ACM40290: Numerical Algorithms

Numerical Differentiation

Dr Barry Wardell
School of Mathematics and Statistics
University College Dublin

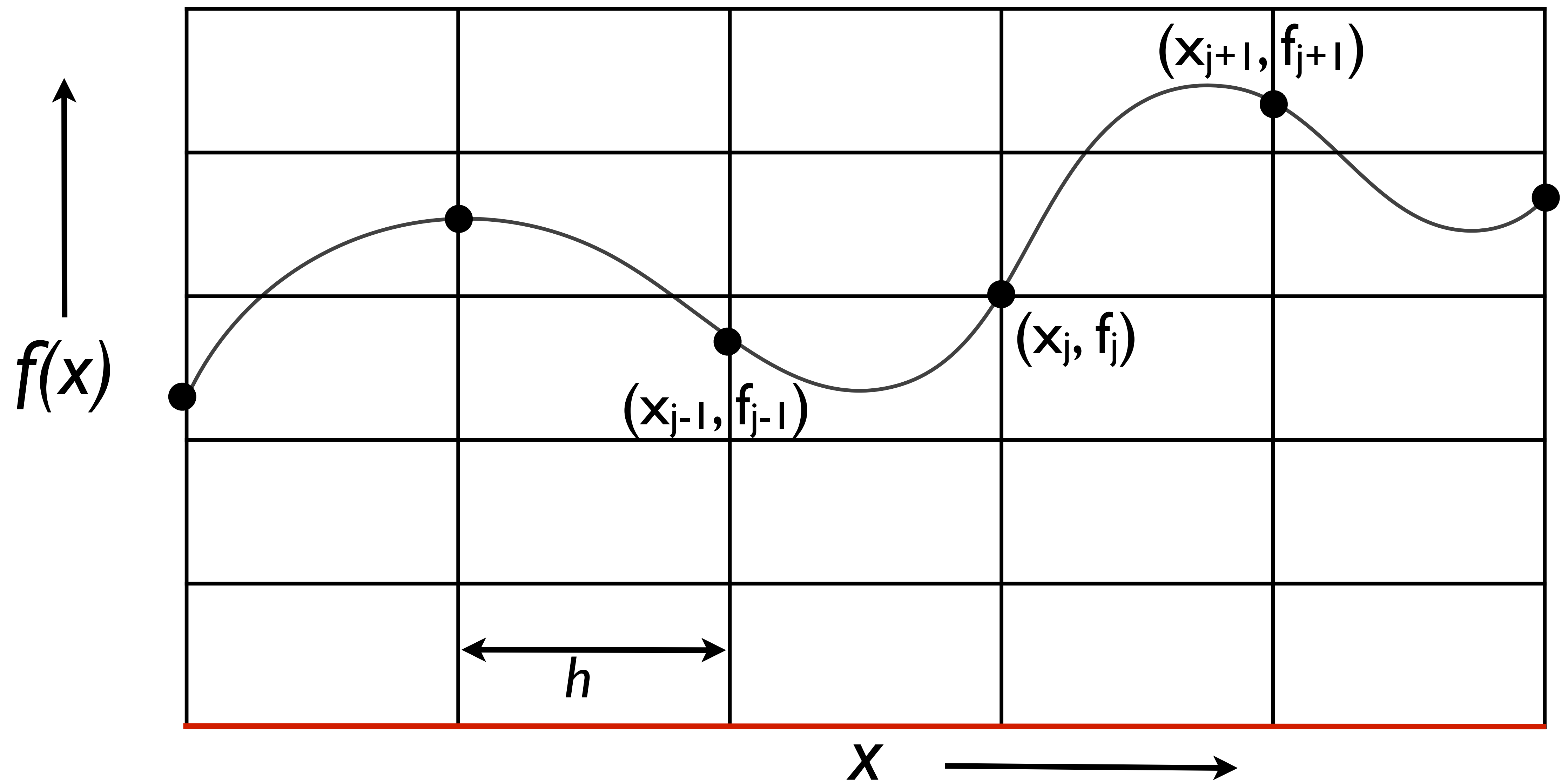
Finite differencing

- We want a method for computing the derivatives of a function.
- Finite differencing is a simple, straightforward approach. In fact, we have already encountered it.
- Start from the definition of a derivative

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Discretise

Introduce a grid of points on which we have values for a function, define $f_j=f(x_j)$.



