatrix} = \begin{pmatrix} 1 & -h \\ \frac{2hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} - \begin{pmatrix} 0 \\ \frac{c^2h}{a^2}\left(\phi_{j+1}^{n+1} + \phi_{j-1}^{n+1}\right) \end{pmatrix}$$

The RHS is a matrix expression that we can invert to get the LHS.

If we do the Von Neumann substitution, $\hat{\phi}_k^n \exp(ikja)$ and $\hat{\psi}_k^n \exp(ikja)$ we get

$$\mathbf{B}\begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \begin{pmatrix} \hat{\phi}_k^n \\ \hat{\psi}_k^n \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 1 & -h \\ hr^2 & 1 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} \hat{\phi}_k^{n+1} \\ \hat{\psi}_k^{n+1} \end{pmatrix} = \mathbf{B}^{-1}\begin{pmatrix} \hat{\phi}_k^n \\ \hat{\psi}_k^n \end{pmatrix}.$$

$\mathbf{B}^{-1}$ has eigenvalues $\lambda_\pm = {(1 \pm ihr)}\Big/{\left(1+h^2r^2\right)}, \left|\lambda_\pm\right| \le 1$

The eigenvalues are the growth factors and these are less than one.

So the implicit method is unconditionally stable.

But solutions decay exponentially! This is a big problem for the wave equation!

Crank Nicholson is an average of Euler Forward and Implicit

EF:

$$\begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} = \begin{pmatrix} 1 & h \\ \dfrac{-2hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n} \\ \psi_j^{n} \end{pmatrix} + \begin{pmatrix} 0 \\ \dfrac{c^2h}{a^2}\left(\phi_{j+1}^{n} + \phi_{j-1}^{n}\right) \end{pmatrix}$$

Implicit:

$$\begin{pmatrix} \phi_j^{n} \\ \psi_j^{n} \end{pmatrix} = \begin{pmatrix} 1 & -h \\ \dfrac{2hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} - \begin{pmatrix} 0 \\ \dfrac{c^2h}{a^2}\left(\phi_{j+1}^{n+1} + \phi_{j-1}^{n+1}\right) \end{pmatrix}$$

CN:

$$\begin{pmatrix} 1 & -h/2 \\ \dfrac{hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n+1} \\ \psi_j^{n+1} \end{pmatrix} - \begin{pmatrix} 0 \\ \dfrac{c^2h}{2a^2}\left(\phi_{j+1}^{n+1} + \phi_{j-1}^{n+1}\right) \end{pmatrix}$$

$$= \begin{pmatrix} 1 & h/2 \\ \dfrac{-hc^2}{a^2} & 1 \end{pmatrix} \begin{pmatrix} \phi_j^{n} \\ \psi_j^{n} \end{pmatrix} + \begin{pmatrix} 0 \\ \dfrac{c^2h}{2a^2}\left(\phi_{j+1}^{n} + \phi_{j-1}^{n}\right) \end{pmatrix}$$

Growth factors: $\lambda_{\pm} = {\left(1 - h^2r^2 \pm 2ihr\right)}\Big/{\left(1 + h^2r^2\right)}, r = \dfrac{c}{a}\sin\left(\dfrac{ka}{2}\right), \left|\lambda_{\pm}\right| \equiv 1$

Crank Nicholson: A Balanced Method

Crank-Nicholson Growth factors:

$$\lambda_{\pm} = \frac{1 - h^2 r^2 \pm 2ihr}{1 + h^2 r^2}, \ r = \frac{c}{a}\sin\left(\frac{ka}{2}\right),$$

$$\boxed{\left|\lambda_{\pm}\right| \equiv 1}$$

For Euler-Forward, the growth factors are greater than one and the solution diverges.

For Implicit, the growth factors are less than one and the solution decays to zero.

For CN, the growth factors are one so the solution neither grows nor decays.

