# Notes on MG Practical Exercises

June 9, 2021

The exercises are presented as a series of Jupyter notebooks that make use of Numpy, Scipy, and pyAMG to demonstrate the concepts covered in the lectures. Some of the notebooks have explicit exercises suggested in them (e.g., changing relaxation parameters or schemes) and some just provide information and give you a chance to "tweak" the settings to look at finer or coarser grids, or look at other aspects of the methods. A rough outline of the material is below:

## MG Basics - one-dimensional Poisson

**01-model-problem:**

- Sets up problem, computes eigenvalues and eigenvectors. Plots some modes
- In the last box, can change which mode is plotted (e.g., plot mode 50 instead of mode 3) to see smooth versus oscillatory modes
- Note shifting and scaling, that is relevant when we get to Jacobi (but need to remember that $I - \frac{1}{2}A$ is just *unweighted* Jacobi iteration matrix, then scale that)

**02-relaxation-Jacobi:**

- Looking at what Jacobi does for smooth errors, then oscillatory
- In Box 4, can change update to `eold[3]` to see different behaviour
- In Box 7, can toggle test between 'random' and 'smooth'
- in Box 9, can change $\omega$ and see effects. Interesting to take $\omega = 1$ and see slow reduction of oscillatory modes

**03-smooth-error:**

- Can vary $\omega$ in Box 4 (and see the effect in later boxes)
- It may be useful to set $\omega = 1$ here

**04-coarse-modes:**

- In Box 2, can vary $m$ and $k$.
- 3 interesting cases:
  - when $k$ is small compared to nc (good coarse representation)
  - when $k$ is close to nc (zero coarse-grid mode for k = nc-1)
  - when $k$ is close to nf (bad coarse-grid representation)

**05-interpolation:**

- Box 3 shows effects of linear interpolation.
- You can vary the coefficient in the sine term defining vc in Box 3
- Last plot is of each column of interpolation

**06-multigrid-two-level:**

- Look at modes in uc in Box 6 and uf in Box 7 (you can adjust $k$ in Box 6 to get finer grids)
- Can modify number of pre- and post-relaxations in 2-level cycle, as well as $\omega$ in definition of relaxation.
  - How do these affect performance?
  - Are two (1,0) cycles better or worse than one (1,1) cycle?
  - What about two (1,1) cycles vs. one (2,2) cycle, etc?
- Note that for 2-level cyles, the distinction between pre- vs. post-relaxation doesn't really matter. Can compare (2,0), (1,1), and (0,2) cycles here to see this.

**07-multigrid-two-level-matrix-form:**

- This is really the same algorithm as in the previous notebook, just with everything in matrix form

**08-multigrid-v-cycle:**

- Compare 2-grid, 3-grid, and full V-cycle for this example
- Note convergence difference between total error and the algebraic error.
- Difference in performance as we vary $\omega$ again
- Coding challenge: Change weighted Jacobi to Gauss-Seidel.

## MG Extensions - two-dimensional problems

**09-multigrid-2d:**

- Construction of matrix is hard to follow, but notice how structure changes when you change (4,4) in second-to-last line of Box 2
- Again can experiment with factor of 4/5 in Jacobi relaxation
- Can experiment with frequencies in uc in Box 6 and u in Box 7
- Look at different relaxation and $\omega$'s in Box 8 and in Boxes 10 and 11 (these are the questions after Box 12)

**10-multigrid-anistropy:**

- Consider effects of $\epsilon$ on relaxation (big, small, O(1))
- Try to understand idea behind Fourier tests
- Note that you can't fix this by changing $\omega$ or relaxation from Jacobi to GS

**11-multigrid-2d-line-relaxation:**

- Main point is in Box 6, switching between point and line relaxation (and directions in line relaxation), as $\epsilon$ varies
- Same for Box 7, in 2-grid cycle
- In Box 11, you can try everything again with lines in opposite orientation

**13-rotated-anisotropy:**

- Demonstrates that you can get good performance for rotated anisotropic diffusion problems using standard interpolation and alternating direction line relaxation
- Note that the main loop is *very* slow to run (in Box 7). You can view the complete results, but if you want to experiment with the methods, it may be better to sample fewer values in the two arrays elist and thlist

## Algebraic Multigrid

Note that these notebooks rely on a package that isn't present in the standard Colab environment, so you need to install it in the virtual machine each time you load a notebook. This takes a minute and isn't persistent when you close/reload a notebook.

**12-AMG-coarse-mesh:**

- Main point is visualization here, but note that you can get two model problems by swapping the data

**14-anisotropy-cf-amg:**

- Provides a general intro to using pyAMG

**15-CF-AMG-interpolation:**

- Some visualization of what happens in CF AMG
- Most interesting might be to vary $\theta$ in Box 5 and see how coarse grid changes

**16-aggregation:**

- Visualizing aggregates in SA, for structured and unstructured graphs
- Comparison of hierarchies for CF and SA (matches lecture notes)

**17-smoothed-aggregation-1d:**

- Small 1D Poisson problem for visualizing unsmoothed vs. smoothed interpolation operators
- Also would be interesting to plot corresponding columns of $P$ and $T$ on same graph
- Could look at larger problems

**18-components-of-smoothed-aggregation:**

- Note that the graph of $A$ is just the mesh here, for the case of Poisson discretized with linear FEM on triangles in 2D
- We are setting near null space here - you could try something far from the vector of ones, and see that it is bad!
- Note that not all connections are "strong"
- Aggregates not uniformly shaped. "Fingering" can be a problem in practice

**19-AMG-advanced-options-anisotropy:**

- Using just the constant vector doesn't work so well, so use more
- Adaptive MG methodology exists to find vectors needed to improve performance
- Several other "advanced" techniques to improve performance here

**20-AMG-advanced-options-nonsymmetric-flow:**

- Standard tools fail for (strongly) convective flow example
- Why? Jacobi doesn't converge. Try something else
- Using *Kaczmarz* stops divergence (provably), but doesn't give good convergence
- Now add more features: evolution strength, improving candidates, energy minimization, Krylov wrapper
- These are "real" options that we use with SA-AMG for hard problems

**21-AMG-advanced-options-systems-elasticity:**

- Rigid body modes of linear elasticity needed to get good convergence
- Again, Krylov wrapper helps