Advanced Numerical Analysis Homework 3

Michael Nelson

Throughout this homework, $\|\cdot\|$ denotes the ℓ_2-norm. We also let $\langle\cdot,\cdot\rangle$ denote the standard Euclidean inner-product on $\mathbb{R}^m$ (thus $\langle v, w\rangle = v^\top w$ for all $v, w \in \mathbb{R}^m$).

1 Problem 1

Exercise 1.

1. Determine the eigenvalues, determinant, and singular values of a Householder reflection $H_v = 1 - 2\frac{vv^\top}{v^\top v}$. For the eigenvalues, give a geometric argument as well as an algebraic proof.
2. Consider the Givens rotation
$$G_\theta = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$
Give a geometric interpretation of the action of G_θ on a vector in $\mathbb{R}^2$. Do the same analysis as part 1 for G, but no geometric interpretation is needed for the eigenvalues.

Solution 1. 1. Let Γ be the hyperplane which is orthogonal to v, i.e.

$$\Gamma = \{w \in \mathbb{R}^n \mid \langle w, v\rangle = 0\}.$$

Note that $\dim\Gamma = n-1$; let $w_1, \dots, w_{n-1}$ be a basis for Γ. Then $\boldsymbol{e} := w_1, \dots, w_{n-1}, v$ is an eigenbasis for H_v. Indeed, clearly $\boldsymbol{e}$ is linearly independent and spans $\mathbb{R}^n$. Furthermore, we have $H_v(w_i) = w_i$ for all $1 \le i \le n-1$ and similarly we have $H_v(v) = -v$. Thus the eigenvalues for H_v are ± 1, and $\boldsymbol{e}$ is a corresponding eigenbasis. The matrix representation of H_v with respect to $\boldsymbol{e}$ is the diagonal matrix:

$$[H_v] := \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & 0 \\ 0 & 0 & \cdots & 0 & -1 \end{pmatrix}.$$

In particular we have $\det H_v = -1$. Finally, the singular values σ_i of H_v are just the absolute values of the eigenvalues since $[H_v]$ is a diagonal matrix, so $\sigma_i = 1$ for all $1 \le i \le n$.

2. The action of G_θ on a vector v is a counter-clockwise rotation by the angle θ. The matrix representation of G_θ with respect to the standard basis is
$$G_\theta = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$
Thus $\det G_\theta = \cos^2\theta + \sin^2\theta = 1$ and $\operatorname{tr} G_\theta = 2\cos\theta$. Thus the eigenvalues λ of G_θ are solutions to the equation:

$$\lambda^2 - 2\cos(\theta)\lambda + 1 = 0.$$

By the quadratic formula, the solutions to this quadratic equation are given by $\lambda = \cos\theta \pm i\sin\theta = e^{\pm i\theta}$. The singular values σ_i of G_θ are the positive square roots of the eigenvalues of

$$G_\theta^\top G_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

clearly $\sigma_1 = 1 = \sigma_2$.

2 Problem 2

Exercise 2. Implement QR factorizations in MATLAB based on:

1. classical Gram-Schmidt (CGS)
2. modified Gram-Schmidt (MGS)
3. MGS with double orthogonalization, and
4. Householder reflectors (for Householder $H = 1 - \frac{2vv^\top}{v^\top v}$, let $v = x + \operatorname{sign}(x_1)\|x\|_2 e_1$ with $\operatorname{sign}(z) = 1$ for $z = 0$ and $e^{i\theta}$ for $z = \rho e^{i\theta} \neq 0$.

Then we construct three matrices as follows.

```
A1 = randn(2^20,15); % (large but well−conditioned)
u = (−1:2/40:1)';
A2 = u.^(0:23); % (partial Vandermonde)
A3 = u.^(0:40); % (full Vandermonde)
```

For each matrix, run the algorithms, then compute

$$\frac{\|A - \widehat{Q}\widehat{R}\|_F}{\|A\|_F} \quad \text{and} \quad \|\widehat{Q}^\top \widehat{Q} - 1\|.$$

Draw conclusions about the backward stability of these algorithms, and the orthogonality of the computed Q factors, probably related to the condition numbers of the matrices.

