UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING
Department of Electrical & Computer Engineering

ECE 204 *Numerical methods*

Approximating values using interpolating polynomials

Douglas Wilhelm Harder, LEL, M.Math.
dwharder@uwaterloo.ca
dwharder@gmail.com

CC BY NC SA

Introduction

- In this topic, we will
  - Discuss how to design interpolating polynomials
  - Observe how we can minimize the condition number of the system we must solve
  - See how we can optimize the interpolating polynomials so as to make them work best with Horner's rule
  - Find interpolating polynomials that estimate the values in the middle of two, three and four points
  - Find interpolating polynomials that estimate the values close to the most recent reading of two, three and four points

Review

- Recall that most data comes from periodically sampled sensors and other devices
  - We will assume that our values are equally sampled in either space or time
  - Other techniques, such as finite-element methods, are beyond the scope of this course
- Thus, we will assume that:
  - Our values are either $x_k = x_0 + kh$ or $t_k = t_0 + k\Delta t$
  - We will assume the values are reasonably exact, represented by $f(x_k)$ and $y(t_k)$
  - We will always design our polynomials to be optimally written for Horner's rule
  - Later, we will see techniques if there is significant error in the readings

Linear interpolation

- Suppose we have two points:

$$(x_{k-1}, f(x_{k-1})) \text{ and } (x_k, f(x_k))$$

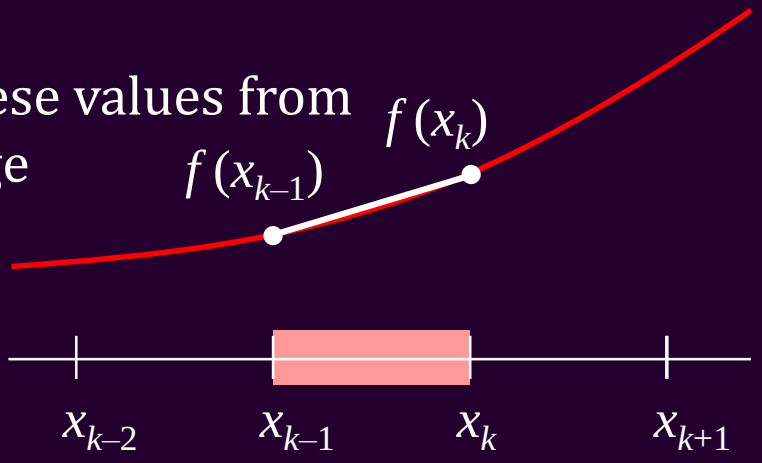
  - Alternatively,

$$(x_k - h, f(x_k - h)) \text{ and } (x_k, f(x_k))$$

- The interpolating polynomial is

$$\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} x + \frac{f(x_{k-1}) x_k - f(x_k) x_{k-1}}{x_k - x_{k-1}}$$

- Issues
  - Nothing prevents these values from being arbitrarily large
  - Numerous subtractions

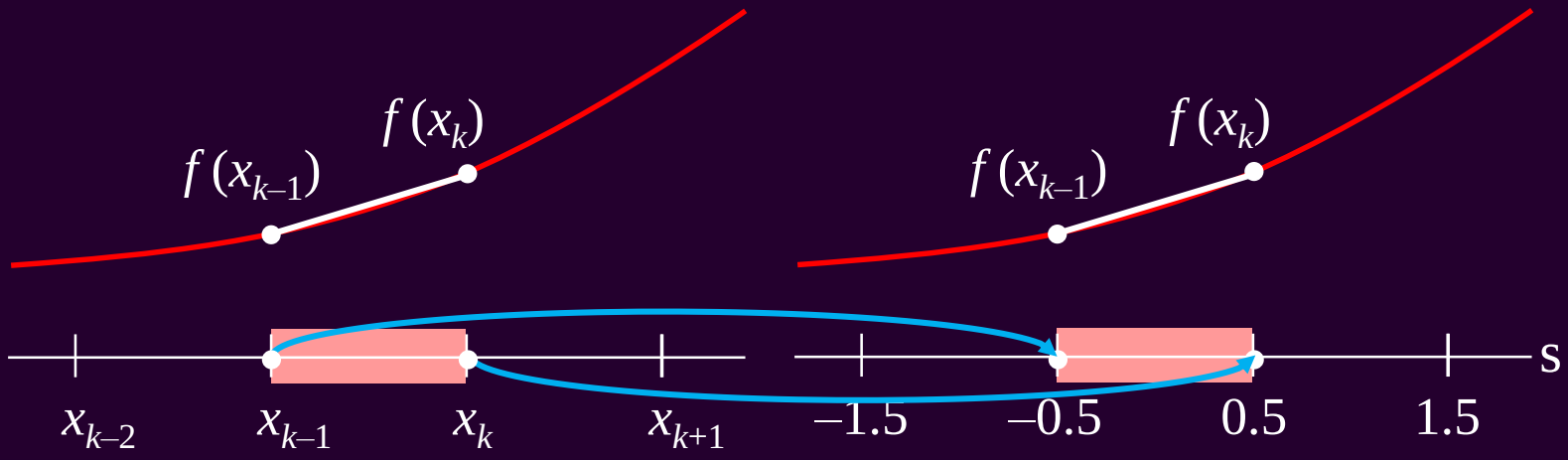


Linear interpolation

- Solution:
  - Let us shift and scale the interval $[x_{k-1}, x_k]$ to $[-0.5, 0.5]$
  - Now, we must solve

$$\begin{pmatrix} x_{k-1} & 1 \\ x_k & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_0 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix}$$

  - Solving this yields $\dfrac{1}{x_k - x_{k-1}} x - \dfrac{1}{2} \dfrac{x_{k-1} + x_k}{x_k - x_{k-1}} = \dfrac{1}{h} x - \dfrac{x_{k-1} + x_k}{2h}$