Finite differencing

- First-order accurate approximation to the derivative is given by using the limiting definition of a derivative on the grid

$$\frac{df}{dx} \approx \frac{f_{j+1} - f_j}{h}$$

- Could equivalently also use a different pair of points on the grid

$$\frac{df}{dx} \approx \frac{f_j - f_{j-1}}{h}$$

- The two approximations have *first order* errors

Finite differencing

- We can get a *second-order* accurate result using a centred derivative

$$\frac{df}{dx} \approx \frac{f_{j+1} - f_{j-1}}{2h}$$

- Likewise for second derivatives,

$$\frac{d^2 f}{dx^2} \approx \frac{f_{j+1} - 2f_j + f_{j-1}}{h^2}$$

Accuracy

- A finite differencing derivative is only an approximation to the actual derivative.
- It becomes increasingly accurate as the grid spacing decreases.
- Can we say more than this?

Finite differencing

- Taylor's theorem

$$f(x+h) = f(x) + f'(x)h + \tfrac{1}{2}f''(x)h^2 + \cdots$$

- Rearranging this, we recover exactly our finite difference formula

$$\frac{df}{dx} \approx \frac{f_{j+1} - f_j}{h} + \mathcal{O}(h)$$

- We call this is a first-order accurate finite difference since the error is order h^1 (assuming f is sufficiently smooth — in this case that it is twice differentiable).

Finite differencing

- Similarly, can also use Taylor's theorem at *x-h*

$$f(x-h) = f(x) + f'(x)(-h) + \tfrac{1}{2} f''(x)(-h)^2 + \cdots$$

- Rearranging this, we recover exactly our finite difference formula

$$\frac{df}{dx} \approx \frac{f_j - f_{j-1}}{h} + \mathcal{O}(h)$$

- Again, this is a first-order accurate finite difference derivative.

Higher order finite differencing

- Combining the two previous results,

$$f(x+h) - f(x-h) =$$

$$f(x) + f'(x)h + \tfrac{1}{2}f''(x)h^2 + \tfrac{1}{6}f'''(x)h^3$$

$$- f(x) - f'(x)(-h) - \tfrac{1}{2}f''(x)(-h)^2 - \tfrac{1}{6}f'''(x)(-h)^3 + \cdots$$

we find that the order h errors cancel and we have a second-order accurate finite-difference formula

