Math 126 Numerical PDEs: Homework 1—debriefing

January 16, 2012

1. [4 points: 2+1+1] See Campbell's soln. Note Nul A = Span $\{\mathbf{u}\}$.

2. [2 pts] See Lin's soln. Notice we need existence of an eigenvector with eigenvalue $\rho(A)$ (this follows from compactness in general).

3. [6 pts: 2+2+2] Knowing $\|A\mathbf{x}\|$ for some $\|x\| = 1$ gives you a *lower bound* on $\|A\|_2$, hence σ_1. For example use $\mathbf{x} = [0, 1, 0, 0, \ldots, 0]^*$ or simply $\mathbf{x} = [1, 0, 0, \ldots, 0]^*$. Jeff achieved the greatest lower bound, using $\mathbf{x} = [1, 1, \ldots, 1]^*$ and an improved estimate from the worksheet. But you were not graded on these tweaks—I just wanted any nontrivial lower bound that shows the exponential growth.

   Here's my code:

```
m=53; A = toeplitz([1 zeros(1,m-1)], [1 2 zeros(1,m-2)]);
x_true = randn(m,1);     % note involves non-integers
b = A*x_true;
x = A\b; norm(x-x_true)/norm(x_true)
```

   The result is about 0.03, not huge but still O(1).

   Note the easiest way to assess error in the vector $\mathbf{x}$ is via the (relative) error norm. This discards potentially-interesting info about which coeffs are wrong and which right (here error varies with index).

   Campbell and Brad did plots of relative error 2-norm vs m which showed lovely exponential blow-up.

4. [4 pts] Note that if the number of loops is p, this is equivalent to saying that roughly m^p calculatoins are needed, i.e. the complexity is $O(m^p)$. (You didn't have to write out actual C codes, or find sneaky ways to write less loops).

5. [4 pts: 2+1+1] I explicitly asked for the number n for single and double, which are 16777217 and 9007199254740993 respectively. Persuading Matlab to print all digits of the latter is necessary (e.g. via `format long e`)

   `(2^53+1)-2^53` gives 0, but `(2^53)-(2^53-1)` gives 1

   `single(2^24+1)-single(2^24)` gives 0, etc. Cute.

6. [5 pts] I left it up to you to decide how dense in m to sample, or if to average several trials with different random matrices (this isn't needed since the SVD is pretty much the same for any matrix). Notice the transition to asymptotic $O(m^3)$ happened at different places on different people's CPUs (controlled probably by whether matrix and workspace fits in L2 cache). Down near $m = 100$ the power law is closer to $O(m^2)$. Vipul's a typical plot. However, some of you (Jeff, Brad) got slopes closer to 4 even at the top of the requested range, which is weird.

   My code: please study it for simplicity, and note the fact that it is very easy to change the parameters. E.g. if you wanted to go up to 2000, you only have to change *one number* in the code:

```
t=[]; ms=100:100:1000;
for i=1:numel(ms); a=randn(ms(i)); tic; svd(a); t(i)=toc; end
figure; loglog(ms, [t; 3e-9*ms.^3], '+-'); xlabel m; ylabel('t (sec)');
```

BONUS: as m increases, the dense matrix and needed workspace will eventually fill RAM and require swapping into the hard-disk, which will require radically longer times (10 or 100 times slower).

7. [5 pts: 3 graphs + 2 explanation]

   This was the one people understand least well. It requires a little thought. See Brad's nice answer here.

   Terms in the expanded poly are as large as 10^5 so rounding errors are ε_{mach} times this, *i.e.* about 10^{-11} absolute errors are produced, easily visible since vertical scale is 10^{-10}. You could test this by playing around. Notice there are massive cancellations (subtractions), *i.e.* condition number wrt coefficients of poly, or wrt x, is huge. However, taking ninth power of $(x-2)$ gives high *relative* accuracy in the answer, by stability of addition and multiplication. It is not merely that expanding the poly involves more flops (45) than factoring it (10 flops); if that were the case, errors would be similar.

   Note condition number wrt the number 2 or wrt x is modest, about $2/|2-x| \sim 10^2$ in a typical region of the plot. (Note the same math problem can have a different condition number because we listed *all* 10 coeffs as input in the first case, and only one coeff in the latter case.)