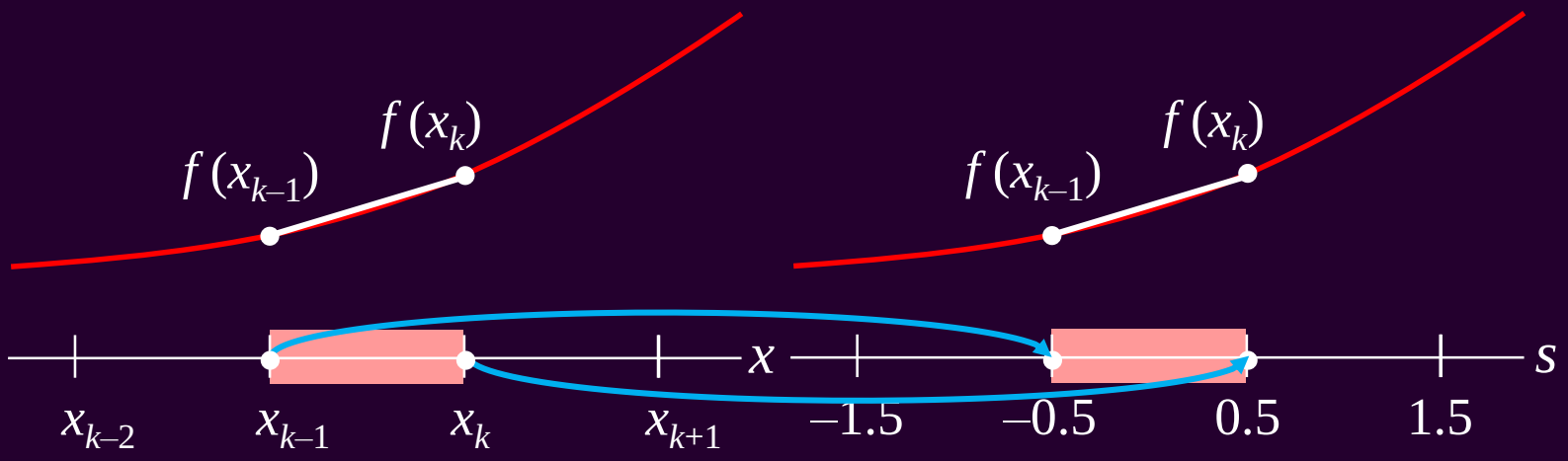
Linear interpolation

- Alternatively, think of this as:
  - Mapping the mid-point to zero (shifting)
  - Dividing by the period (scaling)

mid-point

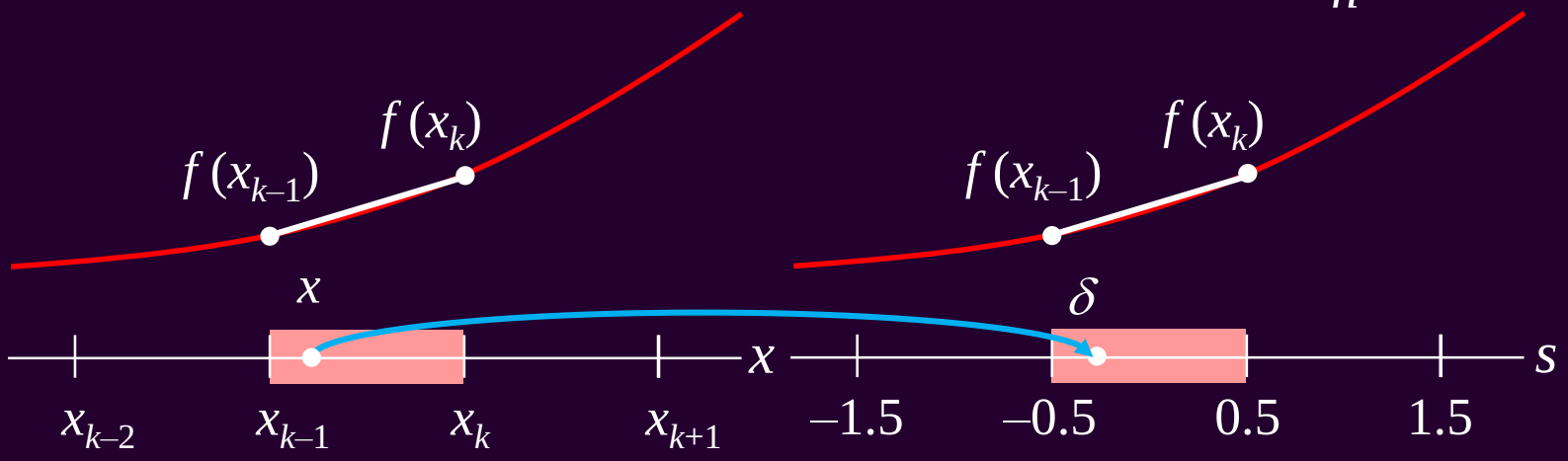
$$s \leftarrow \frac{x - \frac{x_{k-1} + x_k}{2}}{h}$$

sampling period



Linear interpolation

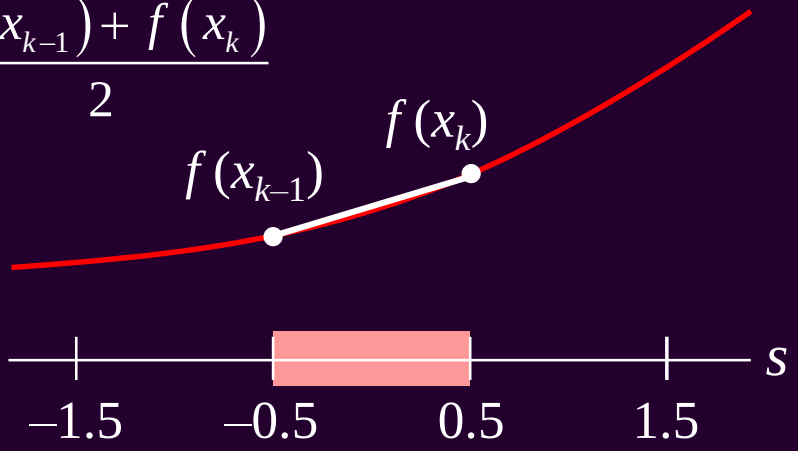
- Solution:
  - Now, given $s \leftarrow \dfrac{x - \dfrac{x_{k-1} + x_k}{2}}{h}$, note that:
    - x_{k-1} is mapped to –0.5
    - The mid-point is mapped to 0
    - x_k is mapped to 0.5
  - Also, $x = \dfrac{x_k + x_{k-1}}{2} + \delta h$ is mapped to $s \leftarrow \dfrac{\dfrac{x_k + x_{k-1}}{2} + \delta h - \dfrac{x_{k-1} + x_k}{2}}{h} = \delta$