$$\frac{df}{dx} \approx \frac{f_{j+1} - f_{j-1}}{2h} + \mathcal{O}(h^2)$$

Higher order finite differencing

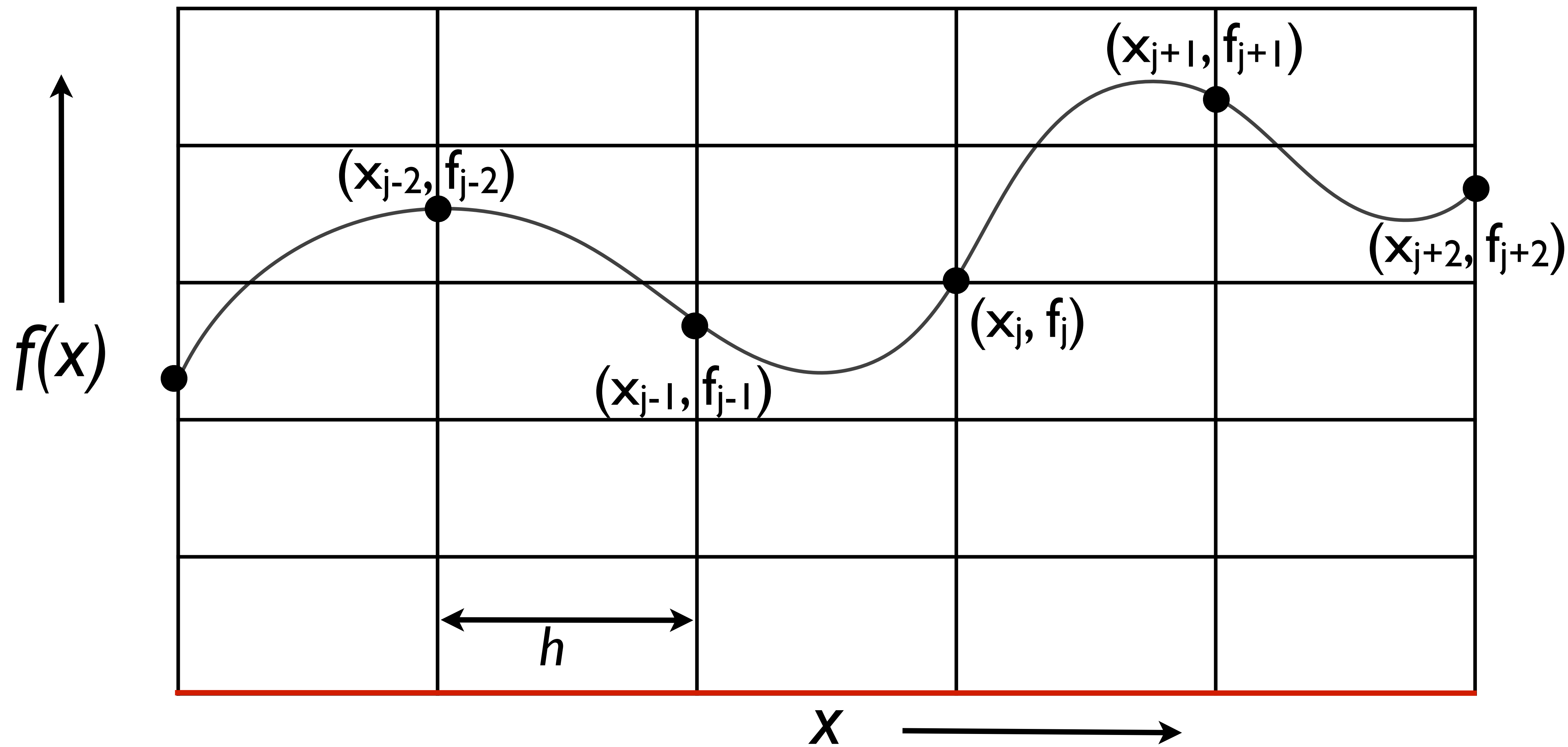
- Even higher order approximations are also possible.
- Fourth order finite difference:

$$\frac{df}{dx} = \frac{f_{j-2} - 8f_{j-1} + 8f_{j+1} - f_{j+2}}{12h} + \mathcal{O}(h^4)$$

- We could have derived this formula by combining the Taylor series for *f(x-2h)*, *f(x-h)*, *f(x+h)* and *f(x+2h)*.
- Higher order is more accurate, but requires more points (more calculation).

Higher order finite differencing

Higher order is more accurate, but requires more points (more calculation).



Higher order finite differencing

- High-order finite difference formulas can be derived by combining the Taylor series for different points, but this is cumbersome.
- A more general approach is to fit a polynomial and differential that.
- Example: determine the unique quadratic passing through the three points (x_{j-1}, f_{j-1}), (x_j, f_j), (x_{j+1}, f_{j+1}), differentiate it and this gives the second-order finite difference formula.
- In general, a finite difference formula using n points will be exact for functions that are polynomials of degree $n-1$ and have asymptotic order at least $n-m$. Sometimes higher asymptotic order because of cancellation.

