

Approximation of a Function

Computational Physics

Approximation of a Function

Outline

- Interpolation Problem
- Interpolation Schemes
 - Nearest Neighbor
 - Linear
 - Quadratic
 - Spline
- MATLAB spline function

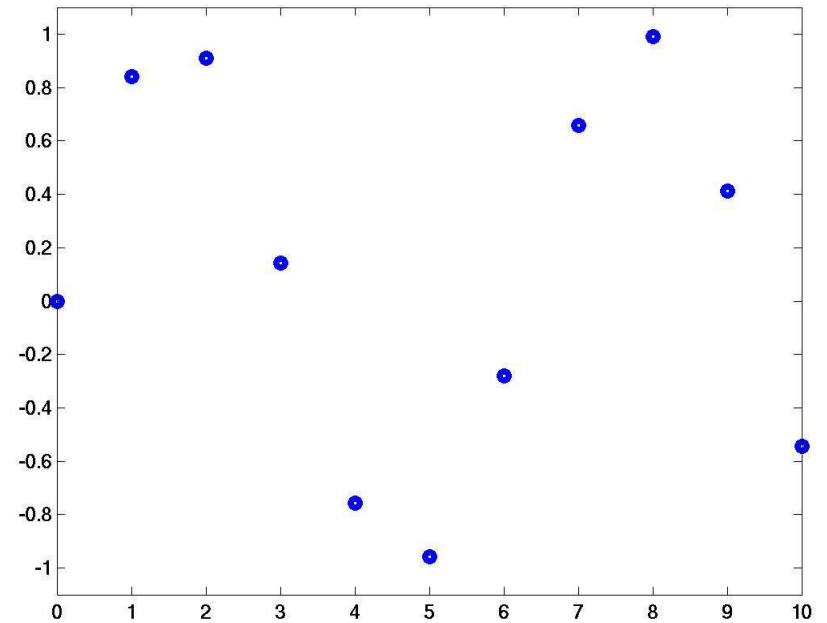
Calculations result in Tables

| Index | T | Y |
|-------|----|-------|
| 1 | 0 | 0 |
| 2 | 1 | 0.84 |
| 3 | 2 | 0.91 |
| 4 | 3 | 0.14 |
| 5 | 4 | -0.76 |
| 6 | 5 | -0.96 |
| 7 | 6 | -0.28 |
| 8 | 7 | 0.66 |
| 9 | 8 | 0.99 |
| 10 | 9 | 0.41 |
| 11 | 10 | -0.54 |

**Interpolation used to find value
between calculated points**

Interpolation

- Nearest Neighbor
- Linear
- Quadratic
- Spline



Basis

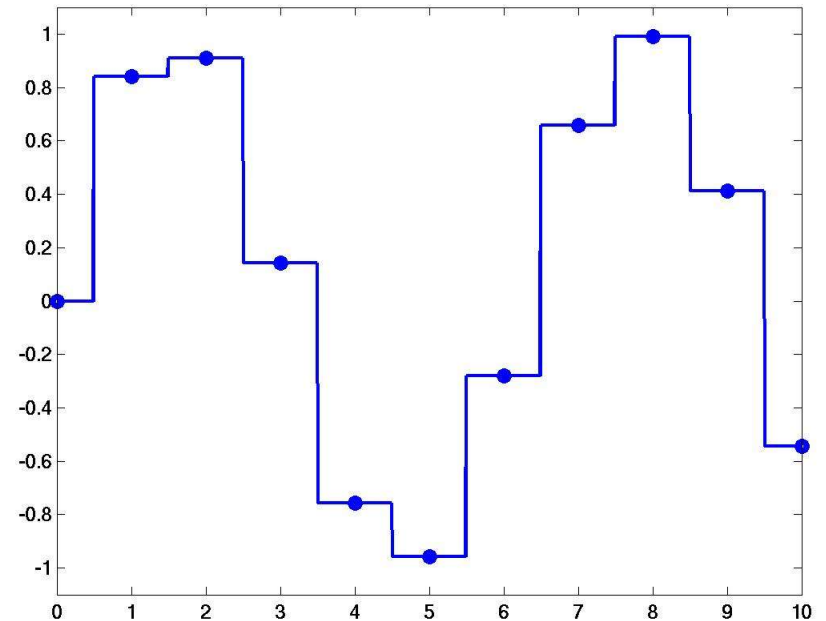
- Taylor Series Expansion of a function
 - We can expand a function, $y(t)$, about a specific point, t_0 according to:

$$y(t) = y(t_0) + \frac{dy}{dt}(t - t_0) + \frac{1}{2} \frac{d^2y}{dt^2}(t - t_0)^2 + \dots + \frac{1}{n!} \frac{d^n y}{dt^n}(t - t_0)^n + \dots$$

- The Taylor Series is used to approximate behavior of functions with a few terms.
- Approximation gets better with fewer terms as $(t-t_0)$ becomes small.

Interpolation

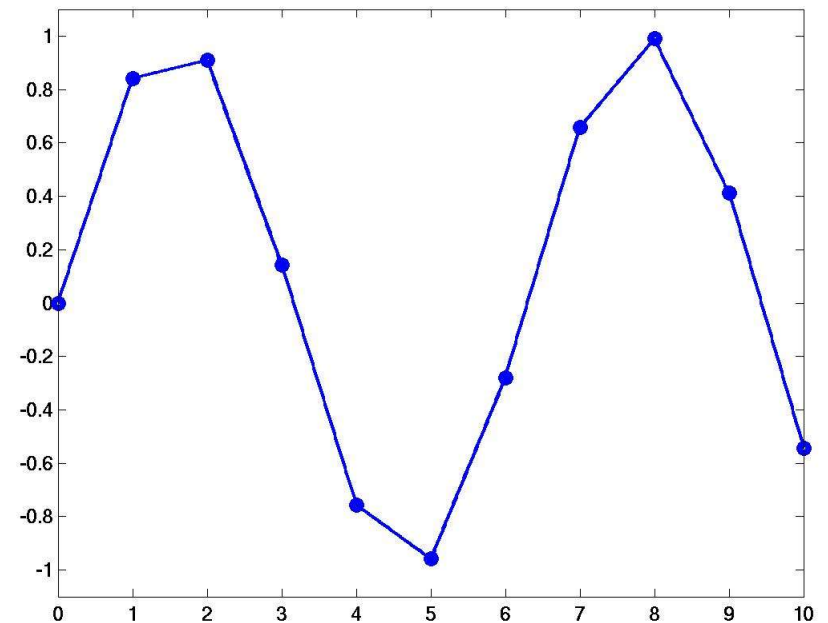
- Nearest Neighbor
- Linear
- Quadratic
- Spline



$$y(t) = y_i$$

Interpolation

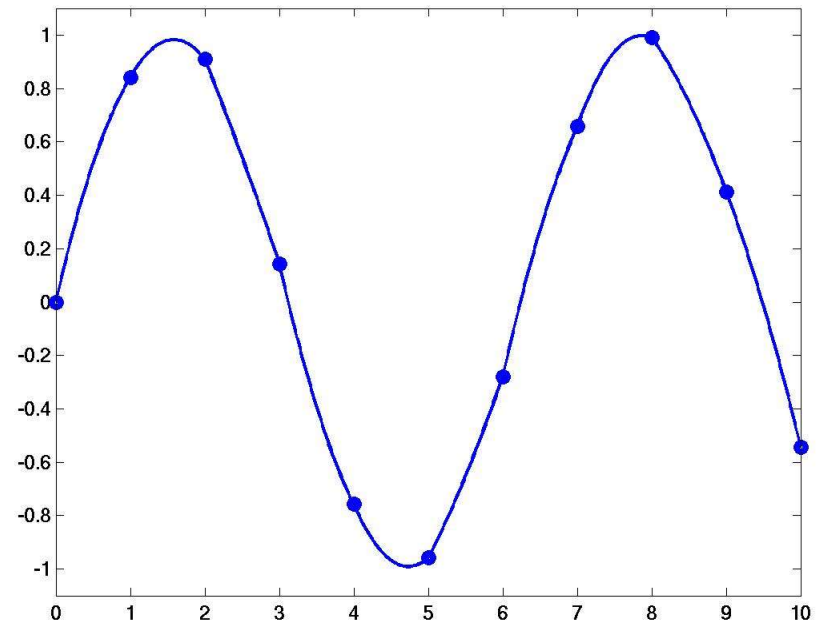
- Nearest Neighbor
- **Linear**
- Quadratic
- Spline