Help for Lab09

- The motivation for Crank Nicholson is to obtain useful solutions of the (second order) wave equation.
- But in Lab 09 we are going to solve the time dependent Schrödinger equation,

$$-\frac{\hbar^2}{2M}\frac{\partial^2\psi}{\partial x^2} = \mathrm{i}\hbar\frac{\partial\psi}{\partial t}.$$

- This is only first order in time – a bit easier.
- Let’s review what Newman says in Exercise 9.8.

CN for TDSE in matrix form

$$\begin{aligned}\psi(x,t+h) - h\frac{\mathrm{i}\hbar}{4ma^2}\big[\psi(x+a,t+h) + \psi(x-a,t+h) - 2\psi(x,t+h)\big] \\ = \psi(x,t) + h\frac{\mathrm{i}\hbar}{4ma^2}\big[\psi(x+a,t) + \psi(x-a,t) - 2\psi(x,t)\big].\end{aligned}$$

CN for TDSE in matrix form

$$\psi(x,t+h) - h\frac{\mathrm{i}\hbar}{4ma^2}\big[\psi(x+a,t+h)+\psi(x-a,t+h)-2\psi(x,t+h)\big]$$
$$= \psi(x,t) + h\frac{\mathrm{i}\hbar}{4ma^2}\big[\psi(x+a,t)+\psi(x-a,t)-2\psi(x,t)\big].$$

$$\boldsymbol{\psi}(t) = \begin{pmatrix} \psi(a,t) \\ \psi(2a,t) \\ \psi(3a,t) \\ \vdots \end{pmatrix}.$$

$$\mathbf{A}\boldsymbol{\psi}(t+h) = \mathbf{B}\boldsymbol{\psi}(t)$$

CN for TDSE in matrix form

$$\psi(x,t+h) - h\frac{i\hbar}{4ma^2}\big[\psi(x+a,t+h)+\psi(x-a,t+h)-2\psi(x,t+h)\big]$$
$$= \psi(x,t) + h\frac{i\hbar}{4ma^2}\big[\psi(x+a,t)+\psi(x-a,t)-2\psi(x,t)\big].$$

$$\boldsymbol{\psi}(t) = \begin{pmatrix} \psi(a,t) \\ \psi(2a,t) \\ \psi(3a,t) \\ \vdots \end{pmatrix}. \qquad \mathbf{A}\boldsymbol{\psi}(t+h) = \mathbf{B}\boldsymbol{\psi}(t)$$

$$\mathbf{A} = \begin{pmatrix} a_1 & a_2 & & & \\ a_2 & a_1 & a_2 & & \\ & a_2 & a_1 & a_2 & \\ & & a_2 & a_1 & \\ & & & & \ddots \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} b_1 & b_2 & & & \\ b_2 & b_1 & b_2 & & \\ & b_2 & b_1 & b_2 & \\ & & b_2 & b_1 & \\ & & & & \ddots \end{pmatrix},$$

$$a_1 = 1 + h\frac{i\hbar}{2ma^2}, \qquad a_2 = -h\frac{i\hbar}{4ma^2}, \qquad b_1 = 1 - h\frac{i\hbar}{2ma^2}, \qquad b_2 = h\frac{i\hbar}{4ma^2}.$$

Help for Lab09

- So the algorithm is as follows:
- Create A and B matrices (these don’t change in time)
- For each time:
  - Calculate RHS at time t. $\mathbf{B}\boldsymbol{\psi}(t)$
  - Solve for wave function by solving the linear system

$$\mathbf{A}\boldsymbol{\psi}(t+h) = \mathbf{B}\boldsymbol{\psi}(t)$$

- Take advantage of the banded structure of A.
- Use a call like:

```
    psi=banded(A, v, 1, 1)
```

- This says “solve Ax=v assuming A has 1 diagonal above the main diagonal and 1 diagonal below.”

What is `banded.py`?

- It is a function that does Gaussian elimination and back substitution taking advantage of the banded structure.
- You reduce required storage by storing the banded matrix compactly.
- The call is

```
      x = banded(A,v,up,down)
```

- `A` is `(1+up+down) x N` elements.
- Let’s look at the code.

