CS 322: Practice Final Exam Solution

1. Assume that `P`, `A`, `q`, and `T` are given column n-vectors that represent "location data" for n separate comets. In particular,

$$x_i(t) = \frac{P_i - A_i}{2} + \frac{P_i + A_i}{2}\cos\left(\frac{2\pi}{T_i}t + q_i\right)$$

$$y_i(t) = \sqrt{P_i A_i}\sin\left(\frac{2\pi}{T_i}t + q_i\right)$$

specifies the location of the i-th comet at time t. You may assume that $0 < P_i \leq A_i$ and $T_i > 0$ for $i = 1{:}n$. Write a MATLAB script that plots in a single window each of these elliptical orbits. Do not worry about color, axis scaling, labelling, etc. Each plot must be based upon 200 points to ensure a smooth rendition of the orbit. Your script should be vectorized and flop efficient.

15 points

The period T and phase delay q have NOTHING to do with the shape of the orbit. That depends only on the P and A values:

```
tau = linspace(0,2*pi,200);
c = cos(tau);
s = sin(tau);
hold on
for i=1:length(T)
   x = (P(i)-A(i))/2 + ((P(i)+A(i))/2)*c;
   y = sqrt(P(i)*A(i))*s;
   plot(x,y)
end
hold off
```

-10 have plot outside the loop so only one orbit
-3 for linspace(0,T(i),200) inside the loop
-5 for linspace(0,2pi,200) outside the loop
-2 for linspace(0,max(T)/2pi,200 outside the loop
-2 for another plotting loop
-5 for trying to use P and A without loops

2. Assume that you are given an n-by-n nonsingular matrix `A` and an n-by-4 matrix `B`. Write a MATLAB script that determines a column 4-vector `d` so that if

$$p(t) = d_1 + d_2 t + d_3 t^2 + d_4 t^3$$

then $p(i) = c_{ii}$ for $i = 1{:}4$ where $C = B^T A^{-1} B$. Make effective use of the \ operator.

20 points

```
X = A\B;
rhs = zeros(4,1);
for i=1:4
   rhs(i) = B(:,i)'*X(:,i);
end
V = [1 1 1 1 ; 1 2 4 8 ; 1 3 9 27; 1 4 16 64];
d = V\rhs;
```

12points for getting the rhs (the diagonal of C) and 8 points for solving Vd = rhs.

-5 if you compute all of C

-7 if you use `inv` because `inv(A)*z` is about 2-3 times as expensive as `A\z`. And you were asked to use backslash.

OK to use LU. Can also get X vis $X(:,j) = A\backslash B(:,j), \qquad j = 1{:}n$.

3. Assume that `a`, `b`, and `T` are given and that `F` is an implementation of a function $F(x)$ that has period `T`. The command

```
numI = quad('F',a,b,.000001)
```

assigns to `numI` an estimate of

$$I = \int_a^b F(x)dx$$

that (usually) has absolute error less than .000001. Write a MATLAB script that does the same thing more efficiently assuming that $b = a + (10/3)T$. You may assume that `F`-evaluations are very expensive and that F is uniformly behaved from a to $a + T$. Make sure your solution has an absolute error less than .000001.

15 points:

```
tol = .000001/7;
I1 = quad('F',a,a+T/3,tol);
I2 = quad('F',a+T/3,a+T,tol);
I = 4*I1 + 3*I2;
```

12 points for this:

```
tol = .000001/4;
I1 = quad('F',a,a+T,tol);
I2 = quad('F',a+T/3,a+T,tol);
I = 3*I1 + 1*I2;
```

For either of these, 5 points was for the correct `tol` adjustment.

4. Assume that `z` and `f` are given column 6-vectors with

$$1 < z_1 < z_2 < 2 < z_3 < z_4 < 3 < z_5 < z_6 < 4.$$

We wish to determine a column 4-vector `y` so that if the continuous piecewise linear function L is defined by

$$L(t) = y_i + (t-i)(y_{i+1} - y_i) \qquad i \le t \le i+1$$

then

$$\phi(y_1, y_2, y_3, y_4) = \sum_{i=1}^{4}(L(z_i) - f_i)^2$$

is minimized. Write a MATLAB script that determines y by solving a least squares problem of the form min $\| Ay - f \|_2$ where A is an 6-by-4 matrix. Make effective use of the \ operator. Do NOT use `fmins`. Do not worry about vectorization.