$$y(t) = y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i} \times (t - t_i)$$

Interpolation

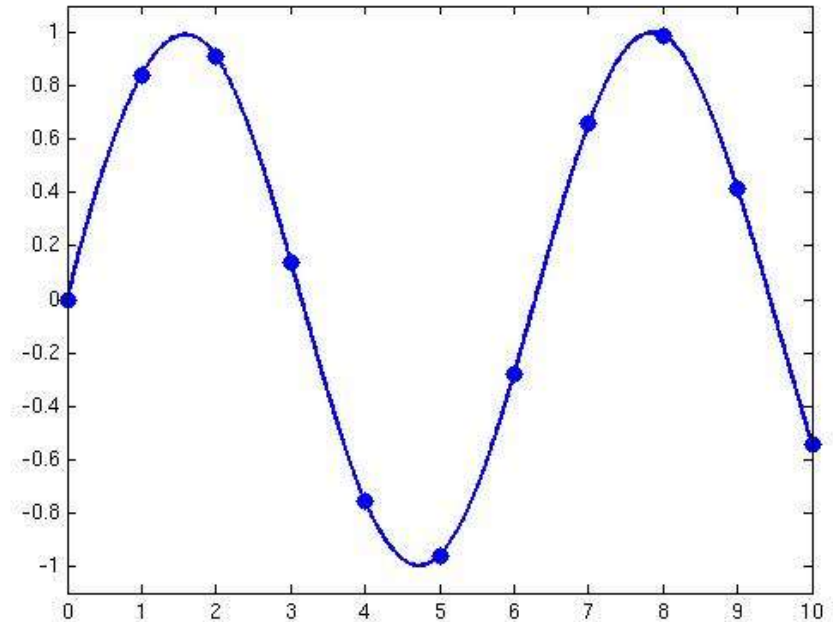
- Nearest Neighbor
- Linear
- Quadratic
- Spline



$$y(t) = y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i} \times (t - t_i) + \frac{y_{i+1} - 2y_i + y_{i-1}}{2\Delta t^2} \times (t - t_i)^2$$

