CS 421: Fall 1997

Solutions to Assignment 3

1. Gaussian elimination without pivoting for pentadiagonal matrices:

```
function [l2,l1,u0,u1,u2]=pentaLU(e2,e1,d,f1,f2)
% LU-factorization of a pentadiagonal matrix A.
% e2, e1, d, f1, and f2 are the subdiagonal, second subdiagonal,
% diagonal, superdiagonal and second superdiagonal of A respecively.
% l1 and l2 are the subdiagonal and second subdiagonal of L.
% u0, u1, and u2 are the diagonal, superdiagonal and second
% superdiagonal of U.

n=length(d);
l2=e2; l1=e1; u0=d; u1=f1; u2=f2;
for k=1:n-2
    l1(k)=l1(k)/u0(k);
    l2(k)=l2(k)/u0(k);
    u0(k+1)=u0(k+1)-l1(k)*u1(k);
    u1(k+1)=u1(k+1)-l1(k)*u2(k);
    l1(k+1)=l1(k+1)-l2(k)*u1(k);
    u0(k+2)=u0(k+2)-l2(k)*u2(k);
end
l1(n-1)=l1(n-1)/u0(n-1);
u0(n)=u0(n)-l1(n-1)*u1(n-1);
```

A solver that uses the factorization (in vector form) produced by pentaLU:

```
function b=pentaSolve(l2,l1,u0,u1,u2,b)
% Solves LUx=b where LU is the factorization of a pentadiagonal
% matrix computed using pentaLU.
% l1 and l2 are the subdiagonal and second subdiagonal of L.
% u0, u1, and u2 are the diagonal, superdiagonal and second
% superdiagonal of U.

n=length(b);
% Solve Ly=b (and overwrite b with solution)
b(2)=b(2)-l1(1)*b(1);
for k=3:n
    b(k)=b(k)-l1(k-1)*b(k-1)-l2(k-2)*b(k-2);
end
% Solve Ux=y (again overwrite b)
b(n)=b(n)/u0(n);
b(n-1)=(b(n-1)-u1(n-1)*b(n))/u0(n-1);
for k=n-2:-1:1
    b(k)=(b(k)-u1(k)*b(k+1)-u2(k)*b(k+2))/u0(k);
end
```

```
>> % Test of functions for pentadiagonal matrices
>> n=7;
>> % Generate a random pentadiagonal matrix A
>> e2=randn(n-2,1);
>> e1=randn(n-1,1);
>> d=randn(n,1);
>> f1=randn(n-1,1);
>> f2=randn(n-2,1);
>> A=diag(e2,-2)+diag(e1,-1)+diag(d)+diag(f1,1)+diag(f2,2)

A =

   -0.9219    0.5913    0.0000         0         0         0         0
    0.2573   -2.1707   -0.6436   -0.3179         0         0         0
    1.1908   -1.0565   -0.0592    0.3803    1.0950         0         0
         0   -1.2025    1.4151   -1.0106   -1.0091   -1.8740         0
         0         0   -0.0198   -0.8051    0.6145   -0.0195    0.4282
         0         0         0   -0.1567    0.5287    0.5077   -0.0482
         0         0         0         0   -1.6041    0.2193    1.6924

>> % LU-factorization
>> [l2,l1,u0,u1,u2]=pentaLU(e2,e1,d,f1,f2);
>> L=diag(l2,-2)+diag(l1,-1)+eye(n,n)

L =

    1.0000         0         0         0         0         0         0
   -0.2791    1.0000         0         0         0         0         0
   -1.2917    0.1459    1.0000         0         0         0         0
         0    0.5995   51.7649    1.0000         0         0         0
         0         0   -0.5688    0.0245    1.0000         0         0
         0         0         0    0.0068    0.3480    1.0000         0
         0         0         0         0   -0.6045    0.4602    1.0000

>> U=diag(u0)+diag(u1,1)+diag(u2,2)

U =

   -0.9219    0.5913    0.0000         0         0         0         0
         0   -2.0056   -0.6436   -0.3179         0         0         0
         0         0    0.0348    0.4267    1.0950         0         0
         0         0         0  -22.9095  -57.6919   -1.8740         0
         0         0         0         0    2.6535    0.0265    0.4282
         0         0         0         0         0    0.5113   -0.1972
         0         0         0         0         0         0    2.0420
```

```
>> norm(A-L*U,'fro')

ans =

   1.0415e-15

>> % Generate a known solution and solve Ax=b
>> x0=randn(n,1)

x0 =

    0.8956
    0.7310
    0.5779
    0.0403
    0.6771
    0.5689
   -0.2556

>> b=A*x0;
>> x=pentaSolve(l2,l1,u0,u1,u2,b)

x =

    0.8956
    0.7310
    0.5779
    0.0403
    0.6771
    0.5689
   -0.2556

>> rel_error=norm(x-x0)/norm(x0)

rel_error =

   2.4202e-15

>> diary off
```

