APMA 4302 Methods - Homework 2

Marc Spiegelman

February 11, 2026

1. (**10 pts**) Some preliminaries: the condition number of a matrix A is defined as

$$\kappa(A) = \|A\|\|A^{-1}\|$$

where $\|A\|$ is a matrix norm with properties $\|A\mathbf{x}\| \leq \|A\|\|\mathbf{x}\|$.

   (a) Assuming A is invertible, show that if $A\mathbf{u} = \mathbf{b}$ then the relative error in $\mathbf{u}$ is bounded by the condition number times the relative error in $\mathbf{b}$, i.e.

$$\frac{\|\mathbf{u} - \hat{\mathbf{u}}\|}{\|\mathbf{u}\|} \leq \kappa(A)\frac{\|\mathbf{b} - \hat{\mathbf{b}}\|}{\|\mathbf{b}\|}$$

   where $\hat{\mathbf{u}}$ is the perturbed solution corresponding to a perturbed right-hand side $\hat{\mathbf{b}}$.

   (b) Use a similar argument to show Eq. 2.11 in the Beuler textbook, which states that if $A\mathbf{u} = \mathbf{b}$ and $A\mathbf{e} = \mathbf{r}$ then

$$\frac{1}{\kappa(A)}\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|} \leq \frac{\|\mathbf{e}\|}{\|\mathbf{u}\|} \leq \kappa(A)\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$$

   where $\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}}$ is the error in the solution and $\mathbf{r} = \mathbf{b} - A\hat{\mathbf{u}}$ is the residual. Provide an interpretation of this result.

2. (**10 pts**) Let $A = \frac{1}{h^2}\text{tridag}\left[-1, 2, -1\right]$ be the symmetric positive definite matrix given by the centered finite difference discretization of the second derivative $-u''(x)$ on the interval $x \in [0, 1]$ with homogeneous Dirichlet boundary conditions. Here $h = 1/m$ is the uniform grid spacing and A is an $(m-1) \times (m-1)$ matrix.

   (a) Show that the eigenvectors of A are given by $\mathbf{v}_j = \sin(j\pi x_i)$ where $x_i = ih$ for $i = 1, \ldots, m-1$.

   (b) find the corresponding eigenvalues λ_j.

   (c) Show that the condition number of A grows like $\kappa(A) \sim O(m^2)$ as $m \to \infty$.

3. (**20 pts**) Consider the following boundary value problem (BVP):

$$-u''(x) + \gamma u(x) = f(x), \quad x \in [0, 1]$$

with dirichlet boundary conditions. Do the following:

(a) Find $f(x)$ given the manufactured solution $u(x) = \sin(k\pi x) + c\left(1 - \frac{1}{2}\right)^3$, where k is a positive integer and c is a real constant.

(b) Modify the code `tri.c` from Chapter 2 in `p4pdes` to solve the above BVP using `PETSc`. Your code should have the following features:

- use `PETSc` options handling so that the code can be run from the command line with

```
$ mpiexec -np P ./bvp -options_file options_file
```

  where the `options_file` is a text file of petsc commands that will be provided.

- The code should assemble the matrix and right-hand side vector in parallel using `nP` processes. Be careful to handle the boundary conditions correctly to preserve the symmetry of the matrix. This can be done by calling the `MatZeroRowsColumns` function in `PETSc` with the appropriate arguments after assembling the matrix A, the exact solution u_{exact}, and the right-hand side vector f.

True sltn vs discrete sltn

- The code should compute the relative error in the solution and print it to the console.

- Rebuild your PETSc with the additional flags `-download-hdf5` and include the following code to allow named output of the solution vectors, true solution and rhs in HDF5 format:

```
// output the solution, rhs, and exact solution to an HDF5 file
PetscCall(PetscViewerHDF5Open(PETSC_COMM_WORLD, "bvp_solution.h5",
          FILE_MODE_WRITE, &viewer));
PetscCall(PetscObjectSetName((PetscObject) uexact, "uexact"));
PetscCall(PetscObjectSetName((PetscObject) f, "f"));
PetscCall(PetscObjectSetName((PetscObject) u, "u"));
PetscCall(VecView(f, viewer));
PetscCall(VecView(u, viewer));
PetscCall(VecView(uexact, viewer));
PetscCall(PetscViewerDestroy(&viewer));
```

- I will also provide a Python script `plot_bvp.py` that reads the HDF5 file and plots the solution, true solution, and error.

(c) Modify `plot_bvp.py` to plot the convergence of the error as a function of h with $\gamma = 0$, $k = 1, 5, 10$ $m = 40, 80, 160, \ldots, 1280$ and determine the order of convergence.

4. (**10 pts**) use the default options file to determine the number of iterations required to converge using: and explain your results

(a) jacobi pre-conditioned richardson (i.e. `-ksp_type richardson -pc_type jacobi`)

(b) unconditioned conjugate gradient with 1 processor (i.e. `-ksp_type cg -pc_type none`)

(c) unconditioned conjugate gradient with $c = 0$ (explain your results)

have one line that patches all boundary conditions together petsc

(d) icc pre-conditioned conjugate gradient with 1 processor (i.e. `-ksp_type cg -pc_type icc`) (explain this result)

(e) block Jacobi pre-conditioned conjugate gradient with 4 processors (i.e. `-ksp_type cg -pc_type bjacobi -pc_sub_type icc`)

(f) mumps direct solver (i.e. `-ksp_type preonly -pc_type lu -pc_factor_solver_type mumps`) with 1 and 4 processors .

Performance trade-offs:
-work vs convergence

make it work, see where it breaks, see him

Convergence plot:

-if its good and logarithmically linear then