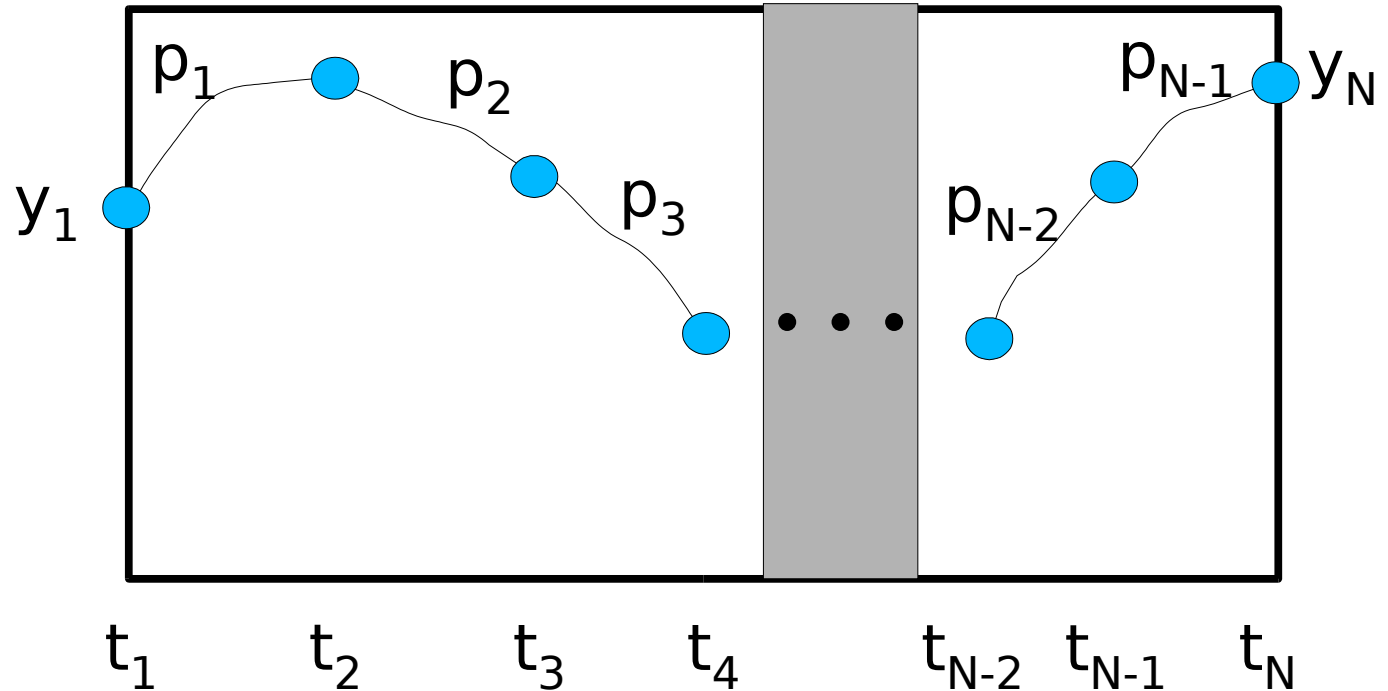
Interpolation

- Nearest Neighbor
- Linear
- Quadratic
- Spline
 - Cubic Function
 - Constraints to match first and second derivatives between segments



Constructing the Spline

N points: t_i, y_i
N-1 cubic
polynomials: p_i
require $4(N-1)$
coefficients



for cubic polynomials 1 through N-2:

$$p_i(t_{i+1}) = y_{i+1}$$

$$p_i(t_{i+1}) = p_{i+1}(t_{i+1})$$

$$p'_i(t_{i+1}) = p'_{i+1}(t_{i+1})$$

$$p''_i(t_{i+1}) = p''_{i+1}(t_{i+1})$$

function reproduces value

continuity condition

continuity of 1st derivative

continuity of 2nd derivative

Constructing the Spline

(continued)

Constraint equations give $4(N-2)$ equations to determine $4(N-1)$ unknown coefficients

Need 4 more constraints. Two are obvious:

$$p_1(t_1) = y_1$$

$$p_N(t_N) = y_N$$

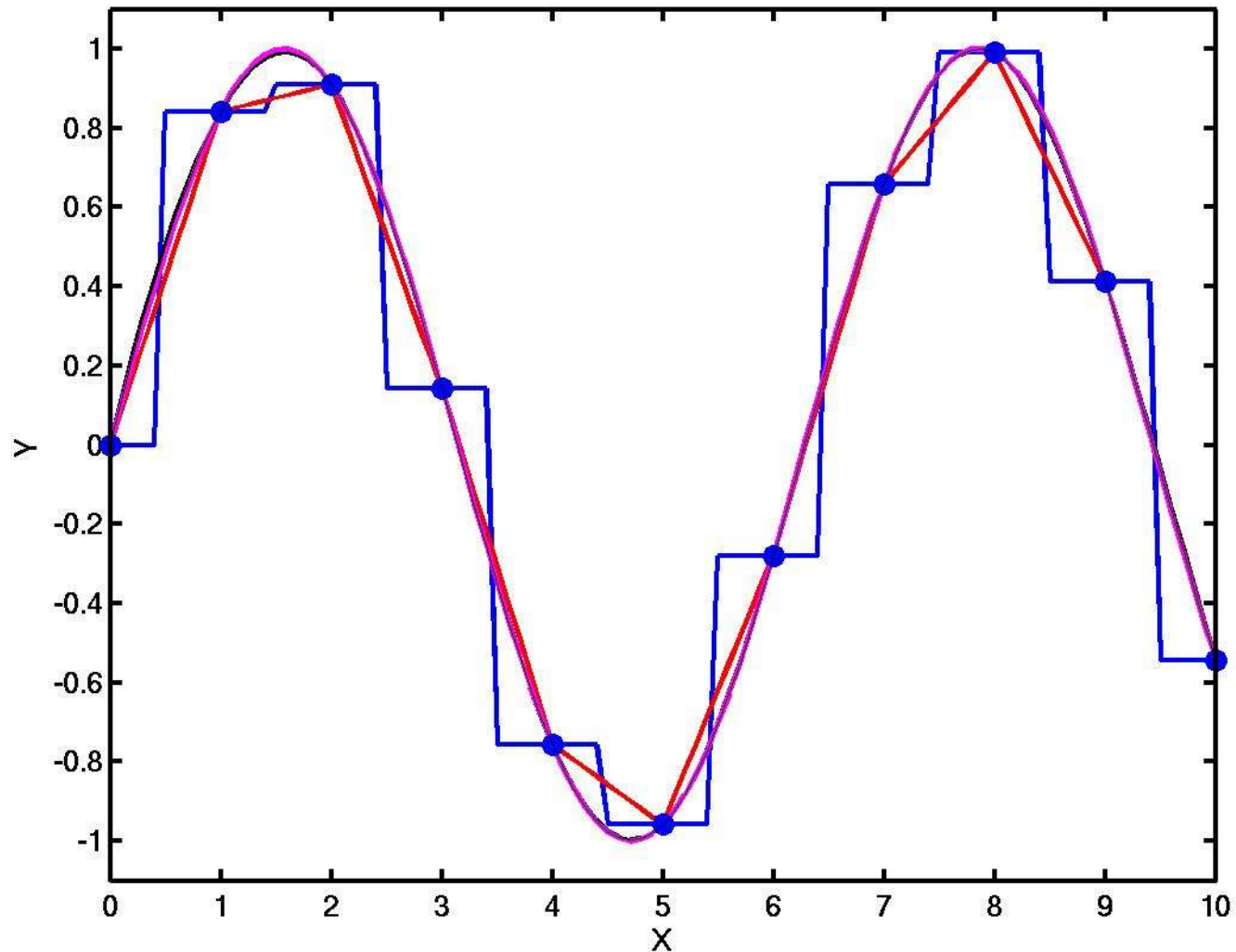
to these we add “Natural Spline” conditions of

$$p''_1(t_1) = 0$$

$$p''_N(t_N) = 0$$

Now have enough constraints to determine all polynomial segments p_i

Summary Example



LEGEND

**NEAREST
NEIGHBOR**

LINEAR

SPLINE

TRUE

MATLAB Interpolation Functions

- MATLAB provides special functions to interpolate arrays.
 - ***Interp1*** – allows nearest neighbor, linear, cubics, and splines.
 - ***Spline*** - cubic spline (also used in `interp1`)
- Syntax `yy = spline(t,y,tt)`
 - `t,y` are the `t` and `y` arrays
 - `tt` is a new array, with finer spacing, for interpolation
 - `yy` is result

Example

```
t = 0:10;  
y = sin(t);
```

```
tt = 0:0.01:10;
```

```
yy = spline(t,y,tt);
```

```
plot(t,y,'o')  
hold  
plot(tt,yy)
```

Define y and t arrays

**create array with
finer spacing**

**call the spline
function**

**plot results for
comparison**