MA614. Computing Project 1 (due W. February 10 2016).

Goal: in this project, we will use MATLAB to investigate the stability and accuracy of three algorithms for solving linear systems of equations

$$Ax = b.$$

The algorithms are Gaussian elimination with partial pivoting, invert-and-multiply, and Cramer's rule.

Introduction

Stability How can you tell if an algorithm for solving the linear system $Ax = b$ is (backward) **stable** — that is if the computed solution $\hat{x}$ satisfies a slightly perturbed system

$$(A + E)\hat{x} = b \tag{1}$$

where[1]

$$\frac{\|E\|}{\|A\|} = \mathcal{O}(\mathbf{u}).$$

One way is to have a backward rounding-error analysis, as we will do for Gaussian elimination in class. But lacking that, how can we look at a computed solution and determine if it was computed stably?

Wonderful Residual If $\hat{x}$ satisfies (1), we can obtain a lower bound on $\|E\|$ by computing the residual

$$r = b - A\hat{x}.$$

Specifically, since $0 = b - (A + E)\hat{x} = r + E\hat{x}$, we have $\|r\| \leq \|E\hat{x}\| \leq \|E\|\|\hat{x}\|$. It follows that

$$\frac{\|E\|}{\|A\|} \geq \frac{\|r\|}{\|A\|\|\hat{x}\|}.$$

Thus if the relative residual norm

$$\frac{\|r\|}{\|A\|\|\hat{x}\|}$$

is large, we know that the solution was not computed stably.

On the other hand, if the relative residual is small, the result was computed stably. To see this, we must show that there is a small matrix E such that $(A + E)\hat{x} = b$. Let

$$E = \frac{r\hat{x}^T}{\|\hat{x}\|^2}.$$

Then

$$b - (A + E)\hat{x} = r - \frac{r\hat{x}^T}{\|\hat{x}\|^2}\hat{x} = 0.$$

So that $(A + E)\hat{x} = b$. But

$$\frac{\|E\|}{\|A\|} = \frac{\|r\hat{x}^T\|}{\|\hat{x}\|^2\|A\|}.$$

It is easy to see that $\|r\hat{x}^T\| = \|r\|\|\hat{x}\|$ (see Problem 2.5.7, in Homework 2). Hence

$$\frac{\|E\|}{\|A\|} = \frac{\|r\|}{\|A\|\|\hat{x}\|}.$$

What we have shown is that the relative residual norm

$$\frac{\|r\|}{\|A\|\|\hat{x}\|}$$

is a reliable indication of stability. A stable algorithm will yield a relative residual norm that is of the order of machine epsilon; an unstable algorithm will yield a larger value.

[1] the 2-norm is expensive to compute, it is used chiefly in mathematical investigation. However, it is ideal for our experiments; and since MATLAB has a function `norm` that computes the 2-norm, we will use it in this project. From now on, $\|\cdot\|$ will denote the vector and matrix 2-norm.

Accuracy From equation (1) of the backward error (stability) analysis of these three algorithms, the perturbation error bound as we obtained in class shows that the relative error between the exact solution x and the computed one $\hat{x}$ is bounded by the following:

$$\frac{\|x - \hat{x}\|}{\|x\|} \leq \kappa(A)\frac{\|E\|}{\|A\|}. \tag{2}$$

Matrices with known condition In order to investigate the effects of conditioning, we need to be able to generate nontrivial matrices of known condition number. Given an order n and a condition number κ we will take A in the form

$$A = UDV^T,$$

where U and V are random orthogonal matrices, and

$$D = \operatorname{diag}(1, \kappa^{-\frac{1}{n-1}}, \kappa^{-\frac{1}{n-2}}, \ldots, \kappa^{-1}).$$

The fact that the condition number of A is κ follows directly from the properties of the 2-norm:

$$\kappa(A) = \|A\|\,\|A^{-1}\| = \|UDV^T\|\,\|VD^{-1}U^T\| = \|D\|\,\|D^{-1}\| = \kappa.$$

Assignment

Test matrix generation The first part of the project is to write a function

```
function a = condmat(n, kappa)
```

to generate a matrix of order n with condition number κ. To obtain a random orthogonal matrix, use the MATLAB function `randn` to generate a random, normally distributed matrix. Then use the function `qr` to factor the random matrix into the product QR of an orthogonal matrix and an upper triangular matrix, and take Q for the random orthogonal matrix.

Invert-and-multiply

The second part of the project is to compare the stability and accuracy of Gaussian elimination with the invert-and-multiply algorithm for solving $Ax = b$. Write a function

```
function invmult(n,kap)
```

where `n` is the order of the matrix A and `kap` is a vector of condition numbers. For each component `kap(i)`, the function should do the following

1. Generate a random $n \times n$ matrix A of condition `kap(i)`.
2. Generate a (normally distributed) random n-vector x.
3. Calculate $b = Ax$.
4. Calculate the solution of the system $Ax = b$ by Gaussian elimination.
5. Calculate the solution of the system $Ax = b$ by inverting A and multiplying b by the inverse.
6. Print the following quantities:

   `i, kap(i), geerr, geres, gebd, imerr, imres, imbd`

   where

   `geerr` is the relative error in the solution by Gaussian elimination.

   `geres` is the relative residual norm for Gaussian elimination.

   `gebd` is the error bound for Gaussian elimination.

   `imerr` is the relative error in the invert-and-multiply solution.

   `imres` is the relative residual norm for invert-and-multiply.

   `imbd` is the error bound for invert-and-multiply.

Note: The MATLAB left divide operate '\' is implemented by Gaussian elimination. To invert a matrix, use the function `inv`. The norm is computed by the function `norm`.

Cramer's rule

The third part is to compare the stability and accuracy of Gaussian elimination with Cramer's rule for solving the 2×2 system $Ax = b$.[2] For such a system, Cramer's rule can be written in the form

$$x_1 = (b_1 a_{22} - b_2 a_{12})/d,$$
$$x_2 = (b_2 a_{11} - b_1 a_{21})/d,$$

where

$$d = a_{11}a_{22} - a_{21}a_{12}.$$

Write a function

```
function cramer(kap)
```

where `kap` is a vector of condition numbers. For each component `kap(i)`, the function should do the following

7. Generate a random 2×2 matrix A of condition `kap(i)`.
8. Generate a (normally distributed) random 2-vector x.
9. Calculate $b = Ax$.
10. Calculate the solution of the system $Ax = b$ by Gaussian elimination.
11. Calculate the solution of the system $Ax = b$ by Cramer's rule.
12. Print the following quantities:

    `i, kap(i), geerr, geres, gebd, cerr, cres, cbd`

    where

    `geerr` is the relative error in the solution by Gaussian elimination.

    `geres` is the relative residual norm for Gaussian elimination.

    `gebd` is the error bound for Gaussian elimination.

    `cerr` is the relative error in the solution by Cramer's rule.

    `cres` is the relative residual norm for Cramer's rule.

    `cbd` is the error bound for Cramer's rule.

Submission

Run your programs for

$$\texttt{kap} = (1, 10^4, 10^8, 10^{12}, 10^{16})$$

using the MATLAB command `diary` to accumulate your results in a file. Edit the diary file and put brief statements and comments in your own words of what the results mean (Just like writing a technical report about your finding). You must include **Summary table** which summarises, for each of the 3 methods, your conclusions using the following crieteria: relative error, backward stabilty, theoretical relative error.

Submit a hard copy and mail the file (in text format) along with the functions to mark.pekker@uah.edu.

[2] just for simplicity, of course, we can use the Cramer's rule for any dimension. Then it will be numerically very unstable to compute the determinant.