Lab 5 - PETSc and Krylov Methods

NERS/ENGR 570 Fall 2020
October 2nd, 2020
DUE BY: Oct. 12th, 2020

Exercise 1 (30 pts) - Install PETSc

In this exercise you are to download, configure, build, test your build, and install PETSc. The purpose of this exercise is to assess how well you have learned the material presented in Lectures 4-5, and Labs 1-4. Your subsequent PETSc installation will be used for the remaining exercises.

For your PETSc installation, use your class directory space located at:
`/gpfs/accounts/ners570f20_class_root/ners570f20_class/$USER`.

1. Download PETSc. What command did you use?
2. Load any modules you think would be prerequisites for PETSc. Document your loaded modules at configure-time.
3. The next step is to configure PETSc.
   - Since you wish to use PETSc--rather than contribute source to it-- you desire your library installation to run as fast as possible.
   - You also wish to **include debug symbols** in case you develop against the library incorrectly.
   - What command did you use to configure PETSc?

4. Also configure PETSc to have examples and tests enabled.
5. Run the tests. What command did you use? Did any fail? If so, which ones?
6. Install your PETSc build. What command did you use?
7. In the following exercises you will be working with PETSc's tutorial example 15. The tutorial is provided in both C and Fortran. Source code for the example can be viewed at the links below.
   https://www.mcs.anl.gov/petsc/petsc-3.11/src/ksp/ksp/examples/tutorials/ex15.c.html
   https://www.mcs.anl.gov/petsc/petsc-3.11/src/ksp/ksp/examples/tutorials/ex15f.F90.html
8. Fortunately for you, the example we'll be working with is installed! Can you find the source file in your install directory? Where did you find the example/tutorial?

   To compile the example, go to the directory with the file. In this directory you should run `make` with the appropriate target name for the example you want to work with (C or Fortran).

9. Additional notes for the next exercises
   - If you modify the source, run `make clean` before you run `make` to ensure that your previous compilations do not affect your latter compilation.
   - You will be running an MPI executable in your script (and we have not covered this in detail in lecture yet). So here is the base command you will use:
     - **`$ mpiexec -n 2 ./ex15`**
   - To report times use the "real" time reported by the time command e.g. **`$ time mpiexec -n 2 ./ex15`** For the various cases you will run, many of the options can be controlled from the command line. Some of the useful command line options are:
     - **`-ksp_type`**
     - **`-pc_type`**
     - **`-mat_view::ascii_info`**
     - **`-log_view`**

     For more information, see PETSc's documentation (see chapter 4) for this assignment. (http://www.mcs.anl.gov/petsc/petsc-current/docs/manual.pdf)
     Try to use `-log_view` and interpret the output information to help understand the meaning of the options provided.

   - The matrix size can also be changed using command line options. In this example, the matrix is set from a 2-D five-point stencil for an `m` by `n` grid, so `A` is always a square matrix of size `m*n`. To change the values of `m` and `n`, use the intuitive command line options:
     - **`-m`**
     - **`-n`**

Extra Credit (5pts)
Write a module file for your PETSc install.

Exercise 2 (40 pts) - Comparison of Krylov Methods

In lecture 09 we covered Krylov methods and showed you some comparisons using a Jupyter notebook. In this exercise we will compare the different Krylov methods as they are implemented in PETSc. Be sure to exclude any preconditioning by using the `-pc_type none` option.

Grid size	Number of Iterations (and run time)				
	CG	BiCGSTAB	GMRES (no restart*)	GMRES (restart=30)	GMRES (restart=200)
m=n=8	10 (00:00:00.164)				
m=n=32					
m=n=128					
m=n=256					
m=n=512					
m=n=1024					

*Let PETSc build the subspace for GMRES up to size m=n.

Discuss the observed results. Which method scaled best with grid size? How did the number of restart vectors affect the number of iterations?

Hint: the different Krylov methods may be chosen with a command line option. The keyword and options are in the User's manual.

Exercise 3 (40 pts) - Preconditioners

In lecture 09 we covered Krylov methods and spent some time discussing how they converge and that this was related to the condition number. We also mentioned that matrices resulting from typical finite difference discretizations have an unbounded condition number as the problem size goes. This means that Krylov methods may struggle for large problems without a preconditioner. The purpose of this exercise is to observe how preconditioners can help this situation.

In this exercise we ask you to perform an experimental analysis using PETSc and look at the effectiveness of various preconditioners, both in terms of performance and ability to reduce the original matrix's condition number.

Part A -Preconditioner Performance

Fill in the following table and discuss the results. When running your calculations to get timing info, be sure to suppress extraneous output (e.g. `-log_view`, `-mat_view`, etc.) and run on the compute nodes.

Grid size	Number of GMRES Iterations (and run time)				
	Jacobi	SOR	Incomplete LU	Incomplete Cholesky	Algebraic Multigrid
m=n=8	10 (00:00:00.164)				
m=n=32					
m=n=128					
m=n=256					
m=n=512					
m=n=1024					

What options did you use to run each preconditioner?
Which preconditioner took the most time?
Which preconditioner performed best?

Create 2 plots. One for run time vs grid size, and the other for iterations vs grid size. Discuss the results.

Part B - Estimate the condition number of various grid sizes

The condition number can be edited from GMRES with the appropriate command line options. These options are not well documented in the manual, but they are documented. Try doing a bit of searching (especially on the PETSc website) to find the discussion of how to edit condition numbers. For the condition number estimates, take the maximum value over all iterations.

Fill in the following table.

Grid size	Estimate of Condition Number for Various Preconditioners					
	None	Jacobi	SOR	Incomplete LU	Incomplete Cholesky	Algebraic Multigrid
m=n=8						
m=n=32						
m=n=128						
m=n=256						
m=n=512						
m=n=1024						

Which preconditioner had the most efficient reduction in the condition number?
Does this corroborate or contradict the results comparing the number of iterations?

Extra-credit (10 pts)

Repeat the calculations for CG without a preconditioner. Verify that the error adheres to the following expression for rate of convergence of CG for a symmetric positive definite matrix:

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k$$

Deliverables and how to submit:

For the overall deliverable you may submit a single word doc or pdf based on these instructions. Name the file as follows:

<username>_Lab5.doc (or docx or pdf).

For exercise 1, include the commands you executed for each part of the exercise for downloading, configuring, compiling, testing, and installing.

For exercise 2, include 1 sample slurm script that you used.

Include answers for all questions.

Please upload your doc/docx/pdf file to canvas.

For the Extra-Credit include this in your submission with this document.