Numerical Methods II, Courant Institute, Spring 2014

http://www.math.nyu.edu/faculty/goodman/teaching/NumMethII2014/index.html

Always check the `classes` message board before doing any work on the assignment.

Assignment 2, due February ??

Corrections: Fixed (8): $m\pi y_j \longrightarrow m\pi y_j$. Fixed the formerly silly formula in part (3d).

1. (*Spectral differentiation*) Suppose $u(x)$ is a function, x_j are some points, and $U_j = u(x_j)$ are the corresponding function values. *Interpolation* means estimating values $u(x)$ for $x \neq x_j$. *Numerical differentiation* means estimating derivatives of u either at sample points (the x_j) or other points. Use the following notation conventions: u is periodic in x with $u(x+L) = u(x)$. There are $|B| = N$ uniformly spaced grid points $x_j = j\Delta x$, with $\Delta x = L/N$. The sample values $U_j = u(x_j)$ form a vector $U = (u_0, \ldots, u_{N-1})^t \in \mathbb{C}^N$. The Fourier coefficients of u are

$$\widehat{u}_n = \frac{1}{L}\int_0^L e^{-2\pi i n x/L} u(x)\, dx \; .$$

The Fourier series representation of u is

$$u(x) = \sum_{-\infty}^{\infty} \widehat{u}_n e^{2\pi i n x/L} \; .$$

The *dual box*, B', is as symmetric about $n = 0$ as possible:

$$B' = \begin{cases} \left\{ \dfrac{-N}{2} + 1, \ldots, 0, \ldots, \dfrac{N}{2} \right\} & , \text{ if } N \text{ is even} \\[2ex] \left\{ \dfrac{-N+1}{2}, \ldots, 0, \ldots, \dfrac{N-1}{2} \right\} & , \text{ if } N \text{ is odd.} \end{cases}$$

The DFT representation of U is

$$U_j = \sum_{n \in B'} \widehat{U}_n e^{2\pi i n x_j/L} \; ,$$

with DFT coefficients

$$\widehat{U}_n = \frac{1}{N}\sum_{j=0}^{N-1} e^{-2\pi i n x_j/L} U_j = \frac{\Delta x}{L}\sum_{j=0}^{N-1} e^{-2\pi i n x_j/L} u(x_j) \; .$$

The *trigonometric interpolant*[1] of u is

$$IU(x) = \sum_{n \in B'} \widehat{U}_n e^{2\pi i n x/L} \; .$$

[1]You can review ordinary polynomial interpolation in the excellent book *Numerical Methods* by Germund Dahlquist and Åke Björk.

This is the *trigonometric polynomial*, $P(x) = \sum_{n\in B'} a_n e^{2\pi i n x/L}$, that agrees with the known values U_j at the interpolation points x_j. You can estimate derivatives of u by differentiating the interpolating trigonometric polynomial. For example

$$\partial_x u(x) \approx \partial_x IU(x) = \sum_{n\in B'} \frac{2\pi i n}{L} \widehat{U}_n e^{2\pi i n x_j/L} \; .$$

An approximation has *spectral accuracy* if the error is exponentially small when applied to an analytic function. Show that trigonometric interpolation and Fourier differentiation (differentiating $IU(x)$) are spectrally accurate. More precisely, prove the inequalities

$$\max_x \; |u(x) - IU(x)| \le Ce^{-RN} \; ,$$

$$\max_x \; |\partial_x u(x) - \partial_x IU(x)| \le Ce^{-RN} \; .$$

2. (*Discrete Poisson equation*) This problem focuses on possibly the worst commonly used way to solve the Poisson problem with Dirichlet boundary conditions on a square domain in $\mathbb{R}^2$. The code is for the Poisson problem, which is the partial differential equation (PDE)

$$\triangle u = \partial_x^2 u + \partial_y^2 u = f(x,y) \; , \tag{1}$$

when $0 < x < L$ and $0 < y < L$, together with boundary conditions

$$u(x,0) = 0 \; , \;\; u(x,L) = 0 \; , \;\; u(L,y) = 0 \; , \;\; u(0,y) = g(y) \; . \tag{2}$$

The values of u on the boundary of the domain are zero except on the left edge, where they are specified by $g(y)$. The computational mesh will be a lattice of grid points $x_i = ih$ and $y_j = jh$ with $h = \Delta x = \Delta y = L/(N+1)$. The *interior* grid points (x_i, y_j) are the ones with $i = 1, \ldots, N$ and $j = 1 \ldots, N$. These form and $N \times N$ uniform mesh, with N points in each direction.

