Stanford University, Management Science and Engineering (and ICME)

CME 338 Large-Scale Numerical Optimization

Instructor: Michael Saunders Spring 2019

Homework 3, Due Wednesday May 9

`http://stanford.edu/class/cme338/homework.html`

1. For the iterative Golub-Kahan orthogonal bidiagonalization with $A \in \mathbb{R}^{m\times n}$ and $\beta_1 u_1 = b \in \mathbb{R}^m$, we have

$$\begin{aligned} AV_k &= U_{k+1}B_k = U_k L_k + \beta_{k+1} u_{k+1} e_k^T, \\ A^T U_{k+1} &= V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T = V_{k+1} L_{k+1}^T, \end{aligned}$$

   where (with exact arithmetic) U_k and V_k are orthonormal and either $\beta_{\ell+1} = 0$ or $\alpha_{\ell+1} = 0$ for some $k = \ell \le \min(m, n)$. Prove the following:

   (a) L_k and B_k have full column rank k for all $k \le \ell$.

   (b) If $\beta_{\ell+1} = 0$, we have $AV_\ell = U_\ell L_\ell$ with L_ℓ nonsingular and $b \in \text{range}(A)$.

   (c) If $\beta_{\ell+1} > 0$ but $\alpha_{\ell+1} = 0$, then $b \notin \text{range}(A)$.

2. From `http://stanford.edu/class/msande318/matlab/`, download the files `bcsstm34.mat` and `CGtest9.m`.[1] Script `CGtest9` shows how to load an $n \times n$ matrix A into MATLAB ($n = 588$). The eigenvalues of A range from $\lambda_1 = -2.6830$ to $\lambda_n = 6.8331$, and $\text{cond}(A) \approx 1.1 \times 10^7$.

   The matrix $B = A + \sigma_1 I$ with $\sigma_1 = 2.7$ is positive definite, and $\text{cond}(B) \approx 2400$. Define column vector $x = [1/j]$ $(j = 1 : n)$ and $b = Bx$.

   (a) Solve $Bx = b$ using some of the iterative solvers provided in MATLAB: pcg, minres, symmlq, and lsqr. Use the parameters `tol = 1e-9` and `maxit = 1000`. For each method, plot the residuals $\|r_k\|$ for each iteration k. Script `CGtest9` already does this in producing figure(1). Add suitable {xlabel, ylabel, legend} and give a table of results.

   (b) The residuals for each solver are somewhat different. Did all solvers terminate at much the same point? Should we expect them to be more different?

   (c) `type pcg` allows you to see MATLAB's implementation of CG. For each solver, find which stopping rule was used for the results shown in figure(1).

[1] These questions use Boeing/bcsstm34, a symmetric indefinite sparse matrix A of size $n = 588$ from the SuiteSparse Matrix Collection maintained by Tim Davis: `http://faculty.cse.tamu.edu/davis/welcome.html`.

3. The matrix $C = A + \sigma_2 I$ with $\sigma_2 = 0.5$ is indefinite, and $\text{cond}(C) \approx 83,000$. With the same $x = [1/j]$, define $b = Cx$.

   (a) Solve $Cx = b$ using pcg, symmlq, minres. Script `CGtest9` already does this in producing figure(2). Again, add suitable {xlabel, ylabel, legend} and give a table of results.

   (b) Note that pcg on $Cx = b$ terminates early. What happened?

   (c) When C is reasonably well-conditioned, lsqr performs much the same as pcg and minres on $C^2x = Cb$ in terms of iterations. But what is the plot for lsqr showing?

4. The script plots the eigenvalues $\lambda(C)$ in figure(3). Does there seem to be any clustering of the eigenvalues? A better picture is given in figure(4). Briefly describe what the script is showing in figure(4). Does it explain why the symmetric solvers needed significantly fewer than n iterations to solve $Cx = b$?

   To check your intuition, find the eigensystem of C from

   ```
   [V,D] = eig(full(C));
   ```

   and find c such $b = Vc$. (What to we know about V? How does this help us find c?) Then do

