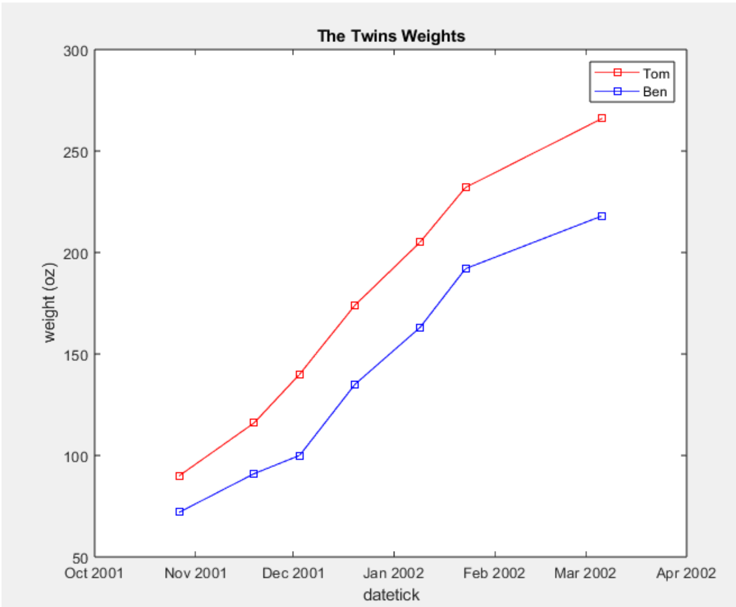
Amelia Rotondo
CWID: 887925113

24

Homework: Chapter 3

3

2. You can use `datenum` to convert the date in the first three columns to a serial date number measuring time in days. Then, plot the twin's weights.



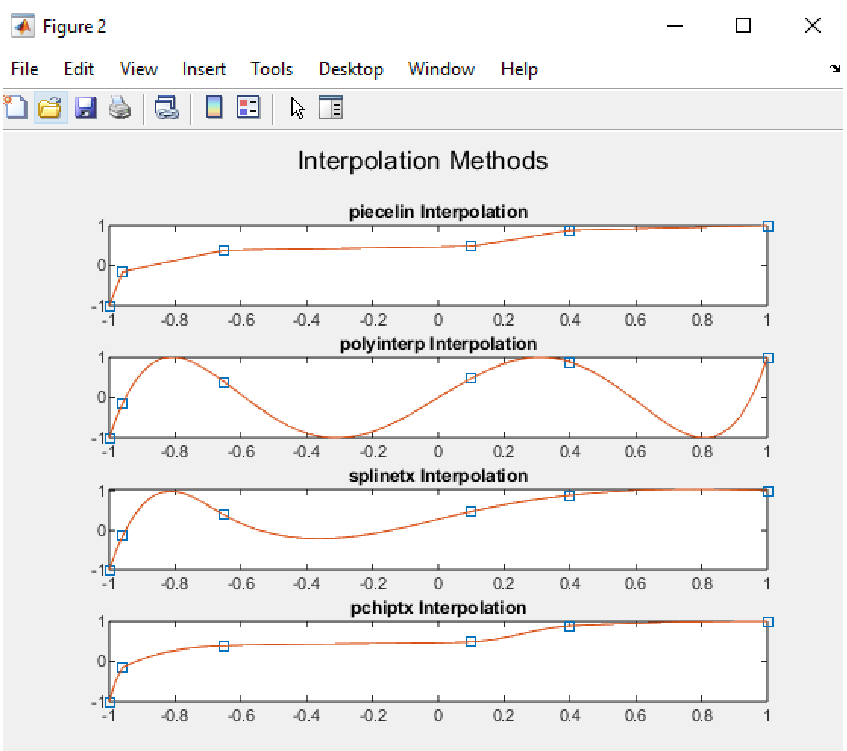
include your code

piecewise linear?

4

3. Interpolation:

a) Interpolate these data by each of the four interpolants discussed in this chapter: `piecelin`, `polyinterp`, `splinetx`, and `pchiptx`. Plot the results for $-1 \leq x \leq 1$.



Amelia Rotondo
CWID: 887925113

Homework: Chapter 3

b) What are the values of each of the four interpolants at x = − 0.3? Which of these values do you prefer? Why?
   - pchiptx : 0.4314
   - piecelin : 0.4283
   - polyinterp : -0.9999
   - splinetx : -0.2036

   Considering the actual positions of the points on the graph- I prefer the splinetx solution above all others- since its use of control parameters and high degree make it appear to best-suit the path described by the points.

Use polyfit(x,y,5)

c) The data were actually generated from a low-degree polynomial with integer coefficients. What is that polynomial?

   This appears to be the function used shown in the polyinterp method above. As much as I'd love to calculate the coefficients, I am doing this assignment way closer to the deadline than I'd have liked to- so I won't. I will give you an encouraging "trust me I can solve a vandermode matrix using matlab I am just in a hurry".

5

7. Prove that the interpolating polynomial is unique. That is, if P (x) and Q(x) are two polynomials with degree less than n that agree at n distinct points, then they agree at all points.