We approximate the PDE using finite difference approximations to the derivatives

$$\partial_x^2 u(x,y) = \frac{u(x+h,y) - 2u(x,y) + u(x-h,y)}{h^2} + O\big(h^2\big)$$

$$\partial_y^2 u(x,y) = \frac{u(x,y+h) - 2u(x,y) + u(x,y-h)}{h^2} + O\big(h^2\big)$$

This motivates the standard 5 point finite approximation to the Laplace operator

$$Au_{ij} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}}{h^2}$$

The PDE (1) is replaced by the finite difference approximation

$$\frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij}}{h^2} = f_{ij} = f(x_i, y_j) \; . \tag{3}$$

This equation is supposed to hold for each (i,j) of the mesh, which is $1 \leq i \leq N$ and similarly for j. It may be rewritten as

$$u_{ij} = \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}\right) - h^2 f_{ij} \ . \tag{4}$$

These are not closed systems of equations because they refer to (i,j) values corresponding to boundary points, which are $i = 0$ or $i = N+1$ and similarly for j. We get those from the boundary conditions. These give

$$u_{i,0} = u_{i,N+1} = u_{N+1,j} = 0 \ , \ \ u_{0,j} = g_j = g(y_j) \ .$$

We use these to modify the equations (4) when $i = 1$ or $i = N$, or $j = 1$ or $j = N$. For example, the equations for $i = N$ and $2 \leq j \leq N-1$ are

$$u_{N,j} = \frac{1}{4}\left(u_{N-1,j} + u_{N,j+1} + u_{N,j-1}\right) - h^2 f_{Nj} \ .$$

Note that $u_{N,j}$ is not related to the average of the three u values on the right because there are three such values and we still divide by 4. This is an essential feature of the discrete problem. The discrete boundary conditions on the left edge ($j = 1$, and $2 \leq i \leq N-1$) are

$$u_{1,j} = \frac{1}{4}\left(u_{2,j} + u_{1,j+1} + u_{1,j-1}\right) + \frac{1}{4}g_j - h^2 f_{1j} \ .$$

We write the inhomogeneous term on the right using $F_{1,j} = (f_{1,j} - \frac{h^{-2}}{4}g_j)$. The unknowns u_{ij} are seen to satisfy a system of linear equations. Let $V = \mathbb{R}^{N\times N}$ be the vector space of all grid functions. The linear transformation $M : V \to V$ is given as $u \rightsquigarrow v = Mu$ if

$$v_{ij} = \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}\right) \ , \tag{5}$$

if $2 \leq i \leq N-1$ and $2 \leq j \leq N-1$. If $j = 1$ and $2 \leq i \leq N-1$ (the bottom boundary), then

$$v_{ij} = \frac{1}{4}\left(u_{i+1,1} + u_{i-1,1} + u_{i,2}\right) \ . \tag{6}$$

The discrete Poisson problem is to find $u \in V$ that satisfies

$$u = Mu - h^2 F \ , \tag{7}$$

where $F_{ij} = f_{ij}$ unless $i = 1$ (the left boundary), in which case $F_{1,j} = (f_{1,j} - \frac{h^{-2}}{4}g_j)$.

(a) Show that the eigenvectors of M are given by $v_{mn} \in V$ whose components are

$$v_{mn,ij} = C\sin\left(\frac{m\pi x_i}{L}\right)\sin\left(\frac{n\pi y_j}{L}\right) \ . \tag{8}$$

Hints: Observe that these grid functions satisfy discrete homogenous boundary conditions $v_{mn,0,j} = 0$, $v_{mn,N+1,j} = 0$, etc. Therefore, you can use the same definition of M at every (i,j) in the grid. The matrix M acts in the i and j directions independently, so you can do two one dimensional finite difference computations rather than one two dimensional calculation. The eigenvalue is the sum of the two one dimensional eigenvalues.

(b) Show that these eigenvectors are orthogonal in the inner product

$$\langle u, w \rangle_V = \sum_{i=1}^{N} \sum_{j=1}^{N} \overline{u}_{ij} w_{ij} \ .$$

That is, $\langle v_{mn}, v_{m'n'} \rangle = 0$ if $n \neq n'$ or $m \neq m'$ with $1 \leq m \leq N$ and $1 \leq n \leq N$. Hint: First express the sines in terms of complex exponentials. Then factor the double sums into single sums over i or j separately. Then use the orthogonality we proved for single exponential sums earlier.

(c) Write simple approximate formulas for $\lambda_{max}(M)$ and $\lambda_{min}(M)$. "Simple" means that the formulas should represent the $h \to 0$ asympotics. For example, write $\lambda_{max}(M) \approx 1 - \pi^2 h^2/2$ if the exact answer is $\lambda_{max}(M) = \cos(\pi h)$, or $\lambda_{max}(M) \approx N^2$ if $\lambda_{max}(M) = (N+1)^2$.

