

WEEK 12

***INTERPOLATION AND
CURVE FITTING***

DEFINING INTERPOLATION

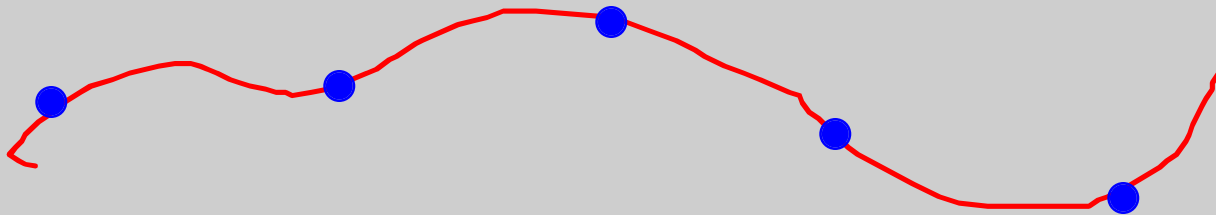
- Interpolation can be used in at least two ways
 - *Subsampling a dense signal and later filling it in*
 - *Generating new data where none was previously available*

SUBSAMPLING

- Say you have a large sound file taking too long to transmit or too much space to store.
- One solution is to subsample it, send(store) the sparse data then fill in the gaps later

EXAMPLE

- Keep the circles and discard the rest.
Later, fill in the gaps



VIDEO COMPRESSION

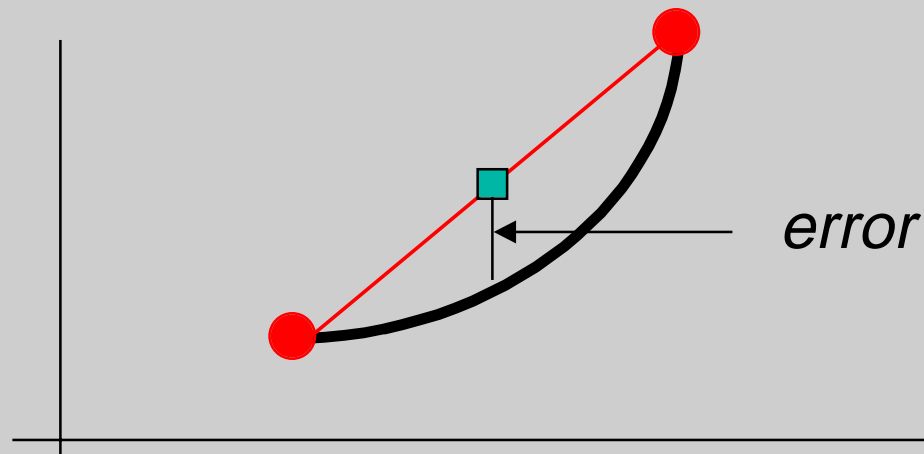
- Video captured at 30 frames/sec. contains a lot of redundancies
- Keep only a fraction of the frames and interpolate between them later
- This is implemented in the MPEG standard

INTERPOLATION METHODS

- There are 3 major interpolation techniques
 - *linear*
 - *cubic-spline(a 3rd degree polynomial)*
 - *polynomial fitting (polynomial of arbitrary order)*

LINEAR INTERPOLATION

- Simplest of its kind, works on the following principle



HOW DOES MATLAB DO IT?

- The main MATLAB's routine for 1-D interpolation is *interp1* with the following syntax
 - *yi=interp1(x,y,xi,'method')*
- (x,y) are the original coarse data. xi's are the new finer positions to be interpolated, yi is the answer

WORKING WITH interp1

- Want to interpolate a sinc function with samples originally located at [-4:1:4]