Solution 2. We implemented each of the four QR factorizations in MATLAB using the code found the in the appendix. In matlab, we found that We collect our results for $\|A - \widehat{Q}\widehat{R}\|_F / \|A\|_F$ in the table below:

$\|A - \widehat{Q}\widehat{R}\|_F / \|A\|_F$	CGS	MGS	MGSD	HOUSE
A_1	$1.4169e-16$	1.4173e $-$ 16	$1.4411e-16$	$1.4252e-15$
A_2	$1.1050e-16$	$9.8748e-17$	$1.2877e-16$	$3.6771e-16$
A_3	$1.4014e-16$	$1.2570e-16$	$1.7105e-16$	$4.1221e-16$

Similarly, we collect our results for $\|\widehat{Q}^\top \widehat{Q} - 1\|$ in the table below:

$\|\widehat{Q}^\top \widehat{Q} - 1\|$	CGS	MGS	MGSD	HOUSE
A_1	$2.7756e-15$	$2.7757e-15$	$3.4417e-15$	$2.8867e-16$
A_2	1.9466	$1.3845e-08$	$7.5919e-16$	$8.4959e-16$
A_3	10.2844	0.9760	$1.2392e-15$	$1.2460e-15$

All of the algorithms produce small relative forward errors (i.e the relative frobenius-norm difference of A and $\widehat{Q}\widehat{R}$ is very small in each algorithm). However there are notable differences in these algorithms when it comes to the orthogonality of the computed Q factors. Indeed, the Q factors from MGSD and HOUSE were always extremely close to being orthogonal for each of the matrices, however both CGS and MGS produced Q-factors of A_3 which weren't close to being orthogonal (MGS did better for A_2 but was still off by an order of 10^8). Having said that, both MGSD and HOUSE were still able to produce Q-factors of A_1 which were close to being orthogonal (only off by a factor of 10). This is related to the fact that A_1 is well-conditioned (the condition number of A is 1.0064).

3 Problem 3

Exercise 3. Evaluate the arithmetic work needed to retrieve the reduced factor $Q_L \in \mathbb{R}^{m\times n}$ from the Householder and Givens reduction of A to R, respectively (second phase of QR). Compare the cost with that for the first phase.

Solution 3. It suffices the Householder algorithm since the analysis of the Givens algorithm is the same. In order to retrieve Q_L, one calculates $Qe_1, Qe_2, \dots, Qe_n$. The calculation of Qe_i is of the order $O(mn)$, so the calculation of Q_L is of the order $O(mn^2)$, which is precisely the same order as the cost of the first phase. Thus retrieving Q_L amounts to doubling our cost.

4 Problem 4

Exercise 4. Implement the algorithm for solving linear system $Ax = b$ or linear least squares problem $\min \|b - Ax\|$ based on Householder QR. Make sure that the reduced Q factor is NOT formed explicitly to save the cost of the second phase. Then solve the linear least squares problem $\min \|b - Ax\|$ where $A = A_2$, and the linear system $Ax = b$, where $A = A_3$ in [Q2], and $b = [1, -1, 1, -1, \dots]^\top$. Report your

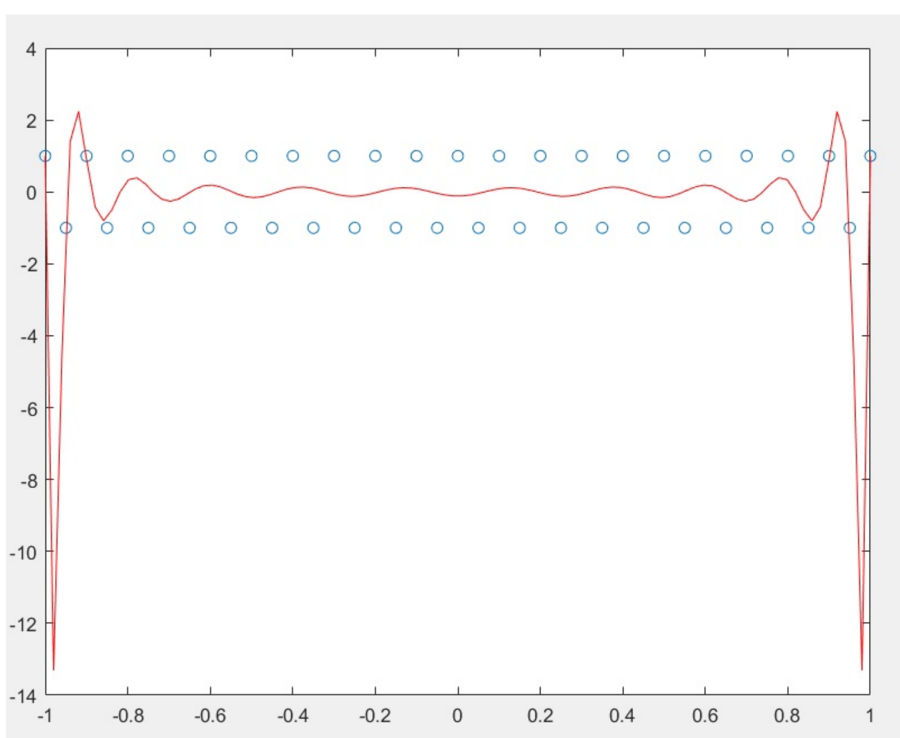
$$\frac{\|b - A\widehat{x}\|}{\|A\|\|\widehat{x}\|}$$

