

## MCS 471 Project Three: Solving Linear Systems

The goal is to investigate the numerical stability of algorithms to solve linear systems and to study the numerical conditioning of this problem. In our experiments we will use MATLAB or Octave. MATLAB is available in all computer labs on campus, but is commercial and expensive. Octave is similar to MATLAB (for our work, Octave runs just as fine), and is free to download from <http://www.octave.org/>.

### 0. Solving Linear Systems in MATLAB or Octave

The general command to solve linear systems is done by the “backslash” operator: for a matrix  $A$  and right-hand-side vector  $b$ , we find the solution  $x$  as  $x = A \backslash b$ . The residual vector is then  $r = b - A*x$ .

The LU factorization is provided in the command `lu`. For a matrix  $A$ , `[l,u,p] = lu(A)` returns in  $l$ ,  $u$ , and  $p$  the lower, upper, and permutation matrix used in the LU factorization with partial pivoting. To solve  $A x = b$ , the `[l,u,p] = lu(A)` is followed by the sequence `pb = p*b; y = l \ pb; x = u \ y;`

The command `cond` provides an estimate for the condition number of a matrix.

### 1. Pivoting to Improve the Numerical Stability

To examine how bad things may get without pivoting, we use the following simple routines (available for download from the class web site):

```
function [l,u] = simple_lu ( a )
%
% [l,u] = simple_lu(a) applies the LU factorization
%   to the matrix in a, without pivoting.
%
% example: a = randn(3,3)
%           [l,u] = simple_lu(a)
%           l*u - a
and
function x = substitute ( l,u,b )
%
% x = substitute(l,u,b) solves l*y = b and u*x = y,
%   where l is lower and u upper triangular.
%
% example: a = randn(3,3); b = randn(3,1);
%           [l,u] = simple_lu(a);
%           x = substitute(l,u,b);
%           b - a*x
```

If we apply these two functions on random linear systems, then everything looks fine, but it is not so hard to come up with families of systems for which the errors get very large.

**Assignment One.** Generate matrices of dimension  $n$ ,  $n$  ranging from 3 to 10, with the sequence `r = randn(n,n); d = r - triu(r,1) - tril(r,-1); a = r - d; and a = a + d*10^(-k);` where  $k$  ranges from 1 to 10. Then with `s = ones(n,1); b = a*s;` we generate a right-hand side vector of a system whose exact solution is  $s$ . For  $n$  ranging from 3 to 10, and  $k$  from 1 to 10, solve the generated linear systems using `simple_lu` and `substitute`. Make a table of errors, as the difference between the computed and the exact solution. How do the errors change as a function of  $n$  and  $k$ ?

**Assignment Two.** Repeat the experiment of assignment one (generate a similar sequence of linear systems), but now solve the systems with the built in `lu` command and the backslash operator to solve the triangular systems. Also here make a table of errors for  $n$  ranging from 3 to 10, and  $k$  from 1 to 10. What is the difference with the errors of assignment one? Explain this difference observing the permutation matrices.

## 2. Condition Numbers to Estimate the Errors

In class we derived the following:

$$\frac{\|\mathbf{x} - \bar{\mathbf{x}}\|}{\|\mathbf{x}\|} \leq \text{cond}(A) \frac{\|A - \bar{A}\|}{\|A\|}, \quad (1)$$

where  $\mathbf{x}$  is the exact solution to  $A\mathbf{x} = \mathbf{b}$ , and  $\bar{\mathbf{x}}$  is the computed solution, satisfying  $\bar{A}\bar{\mathbf{x}} = \mathbf{b}$ . So from the condition number  $\text{cond}(A)$  and the relative error on the input the right-hand side of (1) gives us an upper bound on the relative error on the solution. In the following assignment we will experimentally study how good the bound in (1) can predict the relative error on the solution.

**Assignment Three.** For  $n = 10$ , the sequence `kv = 1:n; kv(n) = 10^k; dv = diag(kv); q = orth(randn(n)); a = q'*dv*q;` for  $k$  ranging from 3 to 12, generates matrices with controlled condition number. Like in previous assignments, we generate the right-hand side vector so that the exact solution equals `s = ones(n,1); b = a*s;`. Make a table with three columns. The first column is the relative error on the computed solution, for  $k$  ranging from 3 to 12. The second column is the condition number of the generated matrix. The third column is the right-hand side of (1), where the relative error on  $A$  is `eps`, the machine precision.

Compare the first and third column with each other. How good does the condition number predict the relative error?

## 3. Deadline is Friday 24 October, at 1PM

Bring your project solution to class. It should contain the following:

1. Tables with numerical data, numbers formatted in scientific format.
2. Answers to the questions in the assignments. Please write complete grammatically correct sentences and avoid spelling mistakes.
3. Sequences of commands used to generate the data for the experiments. It is good practice to store these commands in little `.m` files, e.g., in `assignment_one.m`, `assignment_two.m`, etc.
4. Output of your sessions with MATLAB or Octave as *an appendix*. Typing `diary` followed by the name of a file in a session creates a new file with the given name which will contain everything you see on the screen during the session. You may edit out mistakes from the output of `diary` or `truncate`. This output is only as a backup, used for partial credit (if needed).

See <http://www.math.uic.edu/~jan/mcs471/index.html> for the hypertext version of this project and for the functions `simple_lu.m` and `substitute.m`.

If you have questions, comments, or difficulties, feel free to come to my office for help.