Introduction to computing and data handling 2023

6th class Interpolation-1 *13*th Sep

1. Read the attached data to your program. Write a linear interpolation code and obtain *V-I* color (ie., the difference between *V* and *I* magnitudes) at a given time. Calculate this for all the epochs where there is V-band data, and plot it vs time.

2. Sample the range $-1 \leq x \leq 1$ using 100 points and plot $f(x) = \frac{1 + \tanh(2\alpha x)}{2}$ for $\alpha = 1, 3, 8,$ and 10. See how the function behaves. It should approach a Heaviside step function.

   Now, take $\alpha = 10$ case. Make another x array where the same range is sampled by 11 data points. Write your own Lagrange interpolation code and interpolate through this 11 data points. Overplot the interpolated function on the original. See how high order polynomial interpolation can cause oscillations.

7th class Interpolation-2 *19*th Sep

Cubic Spline

Import the *scipy.interpolate* package.

>>>import scipy.interpolate as si

Make an array of *x* from 0 to 10 with 11 elements. Obtain $y = \cos(-x^2/9)$. Now make a new array *xnew* from 0 to 10 with 1001 elements.

Construct a cubicspline class instance named *spl* This instance can be evaluated at the newly constructed points *xnew*.

>>>spl = si.CubicSpline(x,y,bc_type='natural')

>>>ynew = spl(xnew)

Plot the data points *x,y* and the points interpolated from these arrays, *xnew,ynew* on the same figure.

Now, obtain the first, second, and 3rd order derivatives of *ynew*. Plot all these on top of the interpolated function. Carefully observe the figure and if your results are correct.

To find the derivative numerically, one can use the expression $y' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j}$.

However, the instance *spl* can be called with an argument *nu* whose value represent the order of the derivative. Calculate the 1st, 2nd, and 3rd order derivatives using the *spl* instance. Compare with the output of your own code for the derivatives.