Linear interpolation

- Solution:
  - Now we simply interpolate $a_1 s + a_0$ though
    $$(-0.5, f(x_{k-1})) \text{ and } (0.5, f(x_k))$$
  - Now, we must solve
    $$\begin{pmatrix} -0.5 & 1 \\ 0.5 & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_0 \end{pmatrix} = \begin{pmatrix} f(x_{k-1}) \\ f(x_k) \end{pmatrix}$$
  - Solving this yields:

$$\left[f(x_k) - f(x_{k-1}) \right] s + \frac{f(x_{k-1}) + f(x_k)}{2}$$



Linear interpolation

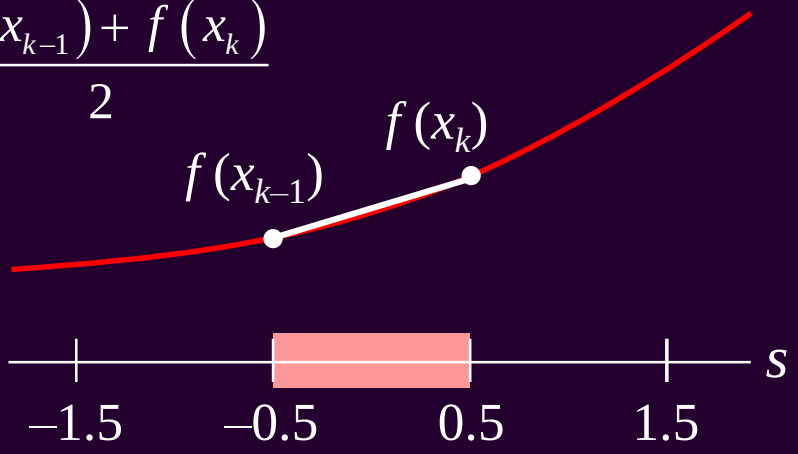
- Thus, to estimate the value of at $x = \frac{x_{k-1} + x_k}{2} + \delta h$

  we will evaluate this new expression at $s = \delta$

  – This requires your design to always think in terms of proportions relative to *h*

  – Note that $\text{cond}\begin{pmatrix} -0.5 & 1 \\ 0.5 & 1 \end{pmatrix} = 2$

$$\left[f(x_k) - f(x_{k-1}) \right] s + \frac{f(x_{k-1}) + f(x_k)}{2}$$



Linear interpolation

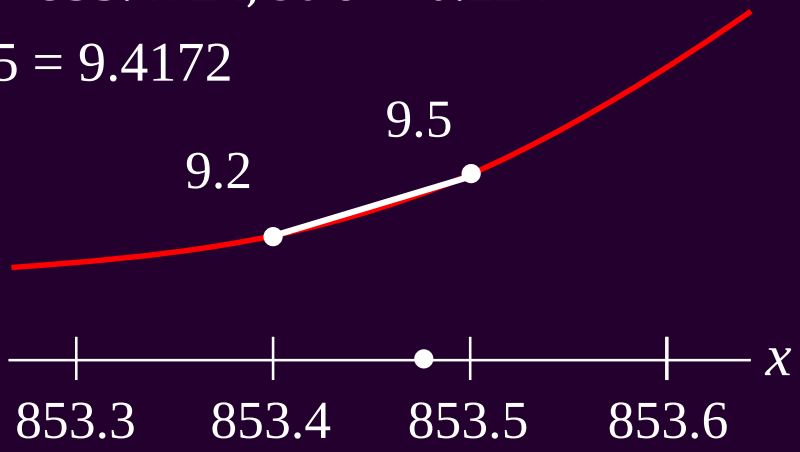
- For example, suppose that we have the following data:

  (853.4, 9.2) and (853.5, 9.5)

  - Thus:
    - The midpoint is 853.45 and the sampling period is $h = 0.1$, so

$$s \leftarrow \frac{x - 853.45}{0.1}$$

    - The polynomial (–0.5, 9.2) and (0.5, 9.5) is $0.3s + 9.35$
  - To approximate at $x = 853.4724$, so $s = 0.224$
  - Thus, $0.3 \cdot 0.224 + 9.35 = 9.4172$

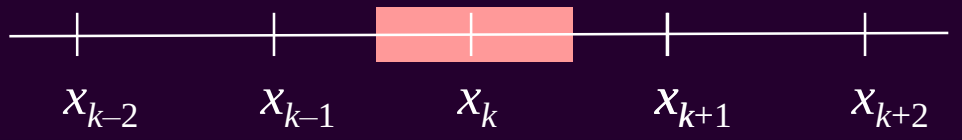


Quadratic interpolation

- Next, suppose we want to approximate values of f around x_k
  - Let us find an interpolating quadratic passing through the three points

$$(x_{k-1}, f(x_{k-1})),\ (x_k, f(x_k)) \text{ and } (x_{k+1}, f(x_{k+1}))$$

$$\left(\frac{(x_{k+1}-x_k)f(x_{k-1})+(x_k-x_{k-1})f(x_{k+1})+f(x_k)(-x_{k+1}+x_{k-1})}{(x_k-x_{k-1})(-x_{k+1}+x_{k-1})(-x_{k+1}+x_k)}\right)x^2$$
$$+\left(\frac{(x_k^2-x_{k+1}^2)f(x_{k-1})+(-x_k^2+x_{k-1}^2)f(x_{k+1})+(-x_{k-1}^2+x_{k+1}^2)f(x_k)}{(x_k-x_{k-1})(-x_{k+1}+x_{k-1})(-x_{k+1}+x_k)}\right)x$$
$$+\left(\frac{1}{(x_k-x_{k-1})(-x_{k+1}+x_{k-1})(-x_{k+1}+x_k)}(-x_kx_{k+1}(-x_{k+1}+x_k)f(x_{k-1})\right.$$
$$\left.+x_{k-1}(x_k(x_k-x_{k-1})f(x_{k+1})+f(x_k)x_{k+1}(-x_{k+1}+x_{k-1})))\right)$$