(d) (*Consistency*) For this part and the next, suppose $g = 0$. Suppose the exact solution is $u(x,y)$ and its values on the grid are $U_{ij} = u(x_i, y_j)$ and suppose that $u(x,y)$ has bounded derivatives up to order 4. The *residual* is what you get when you plug U into the discrete u equations:

$$R = \frac{1}{h^2}(MU - U) - f \ .$$

This is a vector identity in the vector space V. Show that $\|R\|_{l^2(V)} \leq Ch^2$.

(e) (*Stability*) Use what you know about the eigenvalues of M, and the fact that M is a symmetric matrix, to show that

$$\|u - U\|_{l^2(V)} \leq \|R\|_{l^2(V)} \leq Ch^2 \ .$$

The first inequality is called *stability.* The pair of inequalities here is the *Lax stability and consistency argument.*

3. (*Jacobi iteration*) The system of equations (7) need to be solved somehow. They could be solved by *direct* methods such as Cholesky factorization and back substitution. Or they could be solved by *iterative* methods such as the Jacobi method described here. An iterative method produces a sequence of *iterates*, called $u^{(k)}$ here, so that

$$u^{(k)} \to u \ , \quad \text{as } k \to \infty \ .$$

Jacobi iteration is

$$u^{(k+1)} = Mu^{(k)} - h^2 F \; . \tag{9}$$

The *residual* after k iterations is $R^{(k)} = Mu^{(k)} - u^{(k)} - h^2 F$. The *error* after k iterations is $E^{(k)} = u^{(k)} - u$, with u being the exact solution. Your solution algorithm can calculate the residual, but it cannot easily calculate the error.

(a) Show that if $R^{(k)} \to 0$ as $k \to \infty$, then $u^{(k)} \to u$ as $k \to \infty$. Hint: You can use the stability theory of part (2e), or you can derive what you need using the eigenvalues of M.

(b) Show that $R^{(k+1)} = MR^{(k)}$.

(c) You can expand $R^{(k)}$ in the known eigenvectors

$$R^{(k)} = \sum_{m=1}^{N} \sum_{n=1}^{N} a_{mn}^{(k)} v_{mn} \; .$$

Of course,

$$a_{mn}^{(k+1)} = \lambda_{mn} a_{mn}^{(k)} \; .$$

We have implicitly used this already. Show that for large k, $R^{(k)}$ is approximately proportional to the eigenvector corresponding to $\lambda_{max}(M) = \lambda_{11}$ unless $a_{11}^{(0)} = 0$. More precisely, $R^{(k)} \approx \lambda_{11}^k a_{11}^{(0)} v_{11}$ as $k \to \infty$, which might be given even more formally as

$$\frac{\left\| R^{(k)} - \lambda_{11}^k a_{11}^{(0)} v_{11} \right\|}{\left\| R^{(k)} \right\|} \to 0 \quad \text{as} \quad k \to \infty \; .$$

(d) Let W_N be the following measure of the work needed to solve the system by Jacobi iteration:

$$W_n = \# \text{ of iterations so that } \left\| R^{(k+W_N)} \right\| \approx \frac{1}{2} \left\| R^{(k)} \right\| \; ,$$

for large k. This measures the asymptotic convergence rate. Show that $W_N \approx cN^2$ for large N and evaluate c. This makes Jacobi iteration the slowest way to solve the discrete Poisson problem that is sometimes used.

4. (*Poisson solving in C++*) Download the C++ source and the instructions file. The code should run and print some stuff. Note that $u(x,y) = sin(\pi y/L)e^{-\pi x/L}$ is the solution that satisfies the boundary conditions.

(a) It is poor programming practice to *hard wire* parameters in the program. This one has at least one hard wired constant for you to fix. The line: `if ( k%100 == 0 ){` has 100 as a hard wired constant. Replace it with a variable

```
int T\_p;    \\ the number of iterations between printout
```

that you set where the other problem parameters are set near the top of the code. Sometimes you will want it less than 100, sometimes more.

(b) Experiment with various values of N and observe the asymptotic convergence rate. Check that this is in quantitative agreement with the theory of part (3). Find which N values make your code too slow on your computer. You will need larger T to observe the asymptotic convergence rate for larger N.

(c) Add some code to `main` that computes the error $\|u - U\|_{l^2(V)}$. Check the second order computationally. You will have to set T by trial and error so that you have solved the equations accurately for a given N.

(d) The *computational kernel* of the code is the double loop in the procedure `it(...)`. Rewrite this loop to put i in the inner loop and j in the outer loop. Observe that the Jacobi iterations are much faster. (You may have to take large N for this.) The reason has to do with cache misses. Explain.