15 points
Since

$$L(t) = (i+1-t)y_i + (t-i)y_{i+1} \qquad i \le t \le i+1$$

we see that

```
A = zeros(6,4);
A(1,1) = 2-z(1); A(1,2) = z(1)-1;   % Set t = z(1), i = 1 in the above
A(2,1) = 2-z(2); A(2,2) = z(2)-1;   % Set t = z(2), i = 1 in the above
A(3,2) = 3-z(3); A(3,3) = z(3)-2;   % Set t = z(3), i = 2 in the above
A(4,2) = 3-z(4); A(4,3) = z(4)-2;   % Set t = z(4), i = 2 in the above
A(5,3) = 4-z(5); A(5,4) = z(5)-3;   % Set t = z(5), i = 3 in the above
A(6,3) = 4-z(6); A(6,4) = z(6)-3;   % Set t = z(6), i = 3 in the above
y = A\f;
```

Roughly 2 points per row.

5. The backwards Euler method for the initial value problem $y' = f(t, y)$, $y(t_0) = y_0$ is defined by

$$y_{n+1} = y_n + h_n f(t_{n+1}, y_{n+1})$$

where $t_{n+1} = t_n + h_n$. Write a `Matlab` script that makes effective use of this method to produce a plot of $y(t)^T y(t)$ across the interval $[0, 10]$ where

$$y'(t) = Ay(t) + U(t) \qquad y(0) = y_0$$

Assume that `A` is a given, m-by-m matrix and that `U(t)` is an implementation of the function U that returns a column m-vector for any given scalar t. Assume that the initial vector `y0` is available. The plot should be based on estimates of $y(t)$ at $t =$`linspace(0,10,201)`. Make effective use of `[L,U,P] = lu(C)` that returns the factorization $PC = LU$. Do not use `ode23` or any other MATLAB initial value problem solver.

20 points
The step: $(I - h_n A)y_{n+1} = y_n + h_n U(t_{n+1})$

```
h = 10/200;
t = linspace(0,10,201)
[m,m] = size(A);
C = eye(m,m) - h*A;
[L,U,P] = lu(C)
z = zeros(201,1);
z(1) = y0'*y0;
for n=1:200
   ynext = U\L\P*(y0 + h*U(t(n+1)));
   z(n+1) = ynext'*ynext;
   y0 = ynext;
end
plot(t,z);
```

Getting the matrix is 5 points.
Factoring it outside the loop is 4 points.
`z(1)` is 3 points
`ynext` is 5 points
Plotting and `z` is 3 points.

6. Assume that cubic splines S_x, S_y, and S_z interpolate the data (x_i, y_i, z_i), $i = 1{:}n$ in the sense that

$$(S_x(t_i), S_y(t_i), S_z(t_i)) \;=\; (x_i, y_i, z_i) \qquad i = 1{:}n$$

Assume that $0 = t_1 < t_2 < \cdots < t_n = 1$. Let

$$d(t) = |S_x(t)| \;+\; |S_y(t)| \;+\; |S_z(t)|.$$

(a) Write a MATLAB script that computes a scalar t_* so that $d(t_*) = (d(0) + d(1))/2$. You must make use of the method of bisection. (Write it from scratch, do not invoke any Chapter 8 function.) The absolute error of your computed t_* must be less than or equal to `tol` where `tol` is a given positive scalar. Assume that `Sx`, `Sy`, and `Sz` are given implementations of the three splines and that the data is represented in the arrays `x`, `y` and `z`. Recall that if `S` is a cubic spline and `u` is a scalar, then `ppVal(S,u)` is the value of the spline at `u`.

(b) What can go wrong if the value of `tol` is too small? Explain.

15 points

```
d0 = abs(x(1)) + abs(y(1)) + abs(z(1));
n = length(x);
d1 = abs(x(n)) + abs(y(n)) + abs(z(n));
ave = (d0+d1)/2;

L = 0; SL = d0-ave;
R = 0; SR = d1-ave;
while R-L>2*tol
   mid = (R+L)/2;
   Smid = abs(ppVal(Sx,mid)) + abs(ppval(Sy,mid)) + abs(ppval(Sz,mid));
   if Smid*SL<=0
      R=mid; SR = Smid;
   else
      L= mid; SL = Smid;
   end
end
tstar = (R+L)/2;
```

If `tol` is smaller than `EPS` the iteration may not terminate because there will eventually be no floating point numbers in between `L` and `R`.
Bisection on the wrong function -5.
Mistakes because you assume d is monotone increasing up to -8.
3 points for the tol-too-small answer
-4 if more than one function evaluation per iteration