   ```
   figure(5); plot(sort(c,1,'descend'))
   ```

   and describe what you see.

```
% CGtest9.m is a script for comparing {cg, symmlq, minres}
% on sparse matrix Boeing/bcsstm34, n=588, nnz=24270).
% See http://faculty.cse.tamu.edu/davis/welcome.html (Tim Davis).
% The matrix A is from a structural problem.
% It is symmetric indefinite with lambda_min = -2.6830.
%
% 09 Apr 2017: Problem Boeing/bcsstm34 used for Homework 3.
% 22 Apr 2018: Removed pcg and minres on C^2x = Cb.
% 30 Apr 2019: For lsqr, pring resvecS, not lsvec.
%-------------------------------------------------------------

load bcsstm34.mat;   % lambda(min) = -2.6830, lambda(max) = 6.8331
A     = Problem.A;   % Save original matrix A
[n,n] = size(A);
x     = 1./(1:n)';


%-------------------------------------------------------------
sigma1 = 2.7;
B      = A + sigma1*speye(n);   condB = condest(B);
b      = B*x;
tol    = 1e-9;   % Not highly accurate
maxit  = 1000;

[xC,flagC,relresC,iterC,resvecC]       = pcg   (B,b,tol,maxit);
[xL,flagL,relresL,iterL,resvecL]       = symmlq(B,b,tol,maxit);
[xM,flagM,relresM,iterM,resvecM]       = minres(B,b,tol,maxit);
[xS,flagS,relresS,iterS,resvecS,lsvec] = lsqr  (B,b,tol,maxit);

errC  = norm(xC-x,inf);   % The inf-norm is best for large vectors
errL  = norm(xL-x,inf);
errM  = norm(xM-x,inf);
errS  = norm(xS-x,inf);

fprintf('\nPOS-DEFINITE B = A + sigma1*I,')
fprintf('  sigma1 =%5.2f,  condest(B) = %8.1e\n\n', sigma1, condB)
fprintf('                flag  iter   relres    error\n')
fprintf(' CG       Bx = b%4g %5g %8.1e %8.1e  b\n', flagC,iterC,relresC,errC)
fprintf(' SYMMLQ   Bx = b%4g %5g %8.1e %8.1e  r\n', flagL,iterL,relresL,errL)
fprintf(' MINRES   Bx = b%4g %5g %8.1e %8.1e  g\n', flagM,iterM,relresM,errM)
fprintf(' LSQR     Bx = b%4g %5g %8.1e %8.1e  m\n', flagS,iterS,relresS,errS)

figure(1)
hold off;  plot(log10(resvecL),'r-')
hold  on;  plot(log10(resvecC),'b-')
hold  on;  plot(log10(resvecM),'g-')
hold  on;  plot(log10(resvecS),'m-')


%-------------------------------------------------------------
sigma2 = 0.5;
C      = A + sigma2*speye(n);   condC  = condest(C);
b      = C*x;    Cfun  = @(x) C*x;       % Treat  C  as a function
%b2    = C*b;    Cfun2 = @(x) C*(C*x);   % Treat C*C as a function
```

```
[xC,flagC,relresC,iterC,resvecC]       = pcg   (Cfun ,b ,tol,maxit);
[xL,flagL,relresL,iterL,resvecL]       = symmlq(Cfun ,b, tol,maxit);
[xM,flagM,relresM,iterM,resvecM]       = minres(Cfun ,b ,tol,maxit);

errC  = norm(xC-x,inf);
errL  = norm(xL-x,inf);
errM  = norm(xM-x,inf);

fprintf('\n  INDEFINITE C = A + sigma2*I,')
fprintf('  sigma2 =%5.2f,  condest(C) = %8.1e\n\n', sigma2, condC)
fprintf('                flag  iter   relres    error\n')
fprintf(' CG       Cx = b%4g %5g %8.1e %8.1e  b\n', flagC,iterC,relresC,errC)
fprintf(' SYMMLQ   Cx = b%4g %5g %8.1e %8.1e  r\n', flagL,iterL,relresL,errL)
fprintf(' MINRES   Cx = b%4g %5g %8.1e %8.1e  g\n', flagM,iterM,relresM,errM)

figure(2)
hold off;  plot(log10(resvecC),'b-')
hold  on;  plot(log10(resvecL),'r-')
hold  on;  plot(log10(resvecM),'g-')

%-------------------------------------------------------------
% Plot the eigenvalues of C.
%-------------------------------------------------------------
lambda = eig(full(C));
figure(3)
hold off;    plot(lambda,'b.')
xlabel('Eigenvalue number');   ylabel('\lambda(C)');
title('Eigenvalues of C');

% Show if the eigenvalues are clustered.
figure(4)
hold off;    plot(lambda,250*ones(n,1),'b.')
hold on

y1    = -3;      yn    =  7;
step  =  0.25;   nbar  = (yn - y1)/step + 1;
y     = zeros(nbar,1);
nlam  = zeros(nbar,1);

for i = 1:nbar
  y2      = y1 + step;
  nlam(i) = length( find(lambda>y1 & lambda<=y2) );
  y(i)    = y1 + 0.5*step;
  y1      = y2;
end

bar( y, nlam )
xlabel('\lambda(C)');   ylabel('No. of \lambda(C)');
title('Distribution of eigenvalues of C');
```