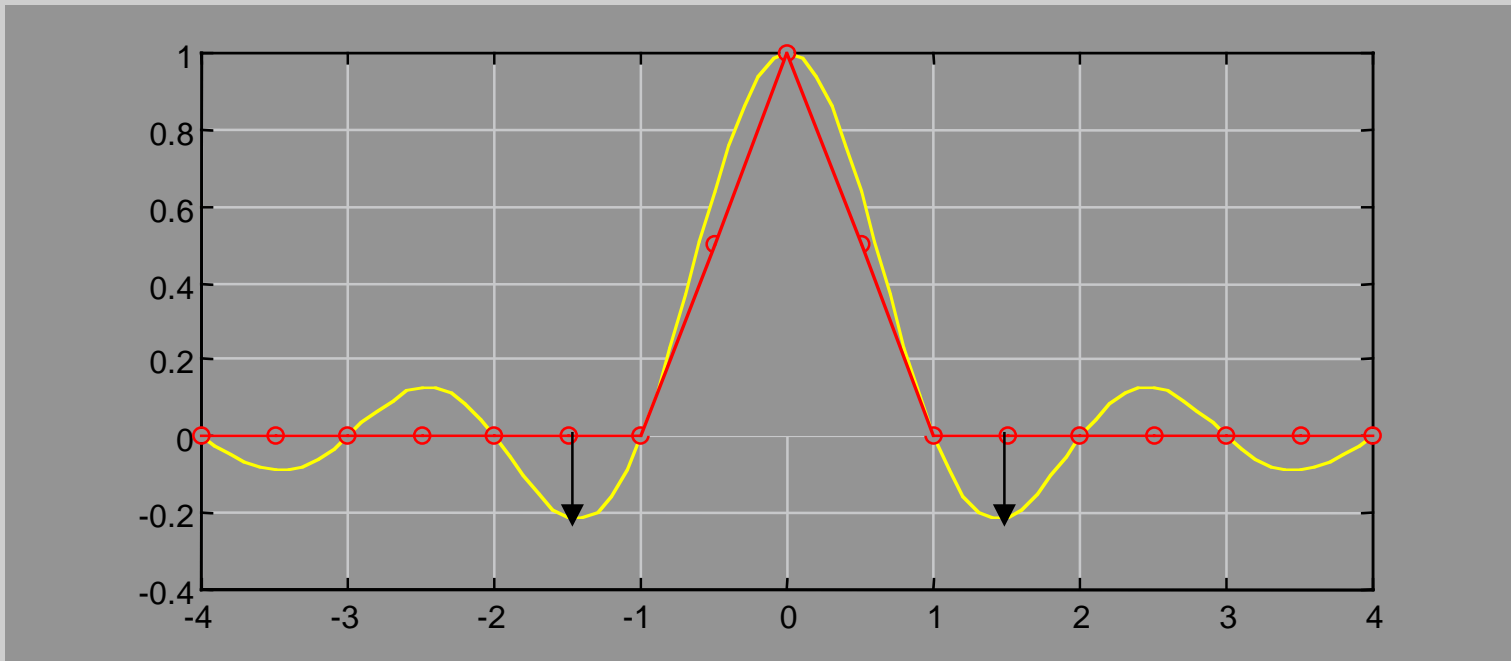


USING interp1

- `x=[-4:.1:4];%x-values`
- `y=[0 0 0 0 1 0 0 0 0];%coarsely sampled data at x=-4,-3,...`
- `xfiner=[-4:0.5:4];%interpolate at -4,-3.5,-3,...`
- `yfiner=interp1(x,y,xfiner,'linear');`**%interpolate at new grid positions**
- `plot(x,y,xfiner,yfiner,'ro',xfiner,yfiner,'r-');`

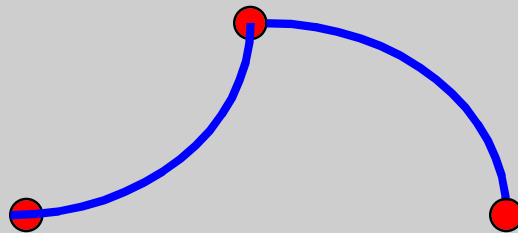
LINEAR INTERPOLATION OF SINC

- Interpolate at 0.5 intervals



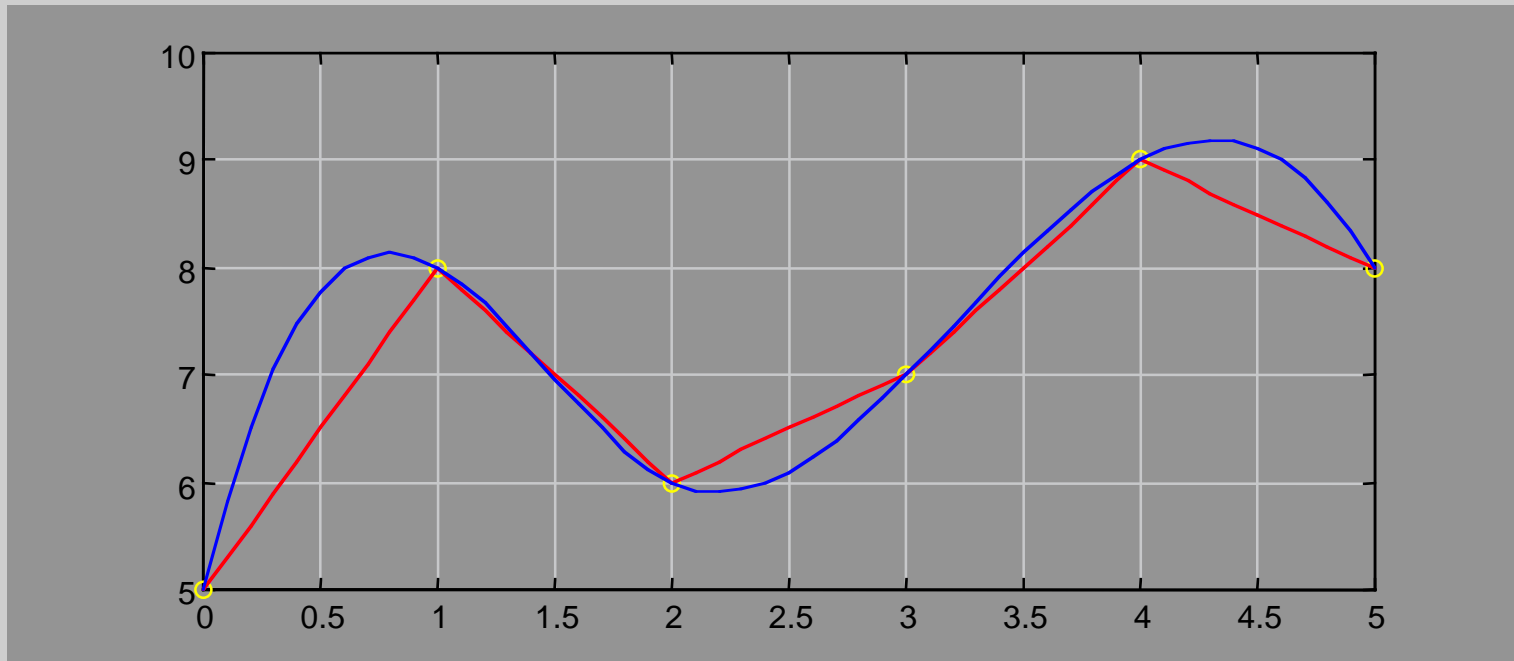
DEFINING CUBIC-SPLINE

- When data is interpolated by cubic spline, it means we pass a 3rd order polynomial through each pair of points