Centered quadratic interpolation

- Instead, use the previous approach:
  - Let us find an interpolating quadratic passing through the three points

$$(-1, f(x_{k-1})),\ (0, f(x_k)) \text{ and } (1, f(x_{k+1}))$$

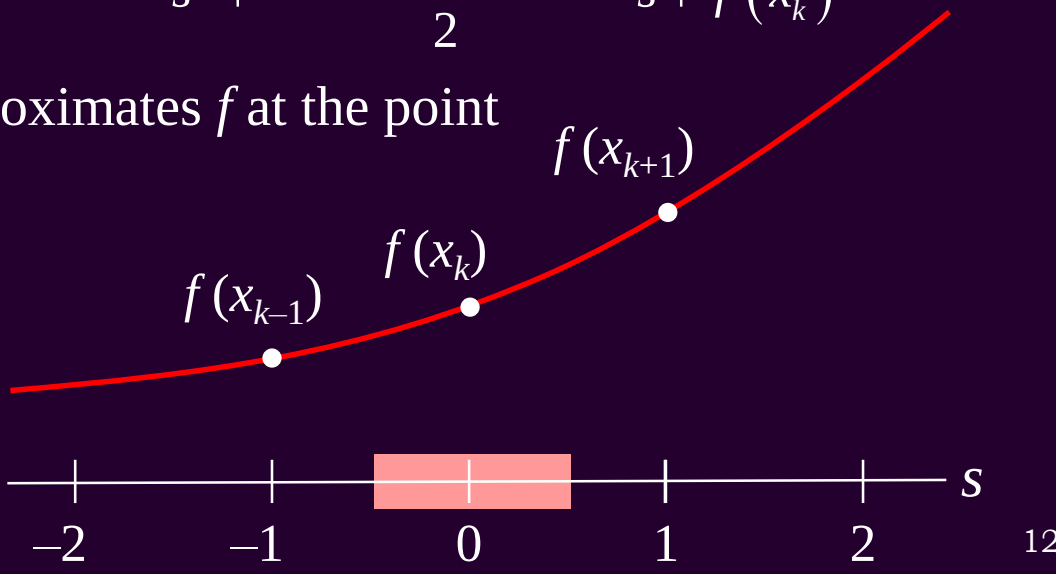
  - The interpolating polynomial is

$$\frac{f(x_{k-1}) - 2f(x_k) + f(x_{k+1})}{2} s^2 + \frac{f(x_{k+1}) - f(x_{k-1})}{2} s + f(x_k)$$

  - As before, it approximates f at the point

$$x_k + \delta h$$

$$\begin{pmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_2 \\ a_1 \\ a_0 \end{pmatrix} = \begin{pmatrix} f(x_{k-1}) \\ f(x_k) \\ f(x_{k+1}) \end{pmatrix}$$

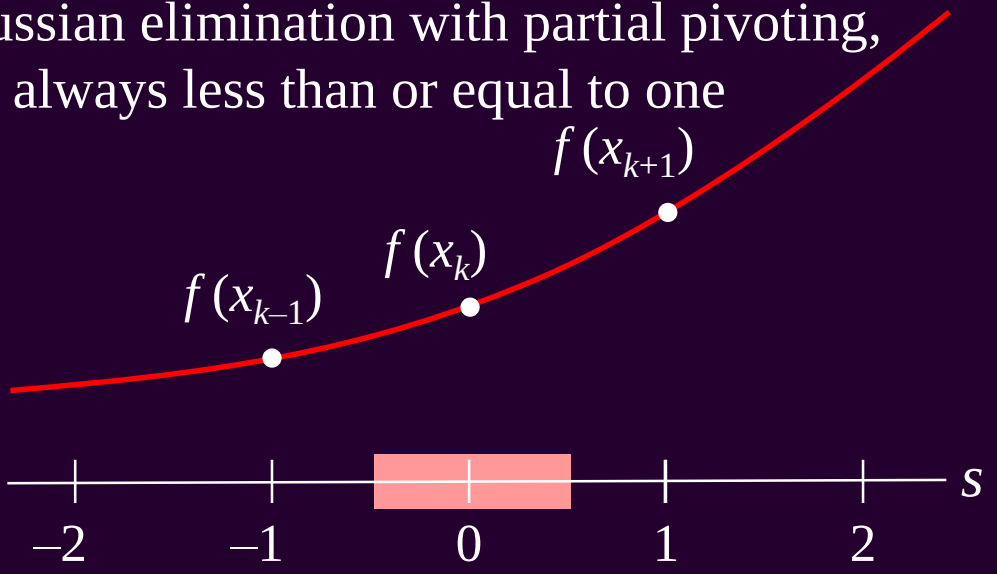


Centered quadratic interpolation

- Why is this beneficial?
  - First, we are guaranteed that $|\delta| \leq 0.5$
  - Thus, this is ideal for Horner's rule as well as avoiding situations like subtractive cancellation or loss of precision

$$\left(\frac{f(x_{k-1}) - 2f(x_k) + f(x_{k+1})}{2} \delta + \frac{f(x_{k+1}) - f(x_{k-1})}{2} \right) \delta + f(x_k)$$

  - Just like with Gaussian elimination with partial pivoting, the multiplier is always less than or equal to one