Higher order finite differencing

- There is a general prescription for a finite difference approximation to an order *m* derivative with *n*+*1* points, evaluated at a point *s* (relative to the left-most point)

SIAM REV.
Vol. 40, No. 3, pp. 685–691, September 1998

012

CALCULATION OF WEIGHTS IN FINITE DIFFERENCE FORMULAS*

BENGT FORNBERG†

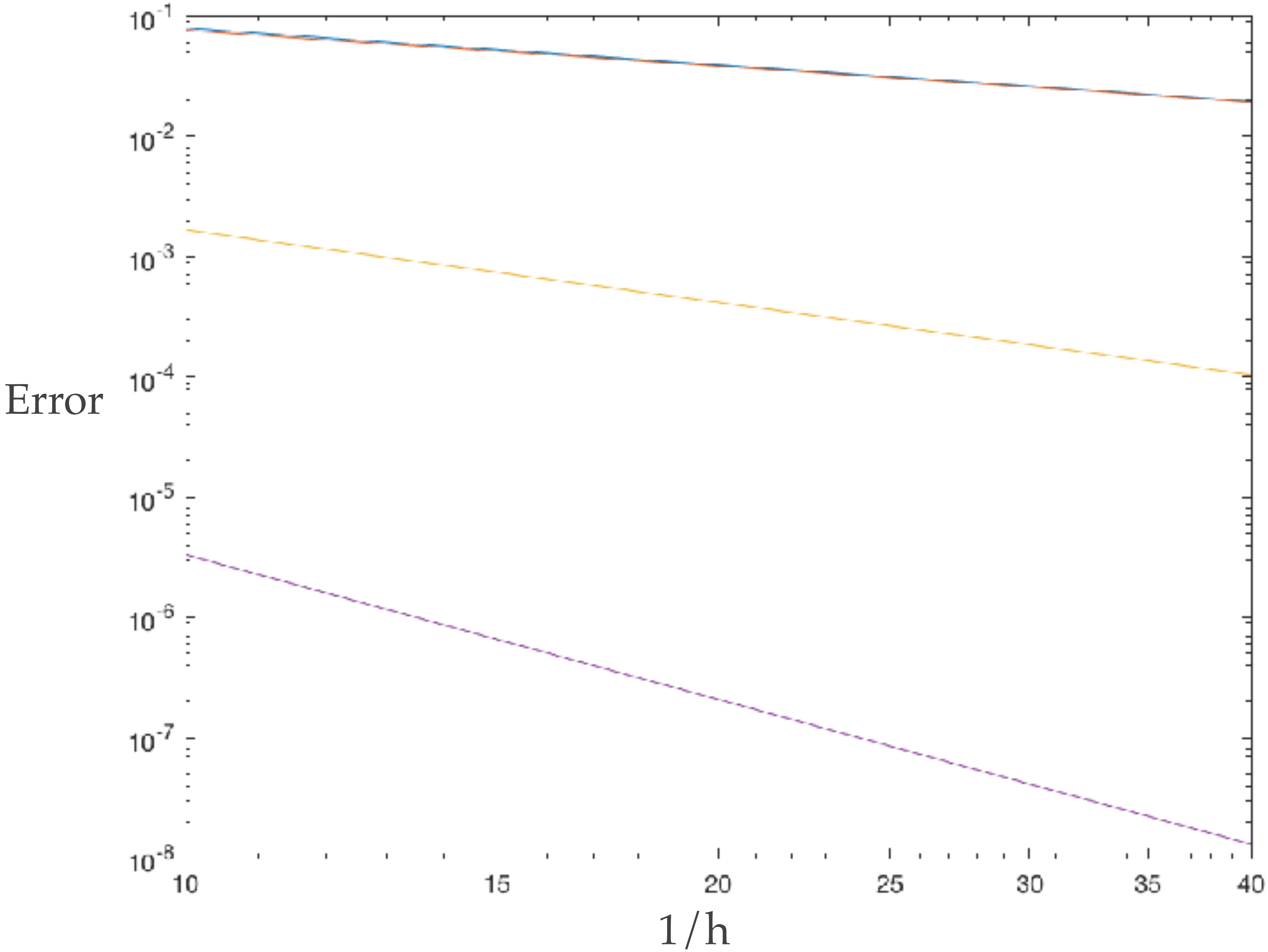
Abstract. The classical techniques for determining weights in finite difference formulas were either computationally slow or very limited in their scope (e.g., specialized recursions for centered and staggered approximations, for Adams–Bashforth-, Adams–Moulton-, and BDF-formulas for ODEs, etc.). Two recent algorithms overcome these problems. For equispaced grids, such weights can be found very conveniently with a two-line algorithm when using a symbolic language such as Mathematica (reducing to one line in the case of explicit approximations). For arbitrarily spaced grids, we describe a computationally very inexpensive numerical algorithm.