for both solves, and compare with this quantity associated with the solutions obtained by MATLAB's backslash.

Solution 4. The code we used for the Householder QR algorithm is in the Appendix. We report our results in the table below:

$\|b - A\widehat{x}\| / \|A\|\|\widehat{x}\|$	BACKSLASH	HOUSE
A_2	$3.1026e-08$	$3.1026e-08$
A_3	$5.4432e-17$	$3.3642e-17$

Both algorithms gives us the same least squares solution in the case of A_2. Below we plot the polynomial which corresponds to this solution (i.e. the degree 23 polynomial interpolant to the 41 data points b):



On the other hand, the quantity $\|b - A\widehat{x}\| / (\|A\|\|\widehat{x}\|)$ for backslash is almost double that of the corresponding quantity for the Householder QR method when it comes to A_3.

Appendix

Classical Gram-Schmidt

```
function [Q,R] = gs(A)

[m,n] = size(A); Q = zeros(m,n); V = zeros(m,n); R = zeros(m,n);

for j = 1:n
   V(:,j) = A(:,j);
    for i = 1:j-1
      R(i,j) = Q(:,i)'*A(:,j);
      V(:,j) = V(:,j) - R(i,j)*Q(:,i);
   end;
   R(j,j) = norm(V(:,j)) ;
   Q(:,j) = V(:,j) / R(j,j) ;
   end;
end;
```

Modified Gram-Schmidt

```
function [Q,R] = mgs(A)

[m,n] = size(A); Q = zeros(m,n); V = A; R = zeros(n,n);

for i = 1:n
   R(i,i) = norm(V(:,i));
   Q(:,i) = V(:,i) / R(i,i);
    for j = (i+1):n
      R(i,j) = Q(:,i)'*V(:,j);
      V(:,j) = V(:,j) - R(i,j)*Q(:,i);
   end;
end;
```

Double Modified Gram-Schmidt

```
function [Q,R] = mgsd(A)

[Q1,R1] = mgs(A);
[Q,R2] = mgs(Q1);
R = R2*R1;
```

Householder Factorization

```
function [V,R] = house(A)

[m,n] = size(A); V = zeros(m,n);

for k = 1:n
   x = A(k:m,k);
   V(k:m,k) = sign(x(1))*norm(x)*eye(m-k+1,1)+x;
   V(k:m,k) = V(k:m,k)/norm(V(k:m,k));
   A(k:m,k:n) = A(k:m,k:n) - 2*V(k:m,k)*(V(k:m,k)'*A(k:m,k:n));
end
```

```
R = A(1:n,:);
```

Evaluate Qb or Q^*b

```
function [b] = houseev(V,b,transpose)

[m,n] = size(V);

if transpose
   for k = 1:n
     b(k:m) = b(k:m) - 2*V(k:m,k)*(V(k:m,k)'*b(k:m));
   end;
else
   for k = n:-1:1
      b(k:m) = b(k:m) - 2*V(k:m,k)*(V(k:m,k)'*b(k:m));
   end;
end;
```

Form $\widehat{Q}$ From House

```
function Q = houseformQ(V)

[m,n] = size(V); Q = zeros(m,n);

for j = 1:n
   x = zeros(m,1);
   x(j,1) = 1;
   Q(:,j) = houseev(V,x,0);
end;
```

Least Squares via Householder QR

```
function x = leastsquareshouseQR(A,b)

[V,R] = house(A);
y  = houseev(V,b,1);
y  = y(1:n);
x = R\y;
```