Handed out September 27, 2000
Due back October 4, 2000
Derivatives, Splines, Edge Detection

Generate 65 points at intervals of 0.1 radian (theta=`[0:0.1:6.4];`) along an Archimedean spiral given by the equation $r=2\theta$
(For more on this curve see http://mathworld.wolfram.com/ArchimedesSpiral.html).
Get the x and y coordinates of your points by converting from polar coordinates to Cartesian coordinates using $x=r\cos(\theta)$ and $y=r\sin(\theta)$
(In Matlab use `x=r.*cos(theta); `).

1. Generate a parameterization along this curve using chord lengths.

   $$s(1)=0, \qquad s(k)=s(k-1)+\sqrt{(x_k-x_{k-1})^2+(y_k-y_{k-1})^2}$$

   The three column vectors x, y and s sample the parametric representation of the curve $x(s)$, $y(s)$.
2. Using the function `polyfit` and the three vectors x, y and s fit polynomial functions of 2nd, 4th, and 16th order for $x(s)$ and $y(s)$. Evaluate these functions at the midpoints of the chords using `polyval` and plot the resulting curve.
3. Read chapter 3.3 of Numerical recipes. To those who do not have the book, you can find a copy on the web at
   http://www.ulib.org/webRoot/Books/Numerical_Recipes/bookcpdf/c3-3.pdf
   (Note that the link given in previous handouts at www.lanl.gov seems to be offline this week)
4. Fit cubic splines through the same data using the function `spline`, and evaluate them at the same points using `ppval`. Plot your curve.
5. You can evaluate the derivatives dx/ds and dy/ds and d^2x/ds^2 and d^2y/ds^2 along the spline by using the code snippet given at
   http://www.mathworks.com/support/solutions/data/3313.shtml
   Implement this snippet, and evaluate dx/ds and dy/ds. Next estimate dy/dx at the points `s(1)` through `s(64)` using

   $$\frac{dy}{dx}=\frac{dy/ds}{dx/ds}$$

   Compare your results with the analytical expression obtained from

   $$\frac{dy}{dx}=-\frac{2\left(\tan\frac{1}{2}\sqrt{(x^2+y^2)}\right)\sqrt{(x^2+y^2)}+x^2+\left(\tan^2\frac{1}{2}\sqrt{(x^2+y^2)}\right)x^2}{-2\sqrt{(x^2+y^2)}+xy+x\left(\tan^2\frac{1}{2}\sqrt{(x^2+y^2)}\right)y}$$

6. Write a function that applies the Roberts operators to a given image to find the gradient components in two directions and compute the magnitude and direction of the gradient vector at every pixel. Your function should perform the non-maxima suppression step discussed in class. Convert the `flowers.tif` image to grayscale using `rgb2ray` and apply your function to it.
   Write a little Matlab function to blur the image using a one-dimensional Gaussian filter with a user specified filter size. Apply your filter successively along the x and y directions with a filter size of 10. Obtain the edge image again using your function on the smoothed image.