Cubic Spline interpolation of a Parametric Equation

In this example, we will take samples from a parametric curve \((x(t), y(t)), a \leq t \leq b\), and interpolate each variable by a natural cubic spline, to gain a representation for the entire curve.

Below is the diary of a matlab session that does this. We begin by sampling a spiral:

\[
\begin{align*}
t & = 0 : .25 : 6; \quad t = \pi \times 6; \\
x & = \cos(t) \times t; \quad y = \sin(t) \times t; \\
\text{plot}(x, y, 'o')
\end{align*}
\]

The last matlab command plots the points \((x, y)\) and points an ‘o’ at that point:

We shall use the m-file that determines the coefficients for the piecewise polynomial representation of the spline for each coordinate. We first pull up the help for the m-file:

\[
\text{help spline3_coeff}
\]

\[
\text{\textbackslash n} \text{function} \quad [a, b, c, d] = \text{spline3}\_\text{coeff}(\text{knots, data}); \\
\text{\textbackslash n} \text{Modification of Cheney Kincaid pseudo-code} \\
\text{\textbackslash n} \text{The natural spline for interpolating data at the knots a=0<....<b;} \\
\text{\textbackslash n} \text{System of equations is solved using the tridiagonal nature}
\]

Figure 1: Interpolation Points
WARNING: THIS PROGRAM REQUIRES AT LEAST 4 KNOTS

INPUT:
knots - distinct points (t_j) of interpolation as a row vector
data - data at the interpolation points as a row vector

OUTPUT:
a - column vector of constant terms for the spline on [t_j, t_j-1]
b - column vector of coefficients of x-t_j for the spline on [t_j, t_j-1]
c - column vector of coefficients of (x-t_j)^2 for the spline on [t_j, t_j-1]
d - column vector of coefficients of (x-t_j)^3 for the spline on [t_j, t_j-1]

Since we want to do the x and y coordinates separately, we use subscripts to distinguish them.

[a1,b1,c1,d1]=spline3_coeff(t,x);
a2,b2,c2,d2]=spline3_coeff(t,y);

We will evaluate the curve at many more points. Here tt will represent the points taken in the parametrization. Then we obtain the x-coordinates for these points from evaluating the spline that interpolated the x-values of the data points, and the y-coordinates for these points from evaluating the spline that interpolated the y-values of the data points

% function y=spline3_eval(a,b,c,d,knots,xx);
% Compute the values at the points xx of the cubic spline with knots ‘‘knots’’ from the coefficients of its polynomial pieces
% INPUTS: (from output of spline3_coeff.m)
% knots - t_j
% a - column vector of constant terms for the spline on [t_j, t_j-1]
% b - column vector of coefficients of x-t_j for the spline on [t_j, t_j-1]
% c - column vector of coefficients of (x-t_j)^2 for the spline on [t_j, t_j-1]
% d - column vector of coefficients of (x-t_j)^3 for the spline on [t_j, t_j-1]
% xx - points at which spline is to be evaluated
% OUTPUT:
% y - value of the spline

x1=spline3_eval(a1,b1,c1,d1,t,tt);
y2=spline3_eval(a2,b2,c2,d2,t,tt);
figure
plot(x,y,'o',x1,y2)
diary off
Here in the plot command, we are plotting the data points marked with ‘o’, and the spline curves \((x_1(tt), y_1(tt))\) traversing through those points.