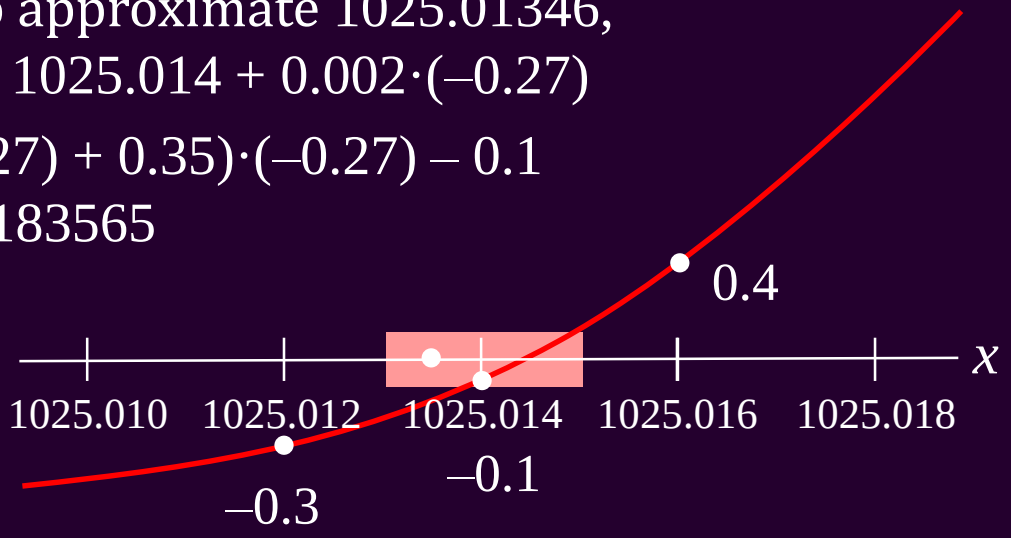


Centered quadratic interpolation

- For example, suppose that we have the following data:

  (1025.012, –0.3), (1025.014, –0.1) and (1025.016, 0.4)

  – Thus:
    - The midpoint is 1025.014
    - The step size is $h = 0.002$
    - The polynomial is $0.15\delta^2 + 0.35\delta - 0.1$
  – If you wanted to approximate 1025.01346, we note this is $1025.014 + 0.002 \cdot (-0.27)$
  – Thus, $(0.15 \cdot (-0.27) + 0.35) \cdot (-0.27) - 0.1$
    $= -0.183565$



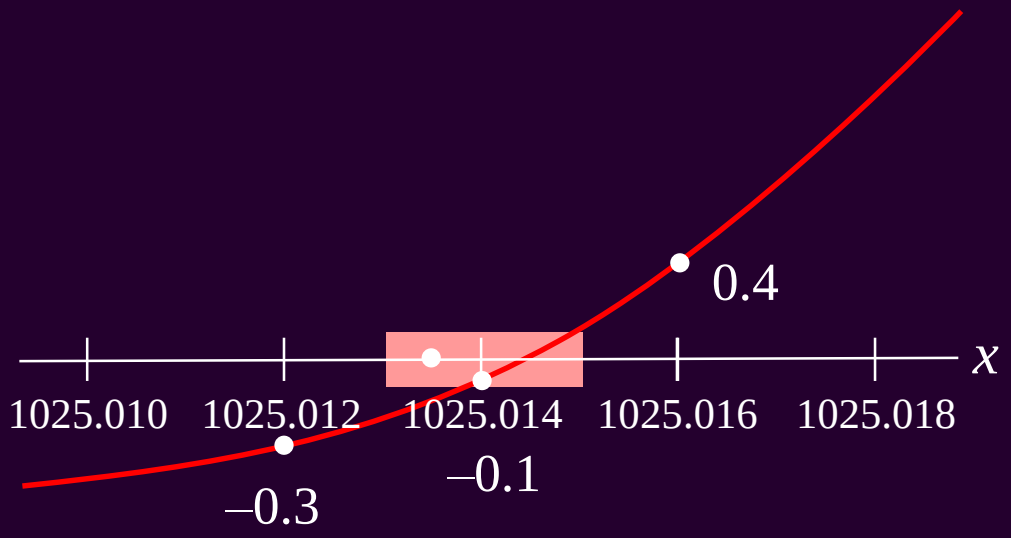
Centered quadratic interpolation

- For example, suppose that we have the following data:

  (1025.012, –0.3), (1025.014, –0.1) and (1025.016, 0.4)

  – Incidentally, if you were to find the interpolating polynomial without shifting and scaling, we'd have:

  $$37501.36x^2 - 76878662.68x + 39400763084.63$$

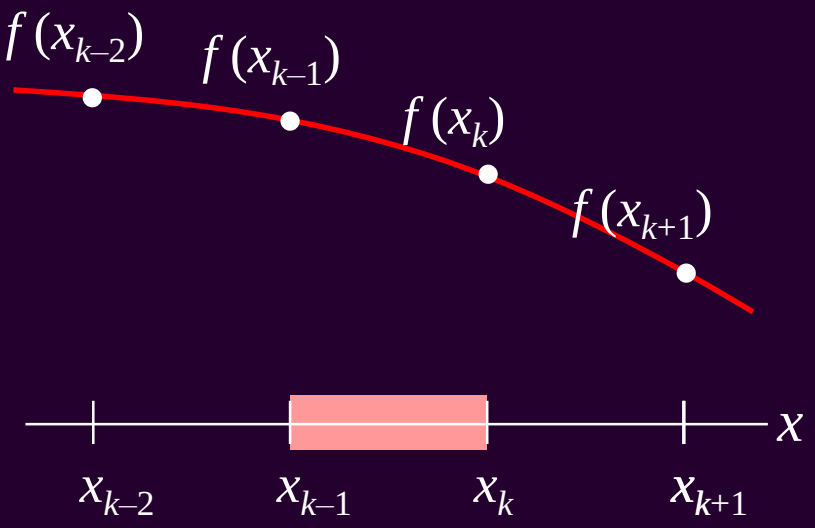


Centered cubic interpolation

- Finally, suppose we have four points:

  $(x_{k-2}, f(x_{k-2}))$, $(x_{k-1}, f(x_{k-1}))$, $(x_k, f(x_k))$ and $(x_{k+1}, f(x_{k+1}))$

  – The interpolating polynomial is too large to display



Centered cubic interpolation

- Again, we shift and scale the

  $(-1.5, f(x_{k-2}))$, $(-0.5, f(x_{k-1}))$, $(0.5, f(x_k))$ and $(1.5, f(x_{k+1}))$

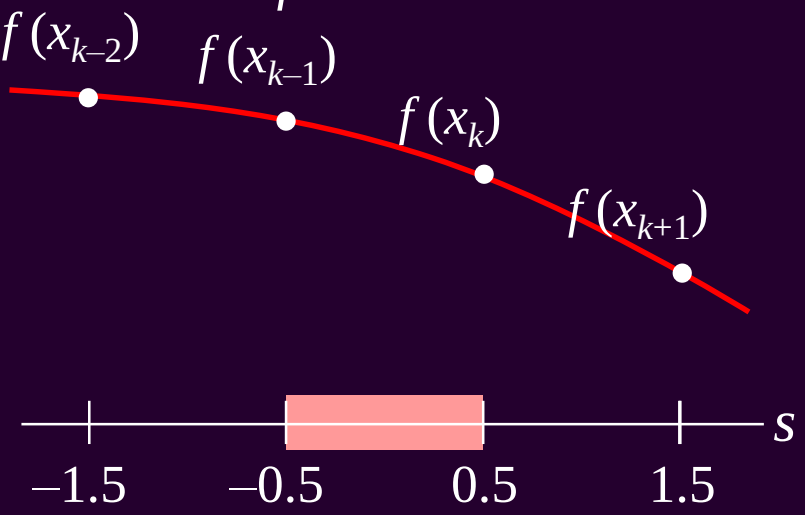
  – The interpolating polynomial is

$$\frac{-f(x_{k-2})+3f(x_{k-1})-3f(x_k)+f(x_{k+1})}{6}\delta^3+\frac{f(x_{k-2})-f(x_{k-1})-f(x_k)+f(x_{k+1})}{4}\delta^2$$
$$+\frac{f(x_{k-2})-27f(x_{k-1})+27f(x_k)-f(x_{k+1})}{24}\delta+\frac{-f(x_{k-2})+9f(x_{k-1})+9f(x_k)-f(x_{k+1})}{16}$$

  – This approximates the function f at

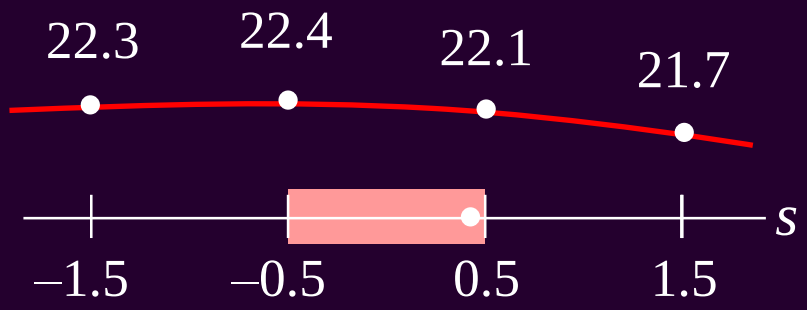
$$\frac{x_{k-1}+x_k}{2}+\delta h$$

$$\begin{pmatrix} -3.375 & 2.25 & -1.5 & 1 \\ -0.125 & 0.25 & -0.5 & 1 \\ 0.125 & 0.25 & 0.5 & 1 \\ 3.375 & 2.25 & 1.5 & 1 \end{pmatrix}\begin{pmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix}=\begin{pmatrix} f(x_{k-2}) \\ f(x_{k-1}) \\ f(x_k) \\ f(x_{k+1}) \end{pmatrix}$$



Centered cubic interpolation

- For example, suppose that we have the following data:
  (105920, 22.3), (106040, 22.4) , (106160, 22.1) and (106280, 21.7)
  - Thus:
    - The midpoint is 106100
    - The step size is $h = 120$
    - The polynomial is $0.05\,\delta^3 - 0.125\,\delta^2 - 0.3125\,\delta + 22.28125$
  - If you wanted to approximate at time 106157,
    we note this is $106100 + 120 \cdot 0.475$
  - Thus, $((0.05 \cdot 0.475 - 0.125) \cdot 0.475 - 0.3125) \cdot 0.475 + 22.28125$
    $= 22.109968$



Backward interpolating polynomials

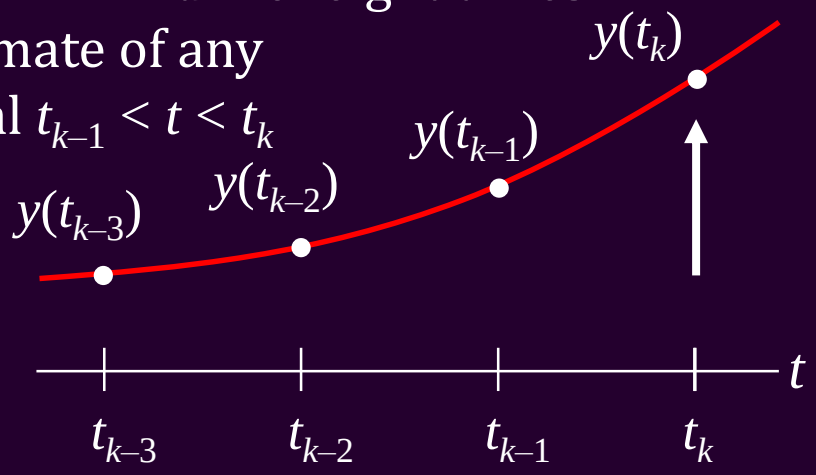
- What happens if you only have prior information:

$$(t_{k-3}, y(t_{k-3})), (t_{k-2}, y(t_{k-1})), (t_{k-1}, y(t_{k-1})) \text{ and } (t_k, y(t_k))$$