If two polynomials, P(x) and Q(x), both of degree less than n, agree at n distinct points, it implies they have at least n+1 common roots. However, a polynomial of degree less than n can have at most n distinct roots by the Fundamental Theorem of Algebra. Therefore, the only way for P(x) and Q(x) to agree at n distinct points is for them to be identical polynomials. In other words, if two polynomials agree at n distinct points, they must agree at all points, demonstrating the uniqueness of the interpolating polynomial.

ok

Amelia Rotondo
CWID: 887925113

Homework: Chapter 3

5

8. Give convincing arguments that each of the following descriptions defines the same polynomial, the Chebyshev polynomial of degree five, T5(x).

(Script to Prove 8.a through 8.f)

```
% Define the golden ratio
phi = (1 + sqrt(5)) / 2;
% Define the evaluation point
x_val = 0.5;
% Calculate T5(x) using the power form representation
T5_power = 16*x_val^5 - 20*x_val^3 + 5*x_val;
% Calculate T5(x) using the trigonometric form representation
T5_trig = cos(5*acos(x_val));
% Calculate T5(x) using the Horner representation
coeffs = [16, 0, -20, 0, 5, 0];
T5_horner = polyval(coeffs, x_val);
% Calculate T5(x) using the Lagrange form representation
x_nodes = [1, phi/2, (phi - 1)/2, -1, -phi/2, -(phi - 1)/2];
y_nodes = [1, -1, 1, -1, 1, -1];
T5_lagrange = 0;
for k = 1:length(x_nodes)
Lk = 1;
for j = 1:length(x_nodes)
if j ~= k
Lk = Lk * (x_val - x_nodes(j)) / (x_nodes(k) - x_nodes(j));
end
end
T5_lagrange = T5_lagrange + Lk * y_nodes(k);
end
% Calculate T5(x) using the factored representation
z_nodes = [sqrt((2 + phi)/4), sqrt((3 - phi)/4), 0, -sqrt((3 - phi)/4),
-sqrt((2 + phi)/4)];
T5_factored = 16 * prod(x_val - z_nodes);
% Calculate T5(x) using the three-term recurrence representation
T5_recurrence = zeros(1, 6);
T5_recurrence(1) = 1;
T5_recurrence(2) = x_val;
for n = 2:5
T5_recurrence(n + 1) = 2 * x_val * T5_recurrence(n) - T5_recurrence(n -
1);
end
```

Amelia Rotondo
CWID: 887925113

Homework: Chapter 3

(Screenshot of Script to Prove 8.a through 8.f)

```
% Display the results
disp(['T5(x) (Power Form) at x = 0.5: ' num2str(T5_power)]);
disp(['T5(x) (Trigonometric Form) at x = 0.5: ' num2str(T5_trig)]);
disp(['T5(x) (Horner Form) at x = 0.5: ' num2str(T5_horner)]);
disp(['T5(x) (Lagrange Form) at x = 0.5: ' num2str(T5_lagrange)]);
disp(['T5(x) (Factored Representation) at x = 0.5: ' num2str(T5_factored)]);
disp(['T5(x) (Three-Term Recurrence) at x = 0.5: ' num2str(T5_recurrence(6))]);
```

Command Window

```
>> Chapter3
T5(x) (Power Form) at x = 0.5: 0.5
T5(x) (Trigonometric Form) at x = 0.5: 0.5
T5(x) (Horner Form) at x = 0.5: 0.5
T5(x) (Lagrange Form) at x = 0.5: 0.5
T5(x) (Factored Representation) at x = 0.5: 0.5
T5(x) (Three-Term Recurrence) at x = 0.5: 0.5
```

4

9. The M-file rungeinterp.m provides an experiment with a famous polynomial interpolation problem due to Carl Runge. Let f (x) = 1/(1 + 25x^2) ,and let Pn(x) denote the polynomial of degree n−1 that interpolates f (x) at n equally spaced points on the interval $-1 \leq x \leq 1$. Runge asked whether, as n increases, Pn(x) converges to f (x) . The answer is yes for some x, but no for others.

a) For what x does Pn(x) → f (x) as n → ∞ ?
   i) Pn(x) approaches f(x) ~~from~~ near x = 0 as n approaches infinity.

b) Change the distribution of the interpolation points so that they are not equally spaced. How does this affect convergence? Can you find a distribution so that Pn(x) → f (x) for all x in the interval?
   i)

```
% Customize the distribution of interpolation points.
% Here, we use Chebyshev nodes instead of equally spaced nodes.
if n == 1
x = 0;
else
% Generate Chebyshev nodes
x = cos((2*(0:n-1) + 1)*pi / (2*n));
end
```

The distribution affects convergence by changing how accurate the interpolated line is to the actual line on each side. It appears to oscillate between being more and less accurate depending on n's specific value.