2. Let col(B) denote the column space of the matrix B.

(a) The columns of Q are orthonormal, so $O^TQ = I_k$ and

$$(QQ^T)^2 = (QQ^T)(QQ^T) = Q(Q^TQ)Q^T = QQ^T,$$
$$(QQ^T)^T = (Q^T)^T(Q^T) = QQ^T.$$

So QQ^T is an orthogonal projector. It projects onto the column space of Q, because

$$y = QQ^Tx \quad \Rightarrow \quad y = Qz \text{ with } z = Q^Tx \qquad (\mathrm{col}(QQ^T) \subset \mathrm{col}(Q)),$$
$$y = Qx \quad \Rightarrow \quad QQ^Ty = QQ^TQx = Qx = y \qquad (\mathrm{col}(Q) \subset \mathrm{col}(QQ^T)).$$

(b) Since the columns of A are linearly independent, A^TA is an invertible $k \times k$ matrix:

$$\begin{aligned} A^TAx = 0 \quad &\Rightarrow \quad Ax \text{ is orthogonal to the columns of } A \\ &\Rightarrow \quad Ax = 0 \quad \Rightarrow \quad x = 0 \end{aligned}$$

The matrix $A(A^TA)^{-1}A^T$ is an orthogonal projector, because

$$(A(A^TA)^{-1}A^T)^2 = A(A^TA)^{-1}A^TA(A^TA)^{-1}A^T = A(A^TA)^{-1}A^T$$
$$(A(A^TA)^{-1}A^T)^T = (A^T)^T((A^TA)^{-1})^TA^T = A((A^TA)^T)^{-1}A^T = A(A^TA)^{-1}A^T$$

And it projects onto the column space of A, since

$$y = A(A^TA)^{-1}A^Tx \quad \Rightarrow \quad y = Az \text{ with } z = (A^TA)^{-1}A^Tx,$$
$$y = Ax \quad \Rightarrow \quad A(A^TA)^{-1}A^Ty = A(A^TA)^{-1}A^TAx = Ax = y.$$

Let $\hat{x}$ be the solution to $\min_x ||Ax - b||_2$. Then $A\hat{x}$ is the orthogonal projection of b onto the column space of A:

$$A\hat{x} = A(A^TA)^{-1}A^Tb \quad \Leftrightarrow \quad \hat{x} = (A^TA)^{-1}A^Tb \quad \Leftrightarrow \quad A^TA\hat{x} = A^Tb.$$

