M607 - First programming Project
Iterative Methods

Peter Monk

Due Oct 5

Hand in code, pictures and comments as requested. Please try to format for minimum use of paper!

1 Gauss-Seidel/SOR

- Using the code from your first homework, write a function with the first line

  ```
  [p,iter,rnorm]=poisson ( p0, rhs, dx, dy, tol, itmax, omega )
  ```

  that performs at most `itmax` iterations of SOR with relaxation parameter `omega`. The iteration should stop when the the maximum norm of the residual is less than `tol`. The other parameters are as follows:

`p0`	The initial guess with the boundary data loaded around the edges of the matrix.
`rhs`	the right hand side matrix (i.e. $rhs(i,j) =_{i,j}$)
`dx`	the mesh size in x
`dy`	the mesh size in y

  Run your code using `omega=1`, `itmax=500` and `tol=1.e-04` to solve $\Delta p = 0$ in the unit square using 10 mesh intervals in the x and y directions (so `dx`=1/10). The boundary data should be $p = 0$ on $x = 1$ and $y = 1$ together with

$$p(x, 0) = \sinh(\pi(1 - x)/2)/\sinh(\pi/2), \quad 0 \le x \le 1,$$

  and

$$p(0, y) = \cos(\pi y/2), \quad 0 \le y \le 1.$$

  Draw a plot of the result using the matlab command `mesh`. You should see the result in Fig. 1 (perhaps after rotating the picture). The exact solution is known in this case

$$u_{exact}(x, y) = \cos(\pi y/2)\sinh(\pi(1 - x)/2)/\sinh(\pi/2)$$

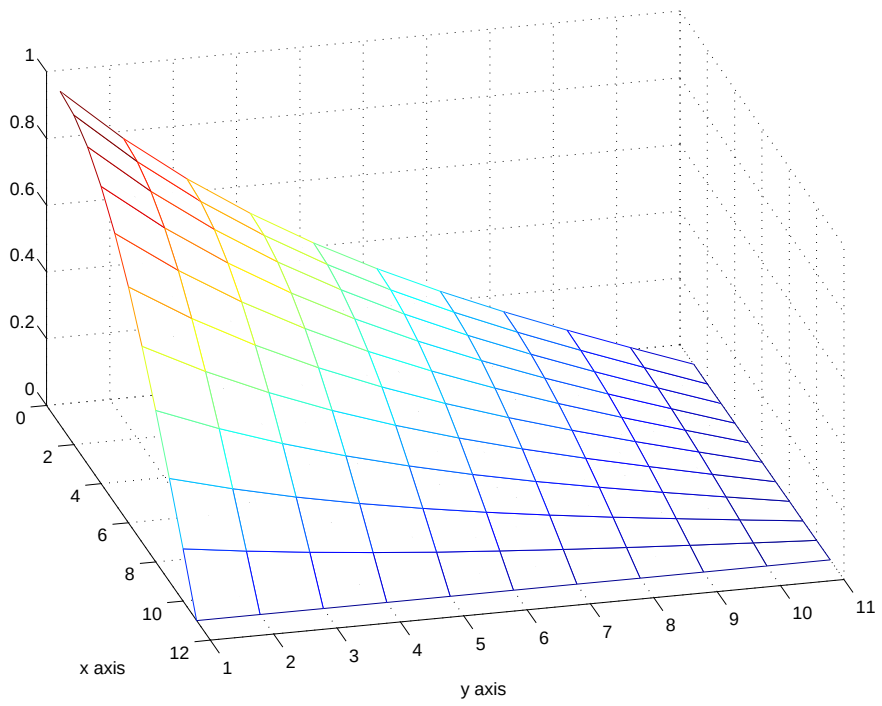

Figure 1: Example using 10 mesh intervals

Compute the maximum error at the mesh points in your SOR solution. You should find it to be 2.18×10^{-4} (this will help you check your code is right)! Rerun the code with `omega=1.7` and observe the decrease in number of iterations needed (how many iterations are needed in each case?). For a smooth solution (i.e. one have four bounded derivatives in x and y) we expect that the truncation error (i.e. error due to finite differences) should behave like

$$e_{trunc} \approx C(dx)^2, \text{ so } \log(e_{trunc}) \approx 2\log(dx) + \log(C).$$

This power law is said to be second order convergence.

Use your SOR function to compute the solution for $5, 10, 15, 20$ subintervals and plot the maximum error in each case against dx using a log-log plot. What is the slope of the line? Does it support theory? The total error in the calculation is $e_{trunc} + e_{it}$ where e_{it} is the error due to stopping SOR. Make sure that you have set a tolerance low enough that e_{it} is small compared to e_{trunc} (in real calculations you want to choose the tolerance so that the two errors are roughly the same so that you do as little work as possible....).

- Now change the boundary data to $p = 0$ at $y = 0$, $y = 1$ and $x = 1$, but keep the cos boundary condition at $x = 0$. Compute the finite difference solution using 12 subintervals and plot the result. You should see that the solution varies very rapidly near $x = y = 0$ and in fact the derivative is unbounded as you approach the corner.

The exact solution is given by

$$p(x,y) = \sum_{n=1}^{\infty} \frac{2n}{\pi(n^2 - 1/4)\sinh(n\pi)} \sin(n\pi y)\sinh(n\pi(1-x))$$

Write a function with the first line

```
function pe=exact(x,y,N)
```

where N is the number of terms in the series to be added. Experiment with N until you find a reasonable choice that computes the solution at $x = 1/2, y = 1/2$ to an error of at most 1.e-4. Now solve the finite difference equation for 4,8,12,16,20,24,28,32 subintervals and plot a log-log error graph using the absolute value of error at $x = 1/2, y = 1/2$ only. Is the convergence consistent with power law as before? Are we seeing the same convergence rate as before?

2 GMRES

Here is a small exercise that demonstrates some properties of GMRES (use the matlab `gmres` function) from the book of Trefethen and Bau.

- Create a random matrix A using

  ```
  m=200;A=2*eye(m)+0.5*randn(m)/sqrt(m);
  ```

  Compute the eigenvalues of A using the matlab command `eig` and plot each eigenvalue as a $*$ in the complex plane (you will need the commands `real` and `imag`). Note that the eigenvalues are clustered in circle of radius $1/2$ around $z = 2$. Hand in a plot of the eigenvalues. Now run GMRES using (in matlab notation) `b=ones(m,1)`. Plot the norm of the residual at each step (using `semilogy`). What is the slope of the line? This is an example of rapid convergence (without even a preconditioner) since the eigenvalues are well clustered.

- Modify the matrix A in the previous example by computing $\theta_k = k\pi/(m-1), \leq k \leq m$ and defining the diagonal matrix D with $d_{k,k,} = (-2+2\sin(\theta_k))+i\cos(\theta_k)$. Then setting $B = A + D$. Now as before, plot the eigenvalues of B (they are no longer clustered) and solve $B\vec{x} = \vec{b}$ using GMRES. Increase the allowed number of iterations to 40 and plot the residual. Convergence of GMRES critically depends on clustering eigenvalues and that can be accomplished by using a good preconditioner (its not easy to say what is a good preconditioner in thus case because the matrix is random).