CS 322: Prelim 1 and 2 Make-Up

1. (15 points) Assume that `x` and `y` are given column vectors and that the command `plot(x,y)` results in the display the ellipse $\mathcal{E}$ defined by

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 = 1.$$

For a given cosine-sine pair (c, s) define the ellipse

$$\mathcal{E}_\theta = \{(\tilde{x}, \tilde{y}) : \tilde{x} = \cos(\theta)x - \sin(\theta)y,\ \tilde{y} = \sin(\theta)x + \cos(\theta)y,\ (x, y) \in \mathcal{E}\}$$

(This amounts to a rotation of $\mathcal{E}$ by θ radians.) Making effective use of `x` and `y`, show how to plot the ellipse $\mathcal{E}_{\pi/4}$. Do not worry about axis scaling. You may use `hold` and `plot` but no other MATLAB function. Express your solution in the form of a MATLAB script.

Solution

```
c = 1/sqrt(2); xtilde = c*(x-y); ytilde = c*(x+y); plot(xtilde,ytilde)

% The following is 50 percent slower...
c = cos(pi/4); s = sin(pi/4); xtilde = c*x-s*y; ytilde = s*x+c*y; plot(xtilde,ytilde) % 10pts
```

2. (a) (10 points) Assume that $u \in \mathbb{R}^n$, $v \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ are stored in n-by-1 arrays `u`, `v`, and `x`. Write a MATLAB script that efficiently evaluates $y = (uv^T)^3x$ and assigns the result to `y`. (Notation: If A is a square matrix then $A^3 = A \cdot A \cdot A$.)

Solution
Since $y = (uv^T)(uv^T)(uv^T)x = u(v^Tu)(v^Tu)(v^Tx))$ we get

```
alfa = v'*u; beta = v'*x; y = (alfa^2*beta)*u      % O(n)   flops   10 points

A = u*v'; y = A*(A*(A*x));                          % O(n^2) flops,   5 points

A = u*v'; y = (A*A*A)x                              % O(n^3) flops    2 points
```

(b) (5 points) If it is applied to a floating point number x, then the MATLAB `exp` function returns the nearest floating point number to the exact value of $\exp(x)$. Does that mean that the absolute error is always less than the machine precision `EPS`? Explain.

Solution
If $\hat{y}$ is the computed version of $y = e^x$ then

$$\frac{|\hat{y} - y|}{|y|} \approx \text{EPS}$$

That is, $|\hat{y} - y| \approx \text{EPS}|y| \approx$ the spacing of the floating point numbers near y. Clearly this can be much greater than `EPS` because $y = e^x$ can be very large.

3. Suppose we are given the data $(x_1, y_1), \ldots, (x_n, y_n)$ with $x_1 < \cdots < x_n$. A cubic spline interpolant S has the property that $S(x_i) = y_i$, $i = 1{:}n$. Moreover, S, S' and S'' are continuous on $[x_1, x_n]$ and S is a cubic on each subinterval $[x_i, x_{i+1}]$.

(a) (10 points) What property does the not-a-knot spline have?

Solution
S''' is continuous at x_2 and x_{n-1}. This means that the first and second local cubics are identical and the last and second-to-last local cubics are identical.

(b) (5 points) Assume that `x` and `y` are given column n-vectors and that `x(1)` $< \cdots <$ `x(n)`. In `Matlab`, `S = spline(x,y)` assigns a representation of the not-a-knot spline interpolant to `S`. How could `spline` be used to generate an interpolant of this data with the condition that the spline's slope is *approximately* zero at `x(1)` and `x(n)`? Hint: Augment `x` and `y`.

Solution
If two abscissae are close together in an interpolant of $f(x)$ then the interpolant will roughly have the same slope as f near these abscissae.

```
n = length(x);
delta = .00001 % or some other small number
xtilde = [x(1)-delta ; x ; x(n)+delta];
ytilde = [y(1); y ; y(n)];
Stilde = spline(xtilde,ytilde)
```

4. (a) (10 points) Does it follow that a 3-point Newton-Cotes rule is more accurate than the 2-point Newton-Cotes rule? Explain.