Spectral Methods

- Finite difference isn’t the only way to discretize spatial part of PDEs.
- Spectral method: use Fourier analysis to break up the system into spatial waves.
- We can do this efficiently using the FFT routines we worked on in earlier labs.
- The trick is to express the equations in the right form.

Fourier Transform Tricks

$$f = \sum_{k=-\infty}^{\infty} e^{ikx} \hat{f}_k \Rightarrow \frac{\partial f}{\partial x} = \sum_{k=-\infty}^{\infty} ike^{ikx} \hat{f}_k$$

Or in shorthand

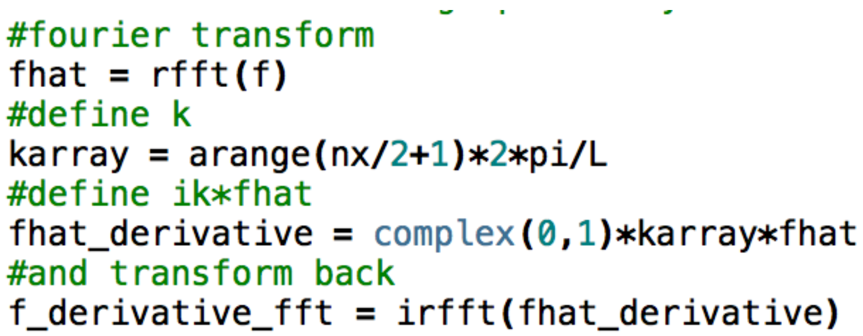
$$\partial f / \partial x \rightarrow ik \ \hat{f}_k, \partial^2 f / \partial x^2 \rightarrow -k^2 \hat{f}_k$$

Pointers on using FFTs to calculate derivatives

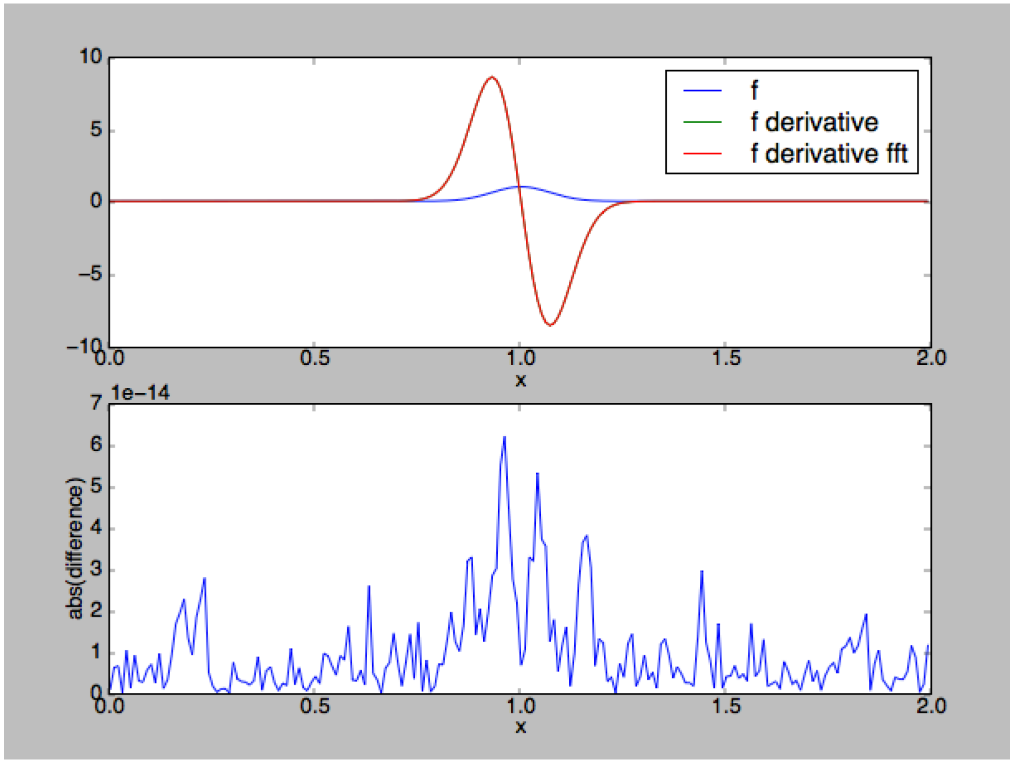
- When you go from exp(ikx) to the FFT notation, you need to be careful to define your wavenumber axis.
- Suppose you have `f[0…nx]` and call `fhat=rfft(f)`
- Don't worry about wavenumber zero (this does not evolve), which is `fhat[0]`
- `fhat[1]` corresponds to wavenumber k=2π/L, `fhat[j]` corresponds to k=2πj/L.
- Transform the coefficients by multiplying by k factors, then transform back using `irfft`.
- Your program should be really fast to run.