  - Question: Do we want to only approximate values in the past, or do we want to extrapolate values into the future?
  - Unfortunately, the error grows quickly outside any interval

$$t_{k-n}, \ldots, t_k$$

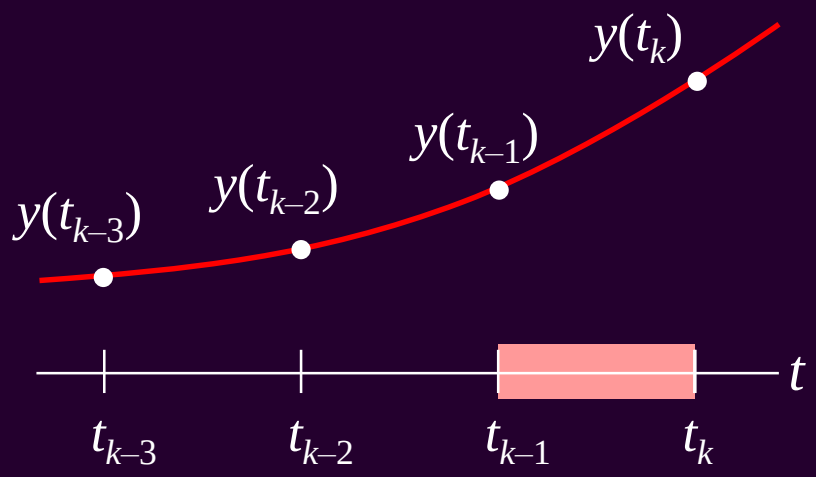
  - Using *interpolating* polynomials to estimate the value at t_{k+1} results in an error a minimum of eight times larger than the estimate of any value on the interval $t_{k-1} < t < t_k$



Backward quadratic interpolation

- Instead, we will focus only on formulas that approximate the function on the interval

$$t_{k-1} < t < t_k$$



Backward quadratic interpolation

- As before, we will shift and scale
  - Solving this, we get the formula:

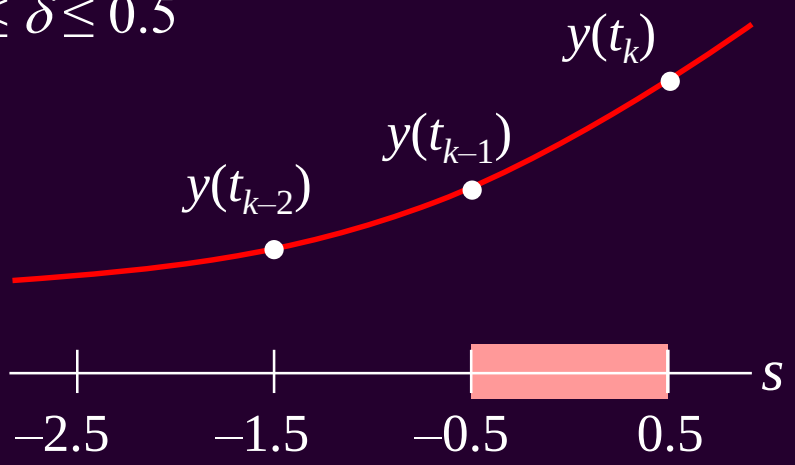
$$\left(\frac{y(t_{k-2})-2y(t_{k-1})+y(t_k)}{2}\right)\delta^2+\left(y(t_k)-y(t_{k-1})\right)\delta+\frac{-y(t_{k-2})+6y(t_{k-1})+3y(t_k)}{8}$$

  - This formula approximates values for

$$\frac{t_{k-1}+t_k}{2}+\delta\Delta t$$

  - The values are $-0.5 \le \delta \le 0.5$

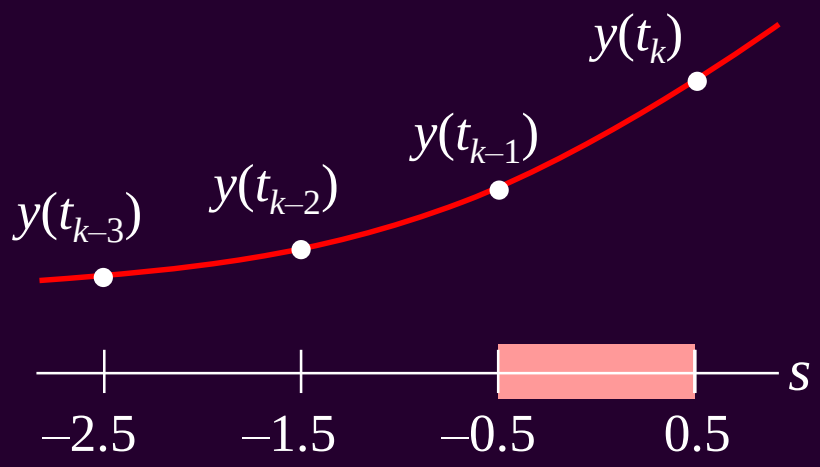
$$\begin{pmatrix} 0.25 & 0.5 & 1 \\ 0.25 & -0.5 & 1 \\ 2.25 & -1.5 & 1 \end{pmatrix}\begin{pmatrix} a_2 \\ a_1 \\ a_0 \end{pmatrix}=\begin{pmatrix} y(t_k) \\ y(t_{k-1}) \\ y(t_{k-2}) \end{pmatrix}$$



Backward cubic interpolation

- Similarly, we could find an interpolating cubic:

$$\frac{-y(t_{k-3})+3y(t_{k-2})-3y(t_{k-1})+y(t_k)}{6}\delta^3+\frac{-y(t_{k-3})+5y(t_{k-2})-7y(t_{k-1})+3y(t_k)}{4}\delta^2$$
$$+\frac{y(t_{k-3})-3y(t_{k-2})-21y(t_{k-1})+23y(t_k)}{24}\delta+\frac{y(t_{k-3})-5y(t_{k-2})+15y(t_{k-1})+5y(t_k)}{16}$$