Amelia Rotondo
CWID: 887925113

Homework: Chapter 3

3

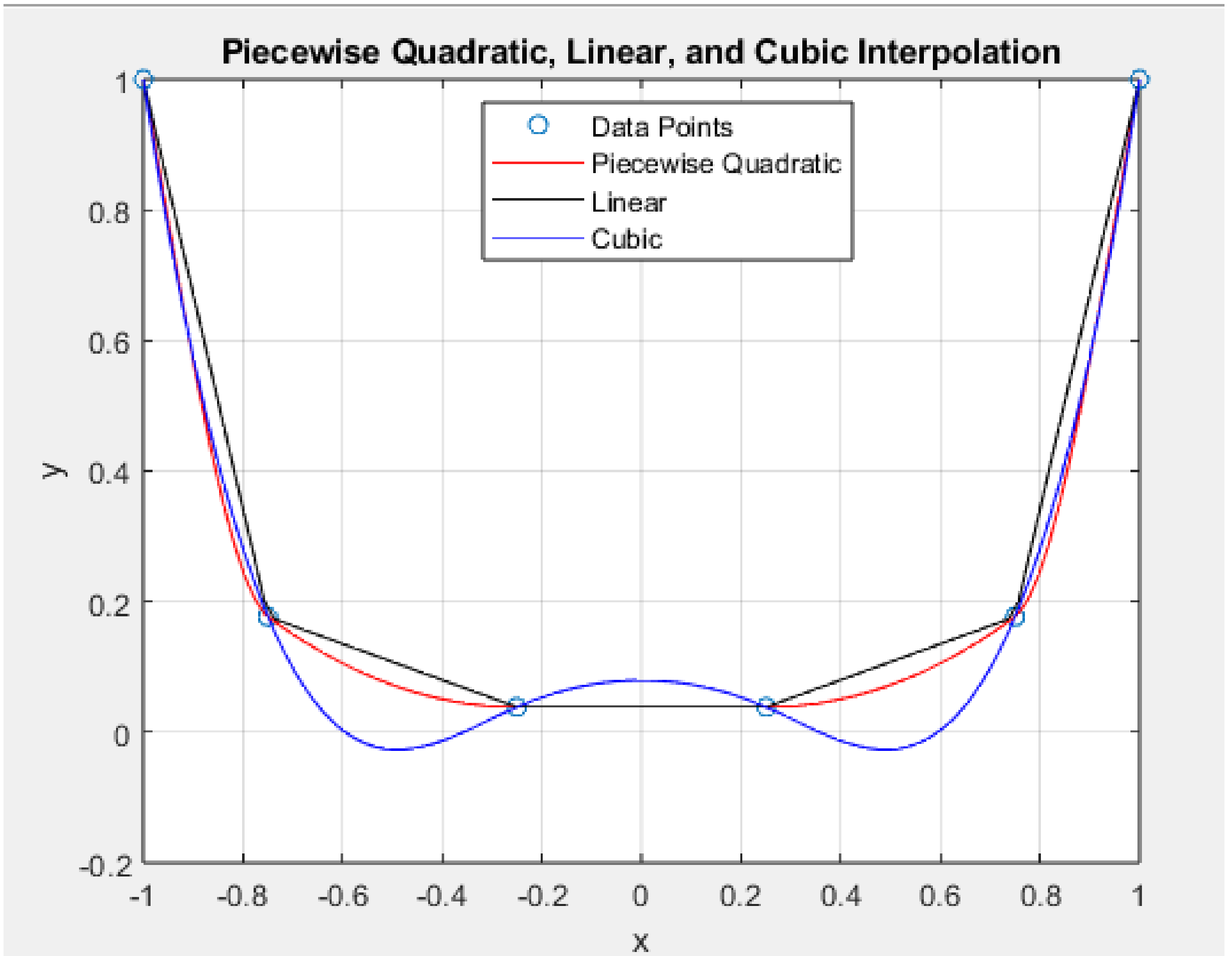
10. We skipped from piecewise linear to piecewise cubic interpolation. How far can you get with the development of piecewise quadratic interpolation?

```
% Problem 10:
% Given data points
x = [-1.00; -0.75; -0.25; 0.25; 0.75; 1.00];
y = [1.0000; 0.1768; 0.0385; 0.0385; 0.1768; 1.0000];
% Domain for interpolation
u = linspace(-1, 1, 100);
% Interpolation using piecewise quadratic interpolation
interp_quad = interp1(x, y, u, 'pchip');
% Interpolation using linear interpolation
interp_linear = interp1(x, y, u, 'linear');
% Interpolation using cubic interpolation
interp_cubic = interp1(x, y, u, 'spline');
% Plot the results
figure;
plot(x, y, 'o', u, interp_quad, 'r', u, interp_linear, 'k', u, interp_cubic,
'b');
title('Piecewise Quadratic, Linear, and Cubic Interpolation');
xlabel('x');
ylabel('y');
legend('Data Points', 'Piecewise Quadratic', 'Linear', 'Cubic');
grid on;
```

pchip is piecewise cubic

Amelia Rotondo
CWID: 887925113

Homework: Chapter 3



As shown in the image above, it's understandable that piecewise quadratics would best fit point mapped by a quadratic function, but they appear to flow more jagged in contrast to cubics.