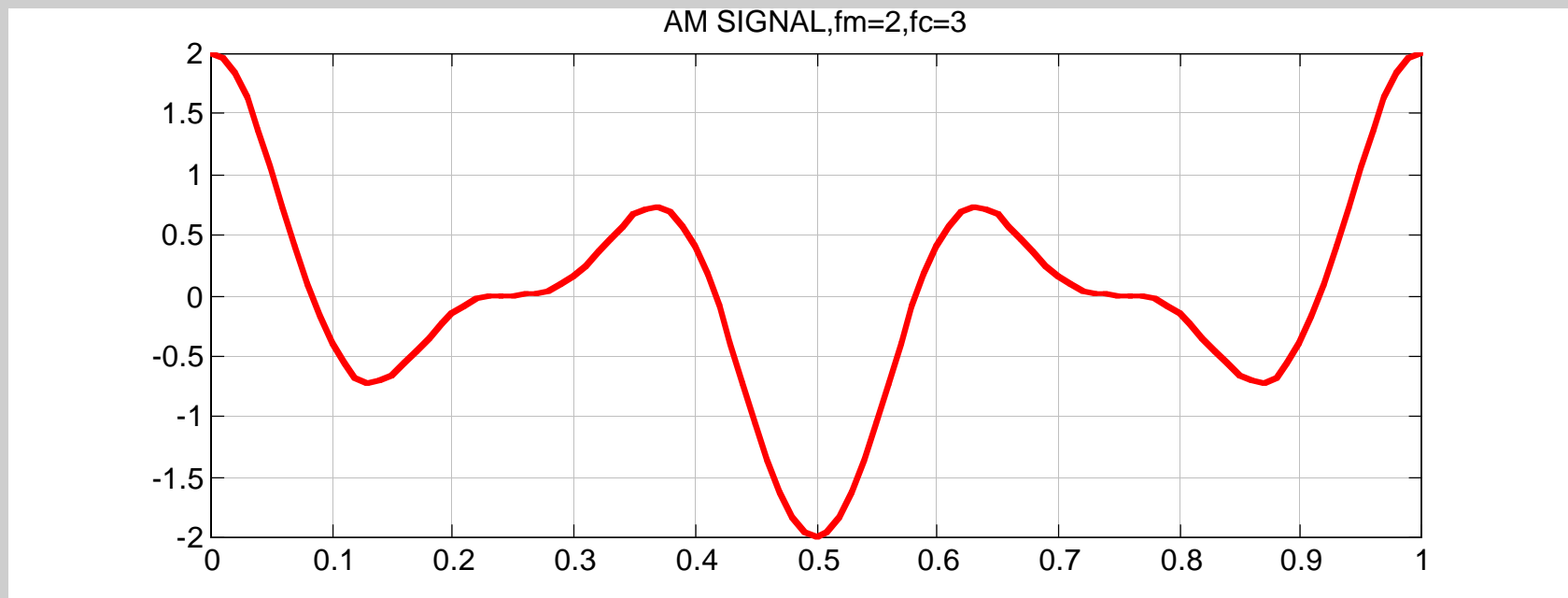
CUBIC-SPLINE

- A more accurate interpolation can be achieved by passing 3rd degree polynomials through coarse data



INTERPOLATING AN AM SIGNAL

- An amplitude modulated signal is
$$s(t) = (1 + \cos 2\pi f_m t) \cos(2\pi f_c t)$$

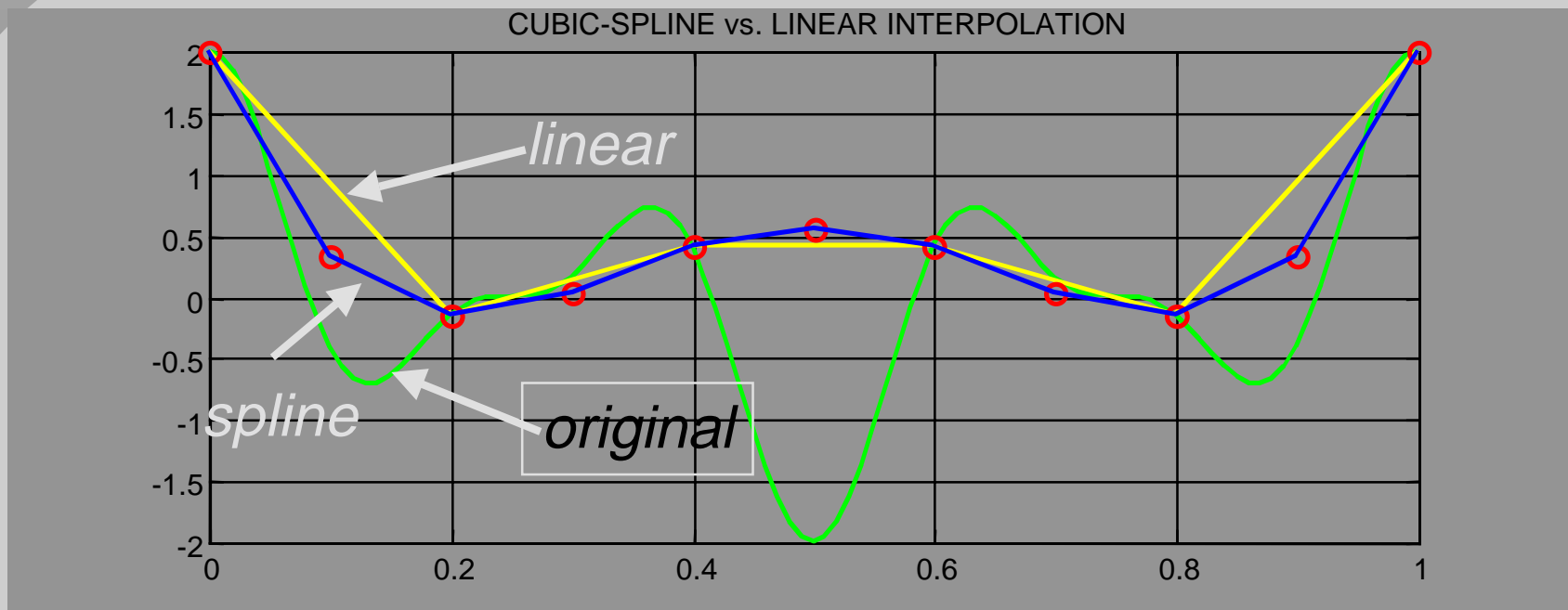


PRACTICE

- Evaluate an AM signal with $f_m=2, f_c=3$ in the range $0 < t < 1$ in increments of 0.01
- Subsample it 20:1.
- Using the kept data points, perform linear and cubic-spline interpolation