Implementations

This C++ code is meant for demonstration purposes only and not required for the examination

```
template <typename T>
class Backward4 {
    public:
        Backward4( T y[4] );
        T evaluate( T delta ) const;

    private:
        T coeffs_[4];
};
```

```
template <typename T>
Backward4<T>::Backward4( T y[4] ):
coeffs_{ (y[0] - 5*y[1] + 15*y[2] +  5*y[3])/16.0,
         (y[0] - 3*y[1] - 21*y[2] + 23*y[3])/24.0,
        (-y[0] + 5*y[1] -  7*y[2] +  3*y[3])/4.0,
        (-y[0] + 3*y[1] -  3*y[2] +    y[3])/6.0 } {
    // Empty constructor
}
```

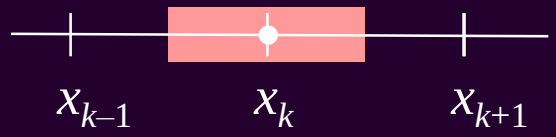
```
template <typename T>
T Backward4<T>::evaluate( T delta ) const {
    assert( (delta >= -0.5) && (delta <= 0.5) );

    return (
        (coeffs_[3]*delta + coeffs_[2])*delta + coeffs_[1]
    )*delta + coeffs_[0];
}
```

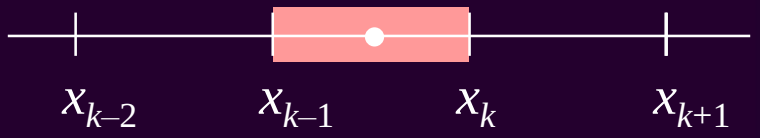
Summary

- Linear $s \leftarrow \frac{1}{h}\left(x - \frac{x_k + x_{k-1}}{2}\right)$

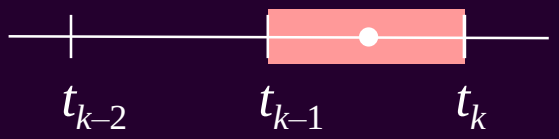
- Centered quadratic $s \leftarrow \frac{1}{h}(x - x_k)$



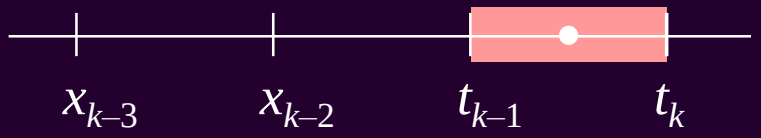
- Centered cubic $s \leftarrow \frac{1}{h}\left(x - \frac{x_k + x_{k-1}}{2}\right)$



- Backward quadratic $s \leftarrow \frac{1}{h}\left(x - \frac{x_k + x_{k-1}}{2}\right)$



- Backward cubic $s \leftarrow \frac{1}{\Delta t}\left(t - \frac{t_k + t_{k-1}}{2}\right)$



You don't have to memorize these formulas; remember the idea

Summary

- Linear $\left[f(x_k)-f(x_{k-1})\right]s+\dfrac{f(x_{k-1})+f(x_k)}{2}$
- Centered quadratic $\dfrac{f(x_{k-1})-2f(x_k)+f(x_{k+1})}{2}s^2+\dfrac{f(x_{k+1})-f(x_{k-1})}{2}s+f(x_k)$
- Centered cubic $\dfrac{-f(x_{k-2})+3f(x_{k-1})-3f(x_k)+f(x_{k+1})}{6}s^3+\dfrac{f(x_{k-2})-f(x_{k-1})-f(x_k)+f(x_{k+1})}{4}s^2+\dfrac{f(x_{k-2})-27f(x_{k-1})+27f(x_k)-f(x_{k+1})}{24}s+\dfrac{-f(x_{k-2})+9f(x_{k-1})+9f(x_k)-f(x_{k+1})}{16}$
- Backward quadratic $\left(\dfrac{y(t_{k-2})-2y(t_{k-1})+y(t_k)}{2}\right)s^2+\left(y(t_k)-y(t_{k-1})\right)s+\dfrac{-y(t_{k-2})+6y(t_{k-1})+3y(t_k)}{8}$
- Backward cubic $\dfrac{-y(t_{k-3})+3y(t_{k-2})-3y(t_{k-1})+y(t_k)}{6}s^3+\dfrac{-y(t_{k-3})+5y(t_{k-2})-7y(t_{k-1})+3y(t_k)}{4}s^2+\dfrac{y(t_{k-3})-3y(t_{k-2})-21y(t_{k-1})+23y(t_k)}{24}s+\dfrac{y(t_{k-3})-5y(t_{k-2})+15y(t_{k-1})+5y(t_k)}{16}$

You don't have to memorize these formulas; remember the idea

Summary

- Following this topic, you now
  - Understand that we can find interpolating polynomials that are designed to:
    - Minimize the condition number
    - Be appropriate for Horner's rule and minimizing numeric error
  - Understand how to use 2, 3 and 4 points to estimate values in the middle of these sets
  - Understand how to use the last 2, 3 and 4 points to estimate values in the most recent time interval
  - Know that this is useful only for interpolating values and not for extrapolation

References

[1] https://en.wikipedia.org/wiki/Polynomial_interpolation

Acknowledgments

Brian Nguyen for noting an error in the order of the vector entries on p.18.

Colophon

These slides were prepared using the Cambria typeface. Mathematical equations use Times New Roman, and source code is presented using `Consolas`. Mathematical equations are prepared in MathType by Design Science, Inc.

Examples may be formulated and checked using Maple by Maplesoft, Inc.

The photographs of flowers and a monarch butter appearing on the title slide and accenting the top of each other slide were taken at the Royal Botanical Gardens in October of 2017 by Douglas Wilhelm Harder. Please see

https://www.rbg.ca/

for more information.

Disclaimer

These slides are provided for the ECE 204 *Numerical methods* course taught at the University of Waterloo. The material in it reflects the author's best judgment in light of the information available to them at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. The authors accept no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.