Week 5: Friday, Sep 25

Iterative refinement revisited

At the end of last lecture, we discussed *iterative refinement*:

$$x_{k+1} = x_k + \hat{A}^{-1}(b - Ax_k).$$

We had shown that if $\|\hat{A}^{-1}E\| < 1$ for $E = \hat{A} - A$, then iterative refinement converges. At least, we get convergence in exact arithmetic. In floating point arithmetic, we actually compute something like

$$x_{k+1} = x_k + \hat{A}^{-1}(b - Ax_k + \delta_k) + \mu_k,$$

where δ_k is an error associated with computing the residual, and η_k is an error associated with the update. If $\|\delta_k\| < \alpha$ and $\|\mu_k\| < \beta$ for all k, and if $\|A^{-1}E\|$ is not too close to 1 (e.g. $\|A^{-1}E\| < 1/2$, then we find something like

$$\|x_k - x\| \leq \|A^{-1}E\|^k \|x_k - x\| + C\left(\alpha\|A^{-1}\| + \beta\right)$$

where C is a not-too-large constant. Showing this (or at least a piece) is left as an exercise on the next homework[1]. If we evaluate the residual in the obvious way, we typically have

$$\begin{aligned} \alpha &\leq c_1 \epsilon_{\text{mach}} \|A\| \|x\|, \\ \beta &\leq c_2 \epsilon_{\text{mach}} \|x\|, \end{aligned}$$

for some modest c_1 and c_2; and for large enough k, we end up with

$$\frac{\|x_k - x\|}{\|x\|} \leq C_1 \epsilon_{\text{mach}} \kappa(A) + C_2 \epsilon_{\text{mach}}.$$

That is, iterative refinement leads to a relative error not too much greater than we would expect due to a small relative perturbation to A; and we can show that in this case the result is backward stable. And if we use *mixed* precision to evaluate the residual accurately enough relative to $\kappa(A)$ (i.e. $\alpha\kappa(A) \lesssim \beta$) we can actually achieve a small *forward* error.

[1] *Hint*: Look up Neumann series in the book.

Condition estimation

Suppose now that we want to compute $\kappa_1(A)$ (or $\kappa_\infty(A) = \kappa_1(A^T)$). The most obvious approach would be to compute A^{-1}, and then to evaluate $\|A^{-1}\|_1$ and $\|A\|_1$. But the computation of A^{-1} involves solving n linear systems for a total cost of $O(n^3)$ — the same order of magnitude as the initial factorization. Error estimates that cost too much typically don't get used, so we want a different approach to estimating $\kappa_1(A)$, one that does not cost so much. The only piece that is expensive is the evaluation of $\|A^{-1}\|_1$, so we will focus on this.

Note that $\|A^{-1}x\|_1$ is a convex function of x, and that $\|x\|_1 \leq 1$ is a convex set. So finding

$$\|A^{-1}\|_1 = \max_{\|x\|_1 \leq 1} \|A^{-1}x\|_1$$

is a convex optimization problem. Also, note that $\|\cdot\|_1$ is differentiable almost everywhere: if all the components of y are nonzero, then

$$\xi^T y = \|y\|_1, \text{ for } \xi = \text{sign}(y);$$

and if δy is small enough so that all the components of $y + \delta y$ have the same sign as the corresponding components of y, then

$$\xi^T(y + \delta y) = \|y + \delta y\|_1$$

More generally, we have

$$\xi^T u \leq \|\xi\|_\infty \|u\|_1 = \|u\|_1,$$

i.e. even when δy is big enough so that the linear approximation to $\|y+\delta y\|_1$ no longer holds, we at least have a lower bound.

Since $y = A^{-1}x$, we actually have that

$$|\xi^T A^{-1}(x + \delta x)| \leq \|A^{-1}(x + \delta x)\|,$$

with equality when δx is sufficiently small (assuming y has no zero components). This suggests that we move from an initial guess x to a new guess x_{new} by maximizing

$$|\xi^T A^{-1} x_{\text{new}}|$$

over $\|x_{\text{new}}\| \leq 1$. This actually yields $x_{\text{new}} = e_j$, where j is chosen so that the jth component of $z^T = \xi^T A^{-1}$ has the greatest magnitude.

Putting everything together, we have the following algorithm

```
% Hager's algorithm to estimate norm(A^{-1},1)
% We assume solveA and solveAT are O(n^2) solution algorithms
% for linear systems involving A or A' (e.g. via LU)

x = ones(n,1)/n;    % Initial guess
while true

  y  = solveA(x);   % Evaluate y = A^{-1} x
  xi = sign(y);     %  and z = A^{-T} sign(y), the
  z  = solveAT(xi); %  (sub)gradient of x -> \|A^{-1} x\|_1.

  % Find the largest magnitude component of z
  [znorm, j] = max(abs(z));

  % znorm = |z_j| is our lower bound on |A^{-1} e_j|.
  % If this lower bound is no better than where we are now, quit
  if znorm <= norm(y,1)
    invA_normest = norm(y,1);
    break;
  end

  % Update x to e_j and repeat
  x = zeros(n,1); x(j) = 1;

end
```

This method is not infallible, but it usually gives estimates that are the right order of magnitude. There are various alternatives, refinements, and extensions to Hager's method, but they generally have the same flavor of probing A^{-1} through repeated solves with A and A^T.

Scaling

Suppose we wish to solve $Ax = b$ where A is ill-conditioned. Sometimes, the ill-conditioning is artificial because we made a poor choice of units, and it

appears to be better conditioned if we write

$$D_1 A D_2 y = D_1 b,$$

where D_1 and D_2 are diagonal scaling matrices. If the original problem was poorly scaled, we will likely find $\kappa(D_1 A D_2) \ll \kappa(A)$, which may be great for Gaussian elimination. But by scaling the matrix, we are really changing the norms that we use to measure errors — and that may not be the right thing to do.

For physical problems, a good rule of thumb is to non-dimensionalize before computing. The non-dimensionalization will usually reveal a good scaling that (one hopes) simultaneously is appropriate for measuring errors and does not lead to artificially inflated condition numbers.