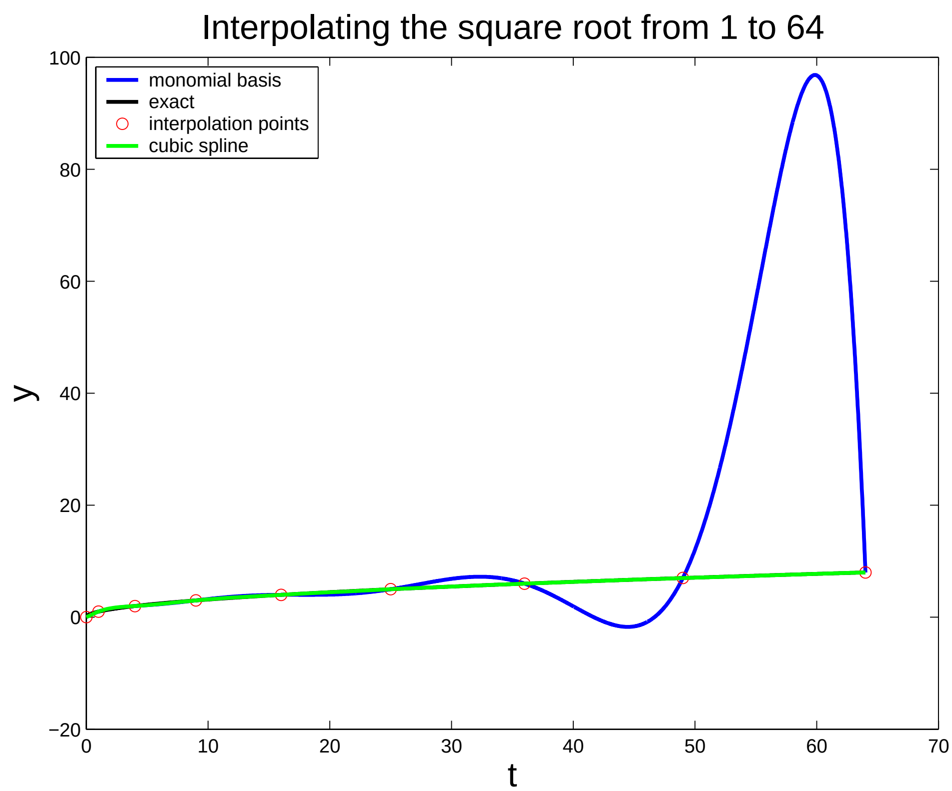
ASSIGNMENT 8 - Solution

Problem 8: Computer Problem 7.6 on p.337 of Heath - Solution

Please take a look at my version of the program for this problem. The exact square root function in the domain $[0, 64]$ is shown in black. The nine data points are plotted as red circles.

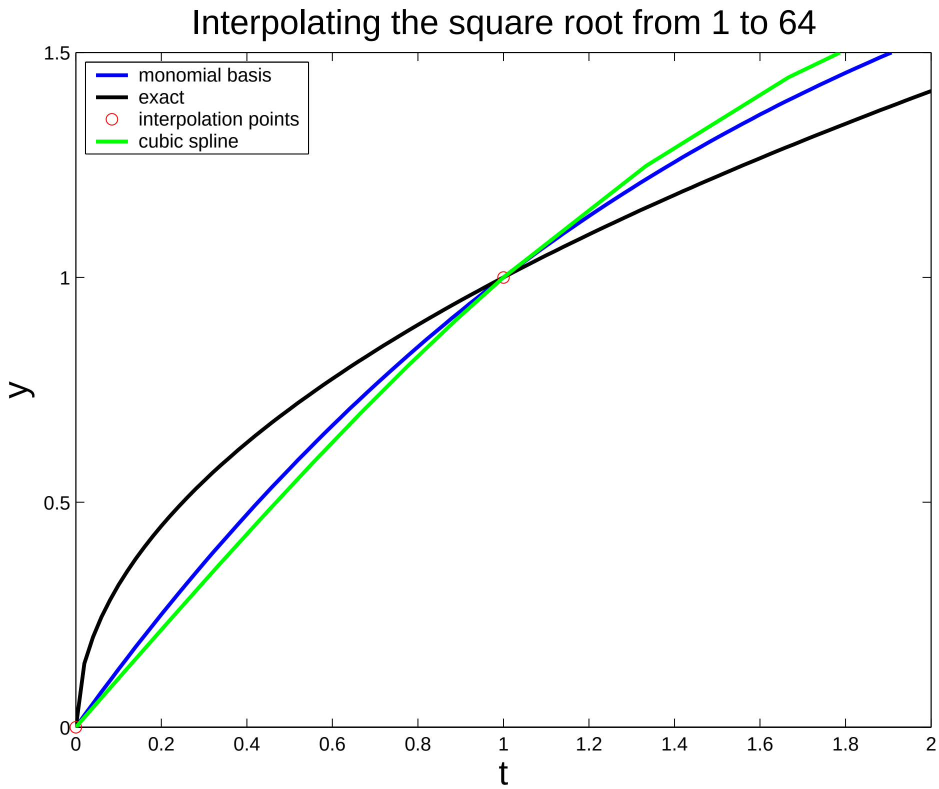
Part (a) - We use a polynomial of degree eight to interpolate the nine data points. The monomial basis is used to obtain the coefficients of the polynomial. One can use other methods such as the Lagrange interpolation or the Newton Interpolation. Of course the resulting polynomial is always the same. To generate the curve to plot the interpolant, the Horner method is employed to evaluate the polynomial (using the function from the course website). One can also use Matlab's polynomial evaluation function `polyval`. However you need to be aware that Matlab writes the coefficient vector in the reversed order. The resulting curve is shown in blue. It is especially bad for large t because the higher powers in t in the polynomial become more dominant there.

Part (b) - We then use a cubic spline interpolation and the resulting interpolant is shown in green. Overall it produces a better interpolant over the domain $[0, 64]$.



Part (c) - The cubic spline interpolation is overall more accurate over the domain $[0, 64]$.

Part (d) - However polynomial interpolation gives a better result when compared with cubic spline in the small t domain. High power terms in the polynomial do not contribute much when t is small. On the other hand the natural cubic spline interpolation always assumes that the function has zero second derivatives at the end points, *i.e.* at $t = 0$ and $t = 64$. In reality the second derivative of the square root function at 0 is actually infinite. Therefore the first segment of the cubic spline starts out too flat.


Interpolating the square root from 1 to 64
monomial basis
exact
interpolation points
cubic spline
y
t
0
0.5
1
1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2