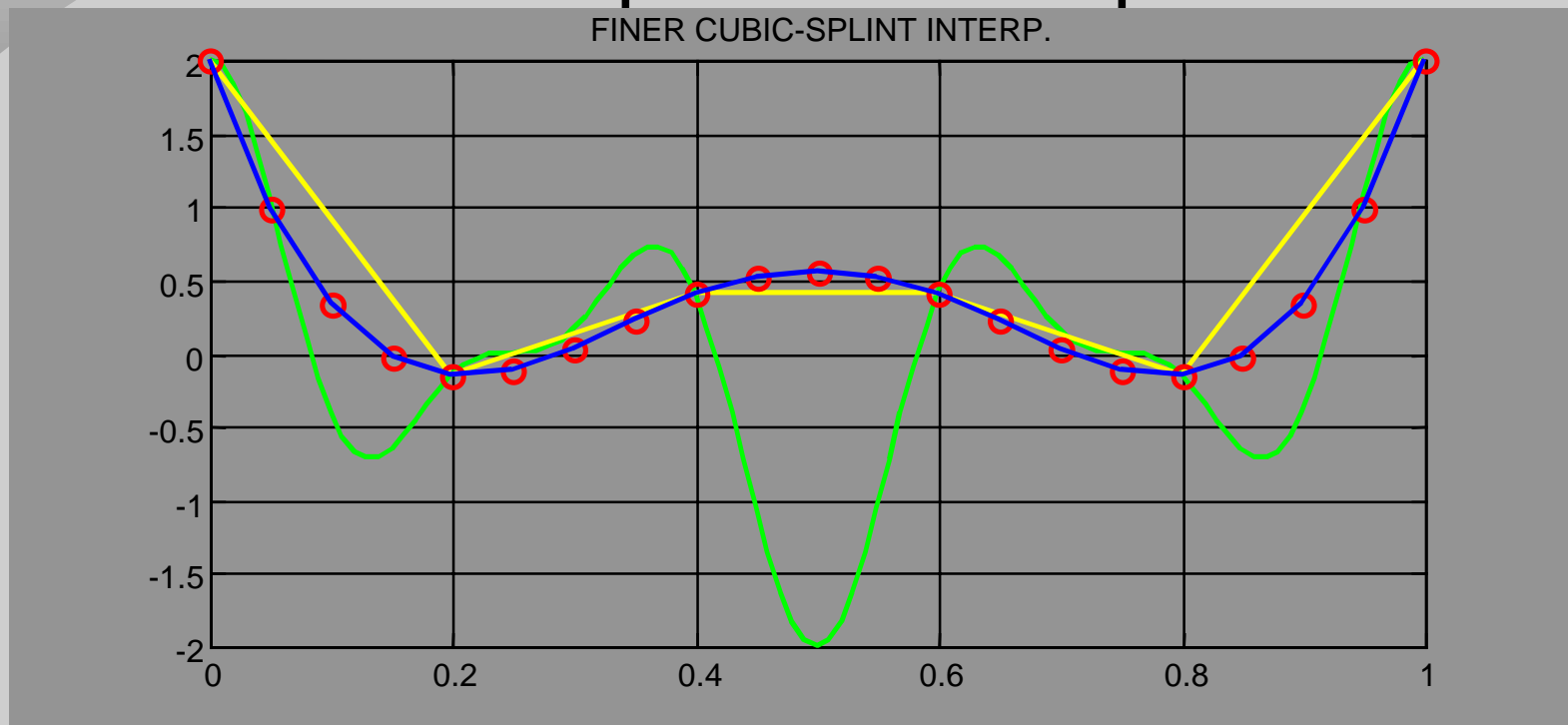
LINEAR VS. CUBIC

- spline gets closer to the actual curve



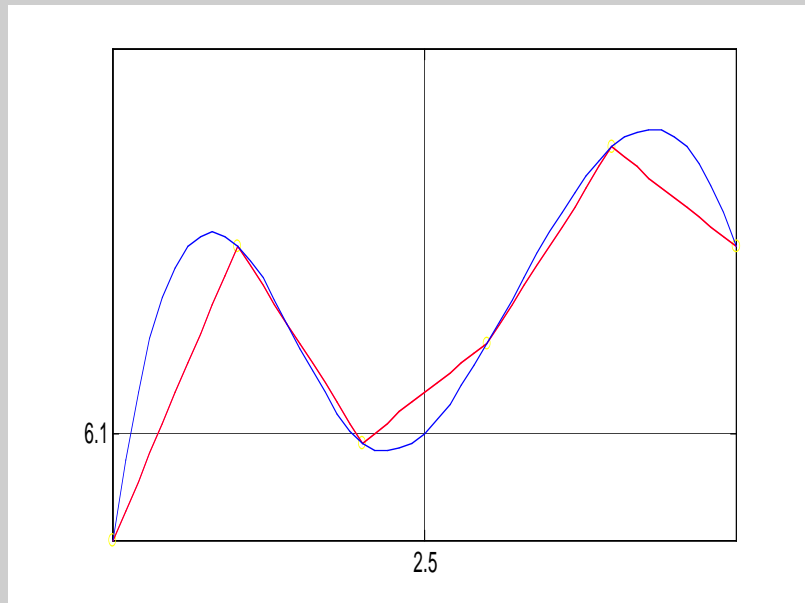
FINER INTERPOLATION

- Smoother interpolation via spline



DIRECT READ-OFF

- `interp1(x,y,2.5,'spline')` returns spline interpolated value of 6.1 at 2.5



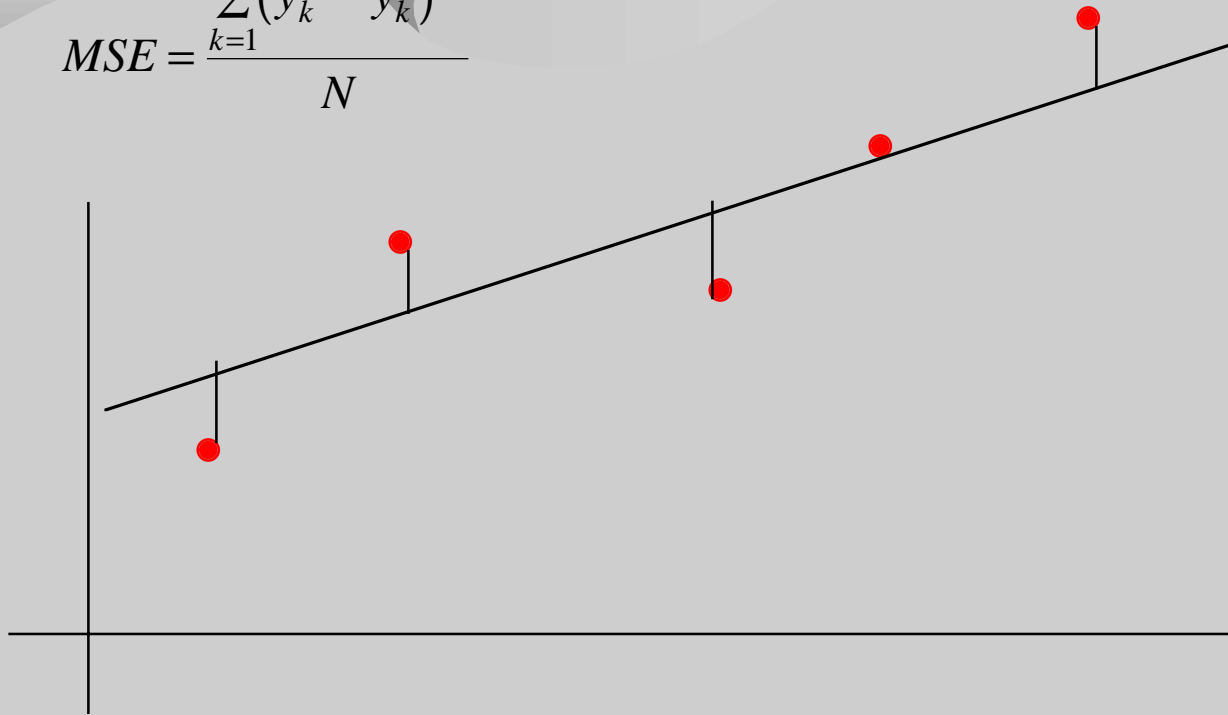
- `x=0:5;`
- `y=[5 8 6 7 9 8];`
- `xi=0:.1:5;`
- `ylin=interp1(x,y,xi);`
- `ys=interp1(x,y,xi,'spline');`
- `point=interp1(x,y,2.5,'spline')`

LEAST SQUARES CURVE FITTING

- Linear and cubic spline interpolations fit curves constrained to go through the coarse data points
- A curve fitted using least squares may not pass through any data point but it will be “close” to all of them in the “least squares” sense

LEAST SQUARES SENSE

$$MSE = \frac{\sum_{k=1}^N (y_k - \hat{y}_k)^2}{N}$$



LEAST SQUARES OBJECTIVE

- Find a function, e.g. a polynomial of whatever order, that minimizes the mean square error
- MATLAB does this through *polynomial regression*

DIFFERENCES WITH CUBIC SPLINE

- Both are polynomials but cubic splines are 3rd order polynomials.
- The big difference is that cubic spline fits separate 3rd degree polynomials per segment
- Least squares fits a single polynomial through all the data points

DEFINING A POLYNOMIAL

- An Nth degree polynomial is specified by N+1 coefficients
- If there are N+1 data points, an Nth degree polynomial will pass through all of them

$$f(x) = a_0x^N + a_1x^{N-1} + \dots + a_{N-1}x + a_N$$

polyfit FUNCTION

- To fit an nth degree polynomial to (x,y) data use
 - $p = \text{polyfit}(x,y,n)$
- It returns a vector p of n+1 coeff. in decreasing powers of x.
- So if $f(x) = a_0x^N + a_1x^{N-1} + \dots + a_{N-1}x + a_N$
then $p = [a_0, a_1, a_2, a_3, \dots]$

Evaluating and Plotting the Fitted Polynomial: *polyval*

- Once polynomial is specified through the vector p , it can be evaluated directly using

– $y = \text{polyval}(p, x)$



coarse x values
vector of polynomial coefficients

The diagram consists of two black arrows pointing upwards. The left arrow points from the text 'vector of polynomial coefficients' to the parameter 'p' in the function call 'polyval(p, x)'. The right arrow points from the text 'coarse x values' to the parameter 'x' in the same function call.