(c) Let P be an orthogonal projector onto a subspace S. Then

$$(I-P)^2 = I - 2P + P^2 = I - 2P + P = I - P,$$
$$(I-P)^T = I^T - P^T = I - P.$$

The column space of $I - P$ is the orthogonal complement of S:

$$\begin{aligned} y \in \mathrm{col}(I-P),\ w \in S \quad &\Rightarrow \quad \exists x, z \text{ s.t. } y = (I-P)x,\ w = Pz \\ &\Rightarrow \quad y^Tw = x^T(I-P)^TPz = x^T(P - P^2)z = x^T0z = 0, \end{aligned}$$

$$y \in S^\perp \quad \Rightarrow \quad P^Ty = 0 \quad \Rightarrow \quad (I-P)y = y - Py = y - P^Ty = y.$$

3.
```
%%%%%%%%%%%%%%%%%%%% PART (a)

% Put data into column vectors
x0=[1.02 0.95 0.87 0.77 0.67 0.56 0.44 0.30 0.16 0.01]';
y0=[0.39 0.32 0.27 0.22 0.18 0.15 0.13 0.12 0.13 0.15]';

% Determine parameters by using linear least squares
n=length(x0);
M=[y0.^2 x0.*y0 x0 y0 ones(n,1)];
w=x0.^2;
p=M\w;
a=p(1); b=p(2); c=p(3); d=p(4); e=p(5);

fprintf('Orbital parameters:\n')
fprintf(' a=%6.4f  b=%6.4f  c=%6.4f   d=%6.4f  e=%6.4f\n\n',p)

subplot(2,2,1)
plot(x0,y0,'o')
hold on
PlotEllipse(a,b,c,d,e,'-');
hold off

%%%%%%%%%%%%%%%%%%%% PART (b)

fprintf('Condition number = %8.4e\n\n',cond(M)')

for k=2:4
    % Random perturbation
    x1=x0+0.01*(rand(n,1)-0.5);
    y1=y0+0.01*(rand(n,1)-0.5);

    % Solve the perturbed problem
    M=[y1.^2 x1.*y1 x1 y1 ones(n,1)];
    w=x1.^2;
    p=M\w;
    a1=p(1); b1=p(2); c1=p(3); d1=p(4); e1=p(5);
    fprintf('Parameters for perturbed problem:\n')
    fprintf(' a=%6.4f  b=%6.4f  c=%6.4f   d=%6.4f  e=%6.4f\n\n',p)

    subplot(2,2,k)
    plot(x1,y1,'o')
    hold on
    PlotEllipse(a,b,c,d,e,'-');
    PlotEllipse(a1,b1,c1,d1,e1,'--');
    hold off
end
```

Function to plot orbit (only works if it is an ellipse):

```
function PlotEllipse(a,b,c,d,e,linestyle)
% Plot elliptical orbit given by
%   a*y^2 + b*x*y + c*x + d*y +e = x^2
% Warnig: Only works if orbit really is an ellipse

A=b^2+4*a; B=2*b*d-4*a*c; C=d^2-4*a*e; D=B^2-4*A*C;
xmin=(-B+sqrt(D))/(2*A);  xmax=(-B-sqrt(D))/(2*A);
x=linspace(xmin,xmax,100);
B=b*x+d; C=c*x+e-x.^2; D=B.^2-4*a*C;
y1=(-B-sqrt(D))/(2*a);  y2=(-B+sqrt(D))/(2*a);
plot([x fliplr(x)],real([y1 fliplr(y2)]),linestyle)
xlabel('x')
ylabel('y')
```