Solution
The error in the 3-point rule depends on a higher derivative than the error for the 2-point rule. If that higher derivative is more poorly behaved then it could be that the 3-point rules is more inaccurate.
(b) (10 points) Assume that the function f is implemented in `f.m` and that if `numI = QUAD('f',L,R)`, then `numI` is a "good enough" approximation to

$$I = \int_L^R f(x)dx.$$

For any positive integer k define

$$I_k = \int_0^k f(x)dx.$$

Write an efficient MATLAB script that sets up a column 10-vector `v` with the property that `v(k)` a "good enough" approximationto I_k for $k = 1{:}10$.

Solution

```
v = zeros(10,1);
v(1) = QUAD('f',0,1);
for k=2:10
   v(k) = v(k-1) + QUAD('f',k-1,k);
   % v(k) = QUAD('f',0,k)                % -8 points
end
```

5. The Euler method for the initial value problem

$$\dot{y} = f(t, y) \qquad y(t_0) = y_0$$

is given by

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

The backwards Euler method for the same problem is given by

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

Suppose $A \in \mathbb{R}^{n\times n}$ and $y_0 \in \mathbb{R}^n$ are given and stored in MATLAB arrays `A` and `y0` and consider the problem $\dot{y} = Ay,\ y(t_0) = y_0$.

(a) For fixed step length $h > 0$ write an efficient `Matlab` script that generates an n-by-N array `Y` with the property that `Y(:,k)` is the Euler approximation to $y(t_k)$ for $k = 1{:}N$. Assume that `h` and `N` are given.

Solution

Since $y_{k+1} = y_k + hAy_k$ we get

```
Y = zeros(n,N);
for k=1:N
   Y(:,k) = y0 + h*(A*y0)       % Note that h*A*y0 involves 50 percent more work than h*(A*y0)
   y0 = Y(:,k);
end
```

(b) For fixed step length $h > 0$ write an efficient `Matlab` script that generates an n-by-N array `Y` with the property that `Y(:,k)` is the backwards Euler approximation to $y(t_k)$ for $k = 1{:}N$. Assume that `h` and `N` are given.

Solution

Since $y_{k+1} = y_k + hAy_{k+1} = (I - hA)^{-1}y_k$ we get

```
Y = zeros(n,N);
[L,U,P] = LU(eye(n,n) - h*A);   % -5 for not getting this out of the loop.
for k=1:N
   Y(:,k) = U\(L\P*y0))
   y0 = Y(:,k);
end
```

6. Suppose we are given n data points $(x_1, y_1), \ldots, (x_n, y_n)$ and an additional point (h, k). We want to compute the minimum value of

$$\phi(r) = \sum_{i=1}^{n} d_i$$

where d_i is the minimum euclidean distance from (x_i, y_i) to a point on the circle $(x - h)^2 + (y - k)^2 = r^2$. Assume that we are given column n-vectors `x` and `y` that house the data points and scalars `h` and `k`.
(a) (5 pts) Write a `Matlab` script that makes effective use of

```
z = fmin('F',L,R,Options,P1,P2,..) attempts to return a value of z which is a local
  minimizer of F(z,P1,P2) in the interval L < z < R.  'F' is a string
  containing the name of the objective function to be minimized and P1, P2,..
  are its parameters.
```

and assigns to `rBest` a local minimizer of ϕ. (You may ignore `Options` in this problem.) Give a justification for the choice of the search interval endpoints `L` and `R` used by your script. **(b)** (10 pts) Give a complete implementation of the objective function that your script passes to `fmin`. Vectorize and be efficient in both parts of this problem.

Solution

See Prelim 2, problem 5.