Mathematica code: CoefficientList[Normal[Series[x^s Log[x]^m, {x, 1, n}]/h^m], x]

Example: derivative of sin(x)

```
hL=0.1; hM=0.05; hH=0.025; h = [hL, hM, hH];

x0 = 1.0;

fp_exact = cos(x0);

fp_fd1u = (sin(x0+h)-sin(x0))./h;
fp_fd1d = (sin(x0)-sin(x0-h))./h;
fp_fd2 = (sin(x0+h)-sin(x0-h))./(2*h);
fp_fd4 = (sin(x0-2*h)-8*sin(x0-h)+8*sin(x0+h)-sin(x0+2*h))./(12*h);

err_fd1u = abs(1-fp_fd1u/fp_exact);
err_fd1d = abs(1-fp_fd1d/fp_exact);
err_fd2 = abs(1-fp_fd2/fp_exact);
err_fd4 = abs(1-fp_fd4/fp_exact);

loglog(1./h, err_fd1u, 1./h, err_fd1d, 1./h, err_fd2, 1./h, err_fd4)
```

Example: derivative of sin(x)



Convergence

- Numerical approximation approaches exact solution as *h* goes to 0.
- Convergence *rate* depends on the numerical scheme.
- Numerical solution differs from exact solution by an amount which depends on the resolution (i.e. *h*)

$$f'(x_0) = f'_{\text{exact}}(x_0) + Ch^p$$

- *p* is called the *convergence rate*.

Convergence Rate

- If we know the exact solution, we can determine the convergence rate by running at two resolutions

$$f'_L(x_0) = f'_{\text{exact}}(x_0) + Ch_L^p$$
$$f'_H(x_0) = f'_{\text{exact}}(x_0) + Ch_H^p$$

- We know $f'_{exact}(x_0), f'_L(x_0) f'_H(x_0), h_L, h_H$
- Solve 2 equations for 2 unknowns: C, p
- Useful to test a algorithm against a known analytic solution — p is determined by the algorithm.

Convergence Rate

- If we don't have an exact solution, we can still determine the convergence rate by using three resolutions

