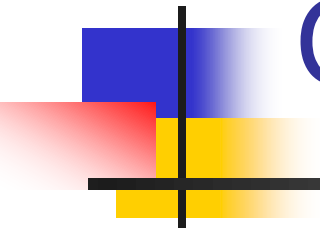
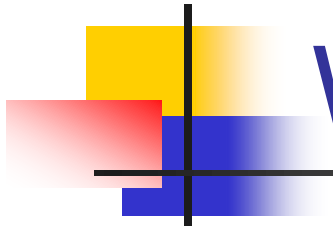


# Computer Methods (MAE 3403)

## Advanced interpolation methods



# We've covered so far

- Linear + Cubic Spline interpolation

- There are more complicated and convenient interpolation methods implemented in Python

# Lagrange Polynomial Interpolation

- Cubic spline: joins multiple cubic polynomials

- Lagrange polynomial *L(x)*: finds a single polynomial that goes through all points.

$$L(x) = \sum_{i=1}^{n} y_i P_i(x)$$

$$P_i(x) = \prod_{j=1, j \neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

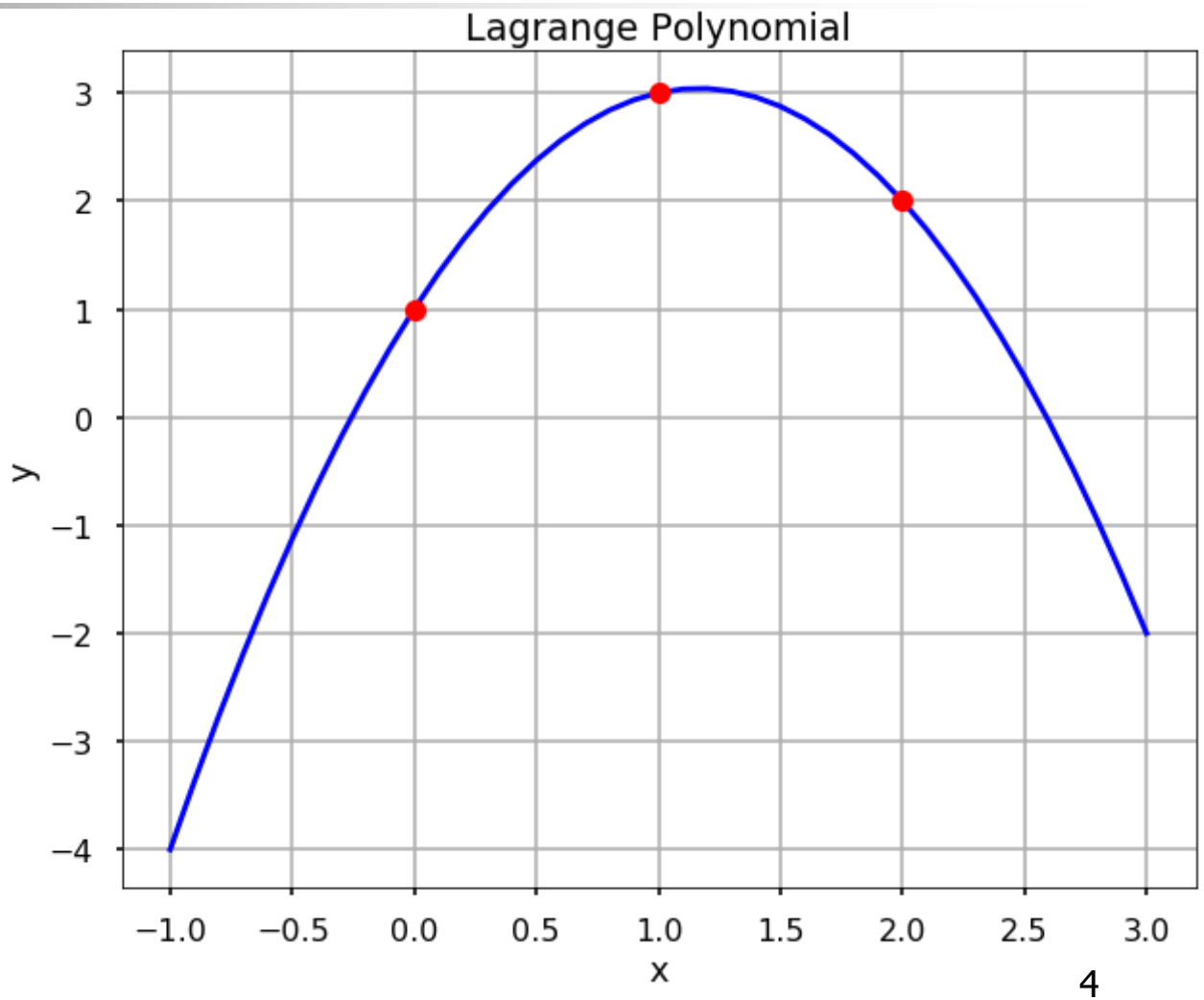
- Can you verify $L(x_i) = y_i$?

# Implementation

- lagrange function in scipy.interpolate does everything for us

```
from scipy.interpolate import lagrange
f = lagrange(x, y)
fig = plt.figure(figsize = (10,8))
plt.plot(x_new, f(x_new), 'b', x, y, 'ro')
```

Refer to interp_example.py



# griddata from scipy.interpolate

- Interpolate unstructured D-dimensional data
- Reference from https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.griddata.html
- Refer to griddata_exp.py

scipy.interpolate.

## griddata

**griddata**(*points*, *values*, *xi*, *method*='linear', *fill_value*=nan, *rescale*=False) [source]

Interpolate unstructured D-D data.

**Parameters:**

**points** : *2-D ndarray of floats with shape (n, D), or length D tuple of 1-D ndarrays with shape (n,).*

Data point coordinates.

**values** : *ndarray of float or complex, shape (n,)*

Data values.

**xi** : *2-D ndarray of floats with shape (m, D), or length D tuple of ndarrays broadcastable to the same shape.*

Points at which to interpolate data.

**method** : *{'linear', 'nearest', 'cubic'}, optional*

Method of interpolation. One of

`nearest`

return the value at the data point closest to the point of interpolation. See `NearestNDInterpolator` for more details.

`linear`

tessellate the input point set to N-D simplices, and interpolate linearly on each simplex. See `LinearNDInterpolator` for more details.

`cubic` **(1-D)**

return the value determined from a cubic spline.

`cubic` **(2-D)**

return the value determined from a piecewise cubic, continuously differentiable (C1), and approximately curvature-minimizing polynomial surface. See `CloughTocher2DInterpolator` for more details.