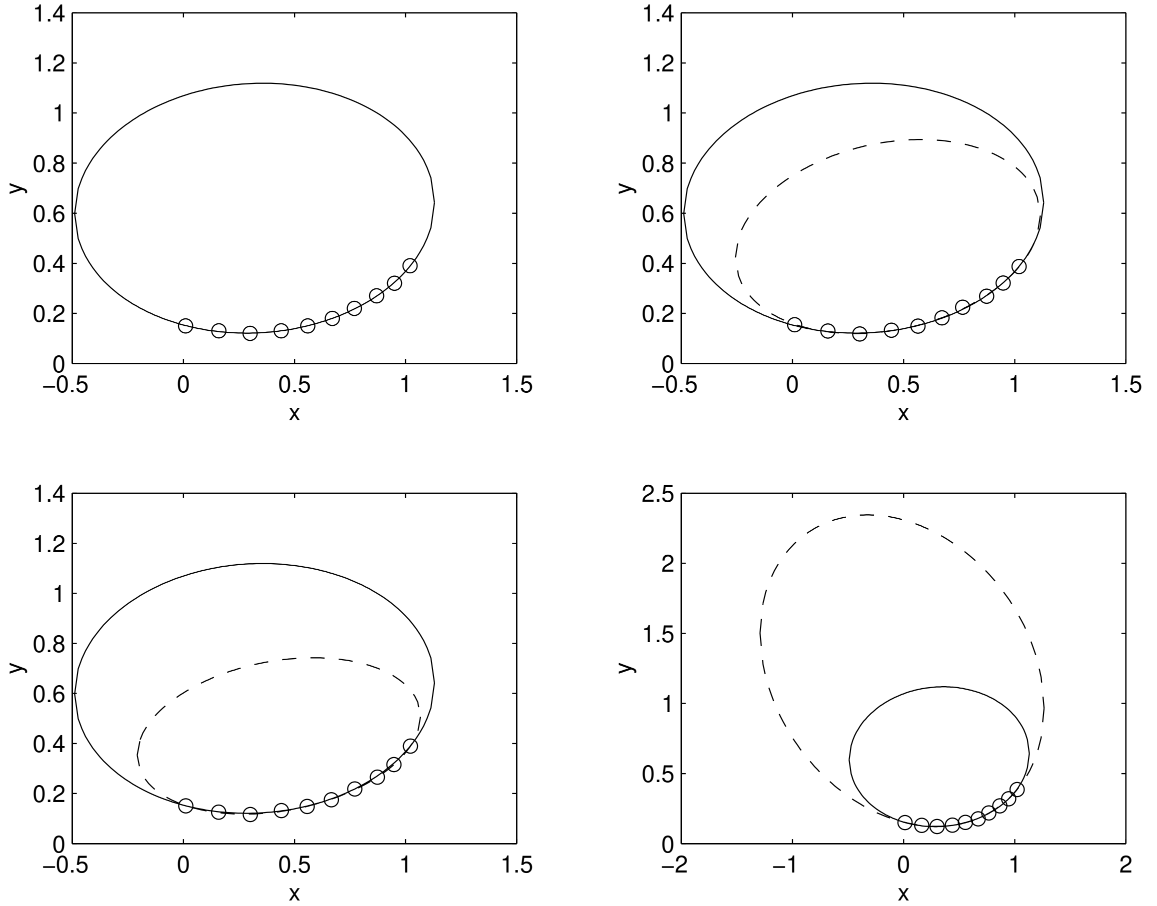
Output:

```
Orbital parameters:
 a=-2.6356  b=0.1436  c=0.5514   d=3.2229  e=-0.4329

Condition number = 6.8843e+02

Parameters for perturbed problem:
 a=-3.1336  b=0.7622  c=0.4720   d=2.8488  e=-0.3747

Parameters for perturbed problem:
 a=-4.1688  b=1.0007  c=0.4287   d=3.1579  e=-0.3896

Parameters for perturbed problem:
 a=-1.3216  b=-0.5549  c=0.6563   d=3.2544  e=-0.4674
```

The relative change in orbital parameters caused by the perturbation is 20.91%, 41.72% and 35.33%, respectively for the three runs (measured in 2-norm). This is a substantial difference and is caused by the relatively big condition number of the coefficient matrix.

The effect of this can be seen in the plots. The difference is biggest at the points furthest away from the observation points.

The size of the condition number can be explained by noting that the observations are only made on a part of the orbit. A wider spread of observation points would give a better conditioned problem.



Solid line: Original solution. Dashed line: Perturbed solution.