Example using polyfit

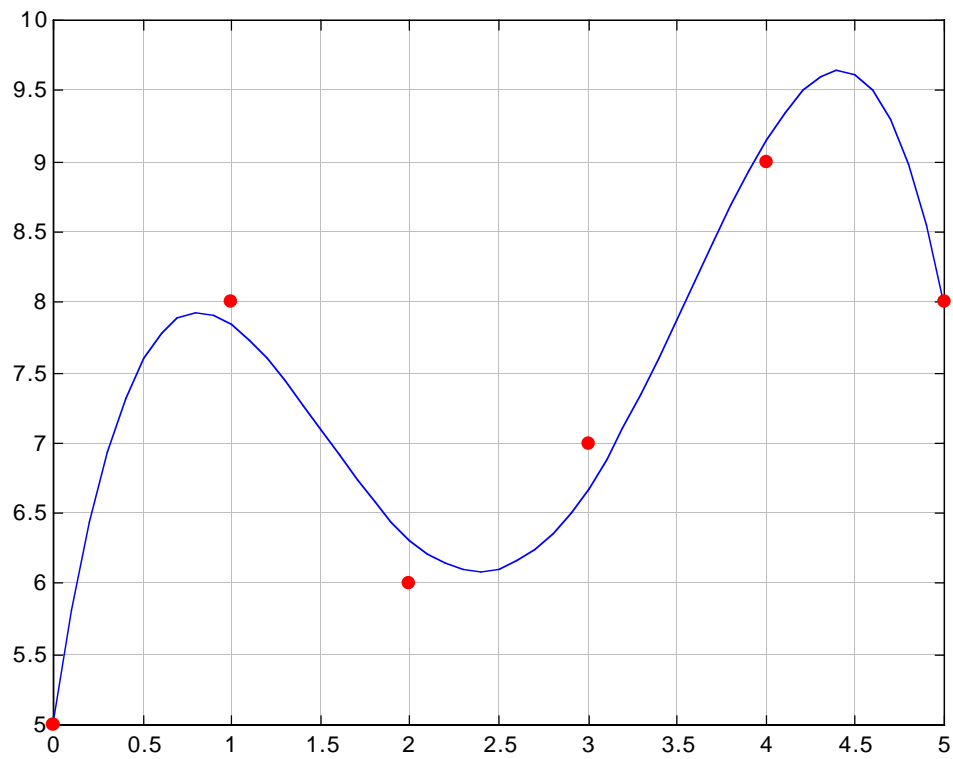
- Let $x=[0\ 1\ 2\ 3\ 4\ 5]$ and $y=[5\ 8\ 6\ 7\ 9\ 8]$ be the coarse data points. This is how to fit a 4th order polynomial

`p=polyfit(x,y,4);%p is the coeff. vector`

now let's evaluate the polynomial at finer positions given by `xfine=[0:.1:5]`

`yfine=polyval(p,xfine);%plot yfine to see`

RESULT

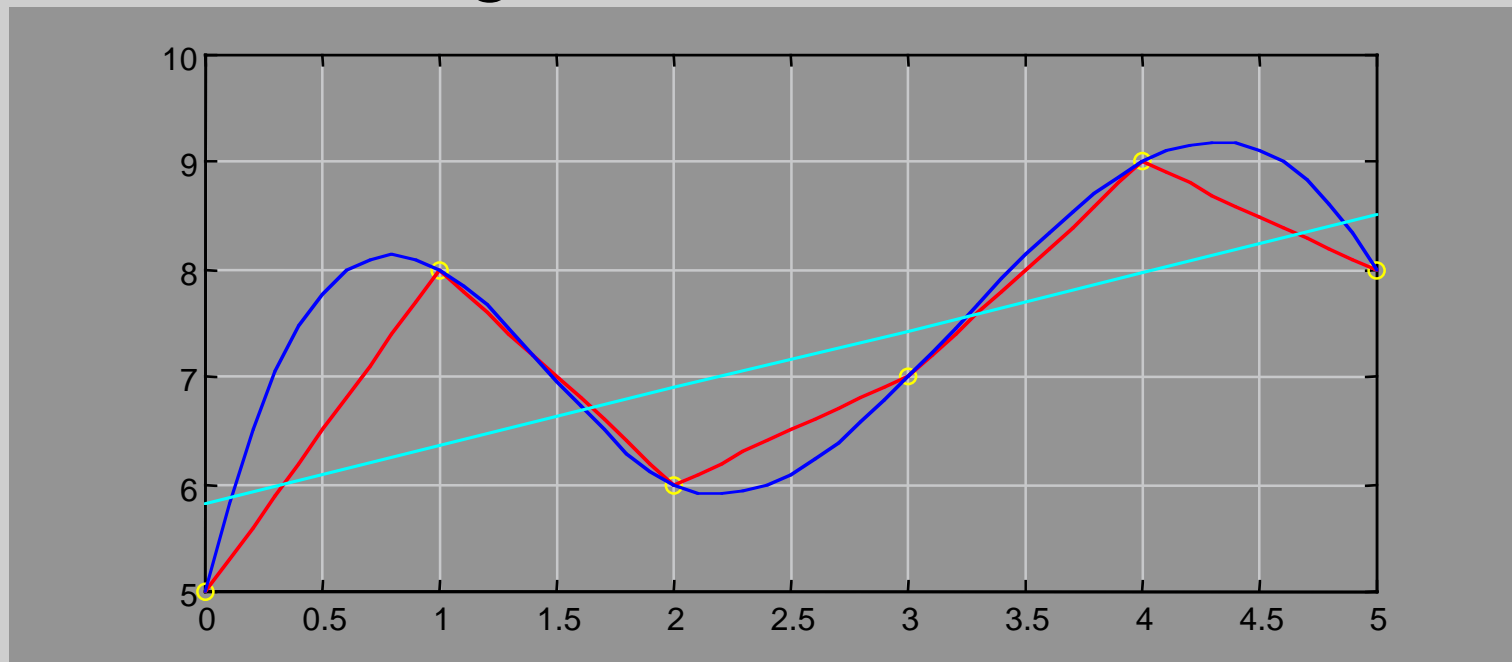


Comparing Interpolations

- In the following slides we start with the following data points
 - $x=[0 \ 1 \ 2 \ 3 \ 4 \ 5];$
 - $y=[5 \ 8 \ 6 \ 7 \ 9 \ 8];$
- We will then interpolate along x in increments of 0.1 using progressively larger order polynomials

