MTH 453/553 – Homework 2

1. Consider the following two point boundary value problem:

$$u''(x) = -\sin(x)$$
$$u(0) = 0,\ u(\pi) = \pi$$

   (a) Using the second order, centered finite difference scheme discussed in class, discretze this two point boundary value problem. What is the resulting system of linear equations?

   (b) Compute the error constants in the truncation error arising in the finite difference approximation to this problem. Use this to estimate h required to achieve a grid function 2-norm of the global error less than 10^{-6}.

   (c) Let N be the number of intervals in the discretization ($m = N - 1$ is the number of interior points and $h = 1/M$ is the mesh width). Download the codes `TPBVP_GE.m` and `GEpivot.m` from my website. Each m-file contains a thorough description of the contents of the file. `TPBVP_GE` constructs the matrix A and the right hand side vector f of the linear system, and then calls the function `GEpivot` to solve this system using Gaussian elimination. The computed solution and the exact solution are plotted on the same graph and the grid function 2-norm of the global error is output to the Matlab prompt. Figure out how these codes work. (You may of course write your own codes.)

   Use these codes to solve this linear system with $N = 5$ and output the results. How large a system can your computer solve with Gaussian elimination?

   (d) The code `TPBVP_GE` outputs the grid function 2-norm of the global error. Explore the order of convergence in this grid function 2-norm and comment on your results. Change the grid function 2-norm to the vector 2-norm. What can you say about the order of convergence?

   (e) Change the norm to the ∞-norm and explore the order of convergence in the ∞-norm. Comment on your results.

   (f) Download the codes `TPBVP_Doo.m` and `tridiag.m` from my website. Each m-file contains a thorough description of the contents of the file. `TPBVP_Doo` constructs the matrix A by storing just the nonzero entries in three different vectors. It then calls the function `triadiag` to solve this system using a variant of Gaussian elimination called the *Doolittle method*. The computed solution and the exact solution are plotted on the same graph and the grid function 2-norm of the global error is output to the Matlab prompt. Figure out how these codes work. (You may of course write your own codes).

   Use these codes to solve the linear system with $N = 1000$ and output the results.

   (g) Compare the runtime for the case where the factorization needs to be computed to the case where the factorization is given (`iflag` $= 0$ and 1, respectively). Also, compare runtimes of Doolittle with Gaussian elimination. Comment on your results.

2. Consider Laplace's equation posed on the unit square with Dirichlet boundary conditions:

$$\nabla^2 u = 0 \text{ in } \Omega = (0,1) \times (0,1) \tag{1}$$
$$u = g \text{ on } \partial\Omega \tag{2}$$

where g is defined as

$$g(x,y) = \begin{cases} \sin(\pi x), & \text{if } y = 1 \text{ and } 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

(a) Verify that the solution to this boundary value problem is

$$u(x,y) = \frac{\sin(\pi x)\sinh(\pi y)}{\sinh(\pi)} \tag{4}$$

where we recall that the hyperbolic sine function is defined as $\sinh(z) = (\mathrm{e}^z - \mathrm{e}^{-z})/2$.

(b) Download the code `lap2d_jacobi.m` from my website. This code implements the Jacobi method on the problem described above using a grid with $(m+2) \times (m+2)$ gridpoints in which the unknowns constitute an $m \times m$ mesh with $h = 1/(m+1)$ (since the boundary values are known). It produces plots of the analytical solution, the solution obtained by directly solving the discrete matrix system, and the solution obtained through Jacobi iteration of the matrix system. As output parameters, it computes the error between the iterative and exact solutions, as well as the error between the iterative and direct solutions.

Using this code as a template, you should implement the following in MATLAB:

i. Modify my code to use the Gauss-Sidel method to solve the linear system, call it `lap2d_gs.m`.

ii. Modify my code to use the SOR method to solve the linear system, call it `lap2d_sor.m`. For the SOR method use the optimal relaxation parameter

$$\omega = \frac{2}{1 + \sin(\pi h)}.$$

(c) In a single window, plot the convergence history (the error versus the number of iterations) for all three methods. Do this for $m = 7$ ($h = 1/8$) and also for $m = 15$ ($h = 1/16$). The error should be relative to the analytical (exact) solution. Use logarithmic (base 2 preferably) scaling for the y-axis. See the sample file `plothistory.m` on my website. Based on these two plots, discuss the differences between the methods. For example, which method has the minimum error? Which method is fastest to converge?

(d) We can see in the plots from the previous question that after a certain number of iterations for each of the methods, the error stops improving. Why isn't the error going to zero? What type of error is left? What is the order of this error

with respect to h? To help understand what is going on, consider computing the error between the exact and direct solutions, or plotting the error between the direct and iterative solutions versus the iteration number. Again, use a logarithmic scale (base 2) for the plot, and recall that *machine epsilon* for double precision is 2^{-52}.

(e) You should be able to estimate the "left-over error" described in the previous problem as a function of h (careful not to round too much!). Implement a stopping criteria in the codes for each of the methods by changing the error tolerance to be this amount (why is this not reasonable to do in general?!). On your own, re-do the plots above and check whether you have done this correctly. Then run your codes for each of the following: $m = \{3, 7, 15, 31, 63\}$. On the same graph, make a plot for each method of k (the number of iterations required to satisfy your stopping criteria) versus m. Compare your results to the theoretical predicitions. It may be useful to also plot $k/log_2(m)$ versus m with logarithmic scaling on both axes (use `axis('equal')` as well).

(f) Using the method of your choice, do a grid-refinement analysis. That is, using $m = \{3, 7, 15, 31, 63\}$, plot the final error versus h (using logarithmic scaling on both axes and `axis('equal')`) to verify that the global truncation error is $O(h^2)$.

3. Suppose that we use the *red black ordering* to order the unknowns in the two-dimensional finite difference system represented by the 5-point stencil for Poisson's problem. This ordering is significant because all four neighbors of a red point are black points, and vice versa, and leads to a matrix structure as shown is equation (3.13). What is the matrix H for the grid shown in Figure 3.2(b)?