Lab Session 5

MA-423 : Matrix Computations Lab 2032 R. Alam

Errors, pivot growth and backward stability

1. Wilkinson's matrix is defined as follows: 1 on the diagonal, -1 everywhere below the main diagonal, 1 in the last column, and 0 everywhere else. Write a MATLAB function `W = Wilkinson(n)` that generates Wilkinson's matrix W of size n using MATLAB functions `eye, tril` and `ones`.

   (a) For $n = 32$, pick a random x and then compute $b := W * x$. Solve $Ax = b$ using MATLAB backslash command and compute the error $\|x - \hat{x}\|_\infty / \|x\|_\infty$ (type `help norm` for more info about computing norm). Does the size of the error confirm that GEPP is unstable for this system? Also compute $\mathrm{cond}(A)$. Can the poor answer be attributed to ill-conditioning of the matrix W? Repeat the test for $n = 64$.

   (b) Repeat the experiment in part (a) using QR decomposition. It is easy in matlab. The command `[Q,R] =qr(A)` gives unitary Q and upper triangular R such that $A = QR$. Solve $Wx = b$ using QR decomposition and compare the results with those in part(a). Which of the two methods appear to give a better answer?

   (c) Pivot growth of Gaussian elimination with partial pivoting (GEPP) is given by $PG(A) = \max_{ij} |U(i,j)| / \max_{ij} |A(i,j)|$, which influences the accuracy of computed solution. Use MATLAB function `[L, U, p] = lu(A)` for computing LU decomposition of a nonsingular matrix A and compute the pivot growth `rho` $= PG(A)$. Use commands `max` and `abs`.

   It is well known that the pivot growth factor for GEPP satisfies $PG(A) \leq 2^{n-1}$ which is attained by the Wilkinson matrix. Verify this graphically by doing the following:

   First plot the graph of 2^{n-1} in $\log 10$ scale for $n = 10 : .5 : 505$ by setting `X = 2.^(n-1)` and then typing `semilogy(n,X,'r')`. Hold this plot by typing `hold on` and type the following sequence of commands (which assumes that the Wilkinson matrix of size n is generated by the function `W = Wilkinson(n)`).

   ```
   n = 10:20:500;  m = length(n); G = zeros(m,1);
   for i = 1:m
   W = Wilkinson(n(i)); [L,U,p] = lu(W);
   G(i) = max(max(abs(U)))/(max(max(abs(W))));
   end
   semilogy(n,G,'b*')
   ```

   The second plot should come in the form of blue dots that fall on the red curve produced by the earlier plot.

   However, statistics suggest that for most practical examples, $PG(A) \leq n^{2/3}$ for GEPP. Verify this graphically by generating random matrices instead of Wilkinson matrices in the sequence of commands given above.

(d) There is no strong correlation between pivot growth and the ill-conditioning of a matrix. This is illustrated by a Golub matrix. A Golub matrix A of size n is an ill-conditioned integer matrix whose LU factorization without pivoting fails to reveal that A is ill-conditioned. The matrix A is given by $A := LU$, where L unit lower triangular with random integer entries and U is unit upper triangular with random integer entries. The function `golub` given below generates a Golub matrix of size n :

```
function A = golub(n)
s = 10;
L = tril(round(s*randn(n)),-1)+eye(n);
U = triu(round(s*randn(n)),1)+eye(n);
A = L*U;
```

Compute LU factorization of A using your function `[L, U] = GENP(A)`. Also, compute the pivot growth $PG(A)$ and the condition number $\text{cond}(A) = \|A\|_2\|A^{-1}\|_2$ using MATLAB command `cond(A)`. If $\text{cond}(A)$ is large then the system $Ax = b$ is ill-conditioned and in such a case A is called ill-conditioned. Does $PG(A)$ reflect the ill-conditioning of A?

2. **Assignment:** Your task is to generate test matrices having pre-specified condition numbers. Write a matlab function to generate an n-by-n matrix with 2-norm condition number `ka`:

```
function A = matgen(n, ka)
```

To obtain such a matrix proceed as follows. Choose random orthogonal matrices U and V from QR factorization of random matrices: `[U, R] = qr(rand(n))` and `[V,R] = qr(rand(n))`. Generate a diagonal matrix $D = \text{diag}(d_i)$ with $d_i := \texttt{ka}^{-(i-1)/(n-1)}$ and then set $A := UDV^*$. Then the 2-norm condition number of A is equal to `ka`.

Check that the matrix A generated by your function `matgen` has the condition number `ka` (call MATLAB function `cond(A)` to compute the condition number of A).

Test the stability of algorithms specified below and analyze the accuracy of solutions returned by them. For this purpose, proceed as follows:

1. For each `ka` $= 10^4, 10^8, 10^{12}, 10^{16}$, use your `matgen` function to generate a random matrix A of size $n = 100$ with condition number `ka`.
2. Compute `b = A*ones(n,1)`. Then `x =ones(n,1)` is the solution of $Ax = b$.

Now consider the following algorithms:

1. `ALG1`: Solve $Ax = b$ using backslash command (`GEPP`)
2. `ALG2`: Solve $Ax = b$ using `GECP`
3. `ALG3`: Solve $Ax = b$ using `GENP`
4. `ALG4`: Solve $Ax = b$ by $A^{-1}b$ (MATLAB command `inv` computes inverse of a matrix).

For each of the above algorithms, prepare a table giving values of the following quantities:

`ka, bkerr, err, errbd`

where

- `bkerr` - is the backward error ($= \|Ax - b\|_2 / \|A\|_2 \|x\|_2$)
- `err` - is the relative error in the solution
- `errbd` - is the relative error bound of the solution. [If $r = b - Ax$ then $(A+E)x = b$ where $E = rx^* / \|x\|_2^2$. Now for error bound, invoke perturbation theory.]

Based on your results comment on the backward stability of `ALG1`, `ALG2`, `ALG3`, `ALG4`, and the accuracy of the solutions returned by these methods. Use the MATLAB commands `tic, toc` (type `help tic` for more information) to determine the time taken by each of these methods to solve the problems. **10 marks**

************End************