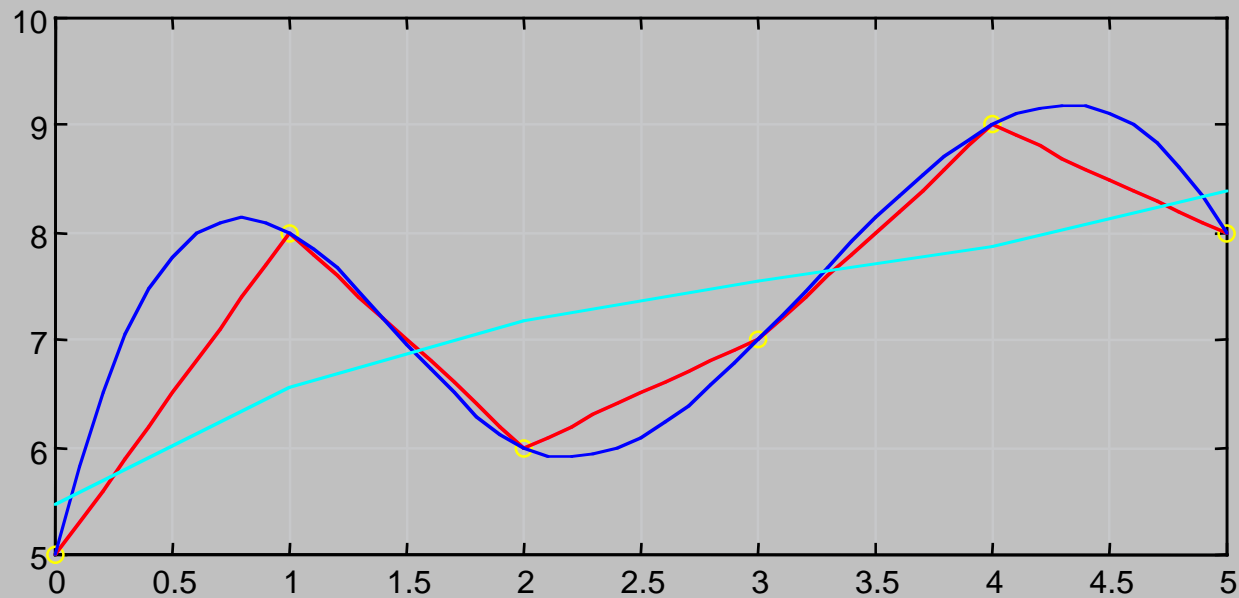
APPLYING *polyfit*

- Let's fit a first degree polynomial to data. We'll get $m=0.54, h=5.8$



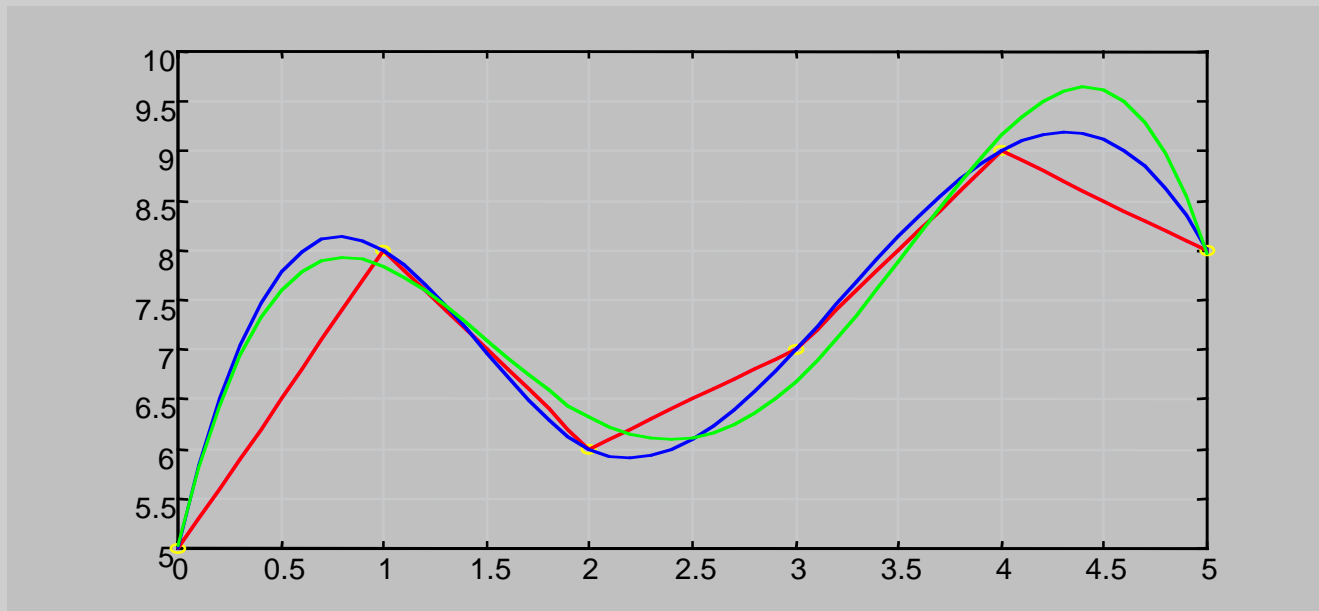
3RD DEGREE POLYNOMIAL

- Coefficients are [0.037 -0.349 1.407 5.460]