Quick Example

- The code derivative_fft.py gives a quick example of defining $\partial f / \partial x \rightarrow ik\ \hat{f}_k$



```
#fourier transform
fhat = rfft(f)
#define k
karray = arange(nx/2+1)*2*pi/L
#define ik*fhat
fhat_derivative = complex(0,1)*karray*fhat
#and transform back
f_derivative_fft = irfft(fhat_derivative)
```



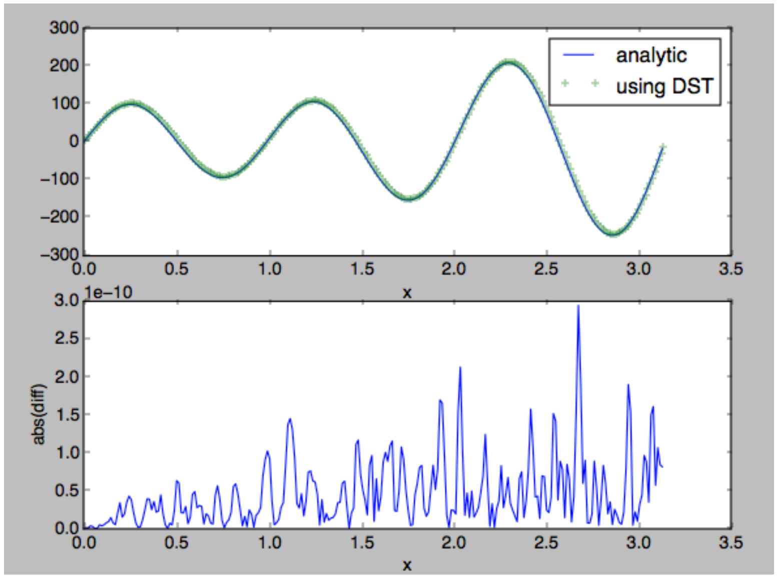
Quick Examples

- The code derivative_dst.py gives a quick example of using the discrete sine transform and taking second derivatives with it.

```
from dcst import dst, idst, dct, idct

N = 256
# x = pi*n/N
x = arange(N)*pi/N
#function is a sine series
f = sin(x)-2*sin(4*x)+3*sin(5*x)-4*sin(6*x)
#do fourier sine series
fCoeffs = dst(f)
print 'Original series: f = sin(x)-2sin(4x)+3sin(5x)-4sin(6x)'
for j in range(7):
    print 'Coefficient of sin(%ix):'%j, fCoeffs[j]/N

print 'See Figure for calculating second derivative'
#second derivative is also a sine series
d2f_dx2_a = -sin(x)+32*sin(4*x)-75*sin(5*x)+144*sin(6*x)
#second derivative using fourier transform
DerivativeCoeffs = -arange(N)**2*fCoeffs
d2f_dx2_b = idst(DerivativeCoeffs)
```



PHY407: Computational Physics
Fall, 2016

Instructor: Paul Kushner

TAs: Miriam Diamond and Heather Fong

Marker: Chad Gu

Lecture 9: Partial Differential Equations, Part 2

- Stability of FTCS, implicit methods, and the Crank Nicholson Method
- Spectral Method