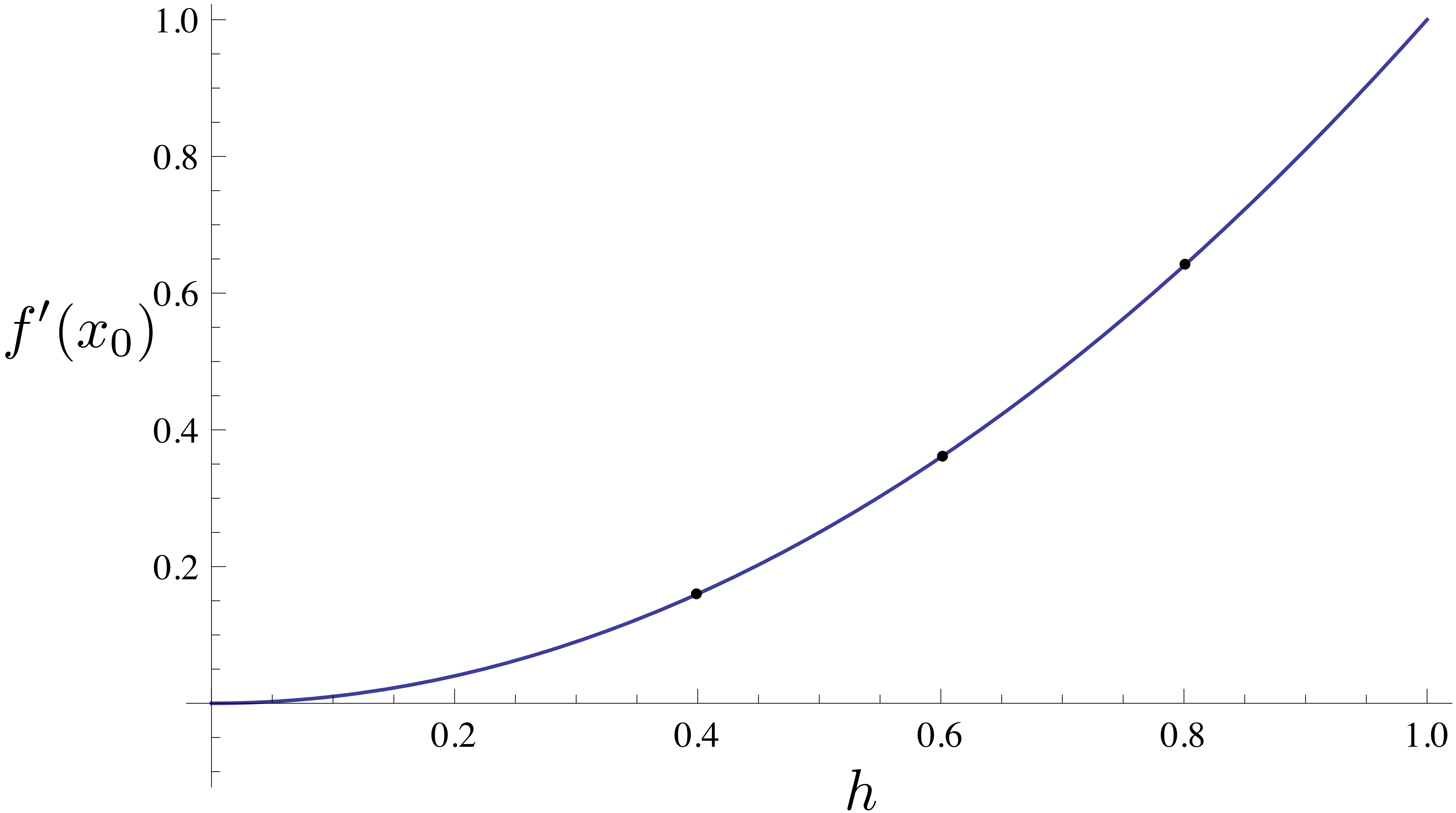
$$f'_L(x_0) = f'_{\mathrm{exact}}(x_0) + Ch_L^p$$
$$f'_M(x_0) = f'_{\mathrm{exact}}(x_0) + Ch_M^p$$
$$f'_H(x_0) = f'_{\mathrm{exact}}(x_0) + Ch_H^p$$

- We know $f'_L(x_0), f'_M(x_0), f'_H(x_0), h_L, h_M, h_H$
- Solve 3 equations for 3 unknowns: $C, p, f'_{\mathrm{exact}}(x_0)$
- Useful when you don't have any known solutions to test against

Richardson Extrapolation

- Having convergence is very powerful
- It allows extrapolation from numerical solutions at specific h to determine what the value would be for infinite resolution.
- The extrapolation procedure is called *Richardson extrapolation* and can dramatically improve the accuracy of numerical results

Richardson Extrapolation



Richardson Extrapolation

- Given numerical solutions at two resolutions

$$f'_L(x) = f'_{\text{exact}}(x) + Ch_L^p$$
$$f'_H(x) = f'_{\text{exact}}(x) + Ch_H^p$$

- If we have already established convergence, then we know p
- We also know $f'_L(x), f'_H(x), h_L, h_H$
- Solve 2 equations for 2 unknowns: $f'_{exact}(x_0), C$
- We now know the exact solution, $f'_{exact}(x_0)$