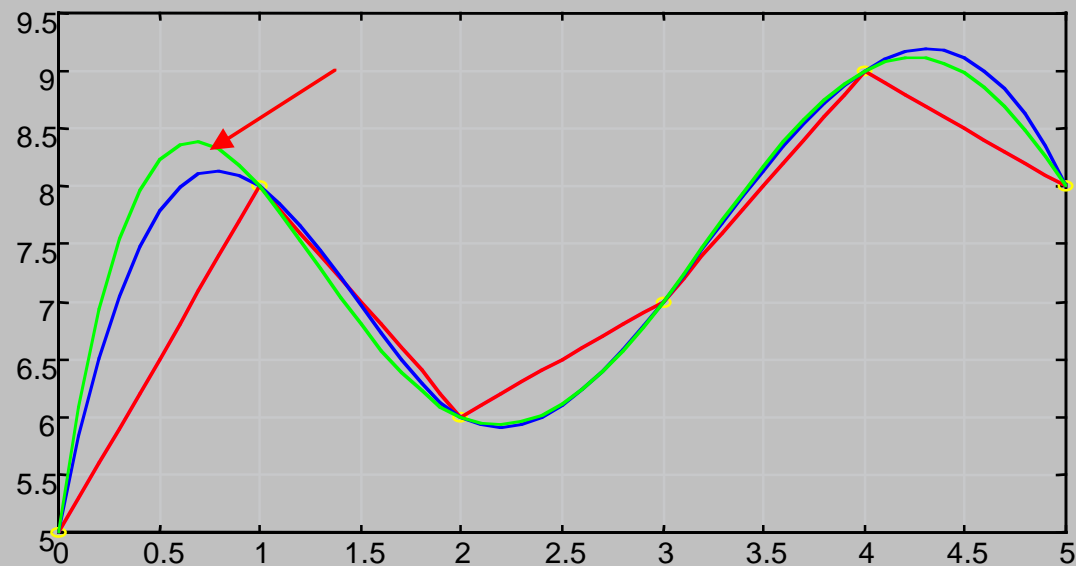
4th DEGREE POLYNOMIAL

■ $\text{coeff} = [-0.2500 \quad 2.5370 \quad -8.0278 \quad 8.5503 \quad 5.0317]$



5th DEGREE POLYNOMIAL

- Expect the polynomial to pass through all the points



In-Class Exercise

- The best way to learn interpolation is to duplicate previous slides. Use data given in slide #28. Then use the following interpolations to fill in the blanks
 - *-linear*
 - *cubic-spline*
 - *polynomials of orders 1 thru 5*

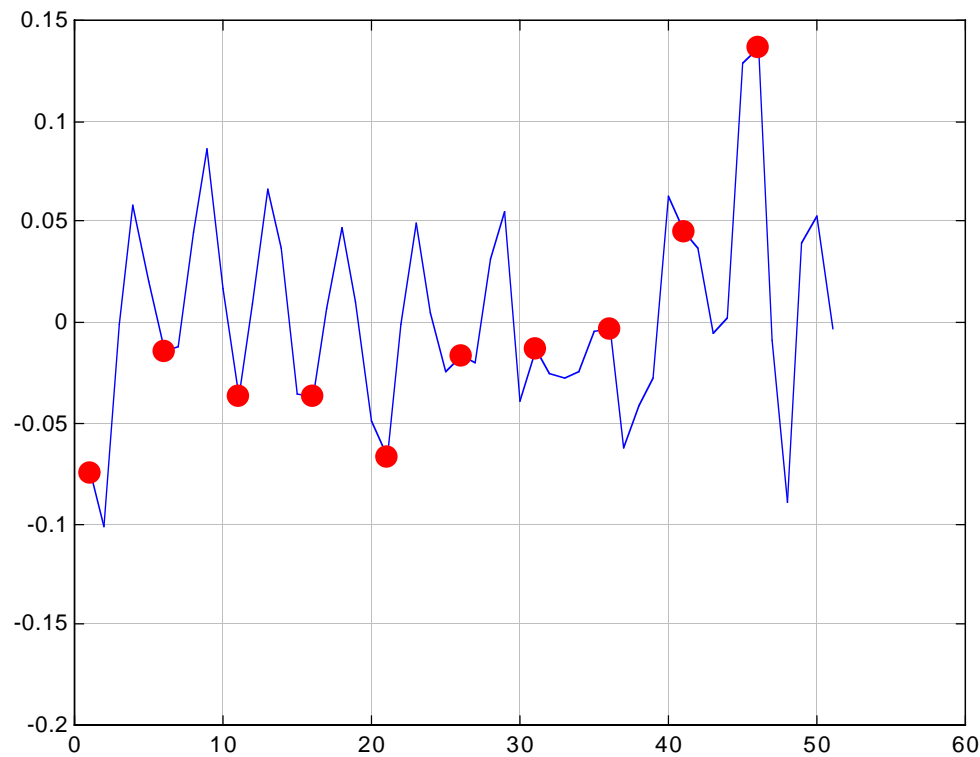
Homework

INTERPOLATING AUDIO

Due: before Thanksgiving break

- For next week's homework do this
 - *load 50 samples of Bond starting at 3000*
 - *subsample it 5:1*
 - *fill in the missing 5 points by linear interpolation*
 - *do the same using cubic spline*
 - *do the same using polynomial fit of degree 5*
 - *superimpose all on the same graph using appropriate symbols*

A graphical statement of the problem



INTERPOLATING AUDIO - optional

- Listen to the original bond to establish a reference
- Subsample by 5 and listen to it
- Now fill-in the missing points by interpolation and then play the segment back. What to watch for?. Is the interpolated audio as good as the original?