Math 541 - Numerical Analysis

Numerical Differentiation and Richardson Extrapolation

Joseph M. Mahaffy,
⟨jmahaffy@mail.sdsu.edu⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://jmahaffy.sdsu.edu

Spring 2018

Outline

1. Numerical Differentiation
   - Ideas and Fundamental Tools
   - Moving Along...

2. Richardson's Extrapolation
   - A Nice Piece of "Algebra Magic"

Numerical Differentiation: The Big Picture

The goal of numerical differentiation is to compute an accurate approximation to the derivative(s) of a function.

Given measurements $\{f_i\}_{i=0}^n$ of the underlying function $f(x)$ at the node values $\{x_i\}_{i=0}^n$, our task is to estimate $f'(x)$ (and, later, higher derivatives) in the same nodes.

The strategy: Fit a polynomial to a cleverly selected subset of the nodes, and use the derivative of that polynomial as the approximation of the derivative.

Numerical Differentiation

Definition (Derivative as a limit)

The derivative of f at x_0 is

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

The obvious approximation is to fix h "small" and compute

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}.$$

Problems: Cancellation and roundoff errors. — For small values of h, $f(x_0 + h) \approx f(x_0)$ so the difference may have very few *significant digits* in finite precision arithmetic.
Smaller h is not necessarily better numerically.

Main Tools for Numerical Differentiation

Again Taylor's Theorem is critical for determining accuracy of our algorithms...

Theorem (Taylor's Theorem)

Suppose $f \in C^n[a,b]$, $f^{(n+1)}$ *exists on* $[a,b]$, *and* $x_0 \in [a,b]$. *Then for all* $x \in (a,b)$, *there exists* $\xi(x) \in (\min(x_0,x), \max(x_0,x))$ *with* $f(x) = P_n(x) + R_n(x)$ *where*

$$P_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k, \quad R_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)^{(n+1)}.$$

$P_n(x)$ *is the* ***Taylor polynomial of degree*** n, *and* $R_n(x)$ *is the* ***remainder term*** *(truncation error).*

Main Tools for Numerical Differentiation

Our second tool for building Differentiation and Integration schemes are the **Lagrange Coefficients**

$$L_{n,k}(x) = \prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j}$$

Recall: $L_{n,k}(x)$ is the nth degree polynomial which is 1 in x_k and 0 in the other nodes $(x_j,\ j \neq k)$.

Previously we have used the family $L_{n,0}(x)$, $L_{n,1}(x)$, $\ldots$, $L_{n,n}(x)$ to build the *Lagrange interpolating polynomial.* — A good tool for providing polynomial behavior.

Now, lets combine our tools and look at differentiation.

Getting an Error Estimate — Taylor Expansion

$$\begin{aligned}\frac{f(x_0+h)-f(x_0)}{h} &= \frac{1}{h}\left[f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(\xi(x))-f(x_0)\right]\\ &= f'(x_0)+\mathbf{\frac{h}{2}f''(\xi(x))}\end{aligned}$$

If $f''(\xi(x))$ is bounded, *i.e.*

$$|f''(\xi(x))| < M, \quad \forall \xi(x) \in (x_0, x_0+h)$$

then we have

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}, \quad \text{with an error less than} \quad \frac{M|h|}{2}.$$

This is the ***approximation error***, which is $\mathcal{O}(h)$.

(Roundoff error, $\sim \epsilon_{\text{mach}} \approx 10^{-16}$, not taken into account).

Using Higher Degree Polynomials to get Better Accuracy

Suppose $\{x_0, x_1, \ldots, x_n\}$ are distinct points in an interval $\mathcal{I}$, and $f \in C^{n+1}(\mathcal{I})$, we can write

$$f(x) = \underbrace{\sum_{k=0}^{n} f(x_k) L_{n,k}(x)}_{\text{Lagrange Interp. Poly.}} + \underbrace{\frac{\prod_{k=0}^{n}(x - x_k)}{(n+1)!} f^{(n+1)}(\xi(x))}_{\text{Error Term}}$$

Formal differentiation of this expression gives:

$$\begin{aligned} f'(x) &= \sum_{k=0}^{n} f(x_k) L'_{n,k}(x) + \frac{d}{dx}\left[\frac{\prod_{k=0}^{n}(x - x_k)}{(n+1)!}\right] f^{(n+1)}(\xi(x)) \\ &\quad + \frac{\prod_{k=0}^{n}(x - x_k)}{(n+1)!} \frac{d}{dx}\left[f^{(n+1)}(\xi(x))\right]. \end{aligned}$$

Note: When we evaluate $f'(x_j)$ ***at the node points*** (x_j) the last term gives no contribution. ($\Rightarrow$ we don't have to worry about it...)

Exercising the Product Rule for Differentiation

$$\frac{d}{dx}\left[\frac{\prod_{k=0}^{n}(x-x_k)}{(n+1)!}\right] =$$

$$\frac{1}{(n+1)!}\left[(x-x_1)(x-x_2)\cdots(x-x_n)+(x-x_0)(x-x_2)\cdots(x-x_n)+\cdots\right] =$$

$$\frac{1}{(n+1)!}\sum_{j=0}^{n}\left[\prod_{k=0,k\neq j}^{n}(x-x_k)\right]$$

Now, if we let $x = x_\ell$ for some particular value of ℓ, only the product which skips that value of $j = \ell$ is non-zero... *e.g.*

$$\frac{1}{(n+1)!}\sum_{j=0}^{n}\left[\prod_{k=0,k\neq j}^{n}(x-x_k)\right]\Bigg|_{\mathbf{x}=\mathbf{x}_\ell} = \frac{1}{(n+1)!}\prod_{k=0,k\neq\ell}^{n}(x_\ell-x_k)$$

The $(n+1)$ point formula for approximating $f'(x_j)$

Putting it all together yields what is known as the $(n+1)$ point formula for approximating $f'(x_j)$:

$$f'(x_j) = \sum_{k=0}^{n} f(x_k) L'_{n,k}(x_j) + \frac{f^{(n+1)}(\xi)}{(n+1)!} \left[\prod_{\substack{k=0 \\ k \neq j}}^{n} (x_j - x_k) \right]$$

Note: The formula is most useful when the node points are equally spaced (it can be computed once and stored), *i.e.*

$$x_k = x_0 + kh.$$

Now, we have to compute the derivatives of the Lagrange coefficients, *i.e.* $L_{n,k}(x)$... [We can no longer dodge this task!]

Example: 3-point Formulas, I/III

Building blocks:

$$L_{2,0}(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}, \quad L'_{2,0}(x) = \frac{(x-x_1)+(x-x_2)}{(x_0-x_1)(x_0-x_2)}$$

$$L_{2,1}(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}, \quad L'_{2,1}(x) = \frac{(x-x_0)+(x-x_2)}{(x_1-x_0)(x_1-x_2)}$$

$$L_{2,2}(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}, \quad L'_{2,2}(x) = \frac{(x-x_0)+(x-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

Formulas:

$$\begin{aligned} f'(x_j) &= f(x_0)\left[\frac{2x_j-x_1-x_2}{(x_0-x_1)(x_0-x_2)}\right] + f(x_1)\left[\frac{2x_j-x_0-x_2}{(x_1-x_0)(x_1-x_2)}\right] \\ &+ f(x_2)\left[\frac{2x_j-x_0-x_1}{(x_2-x_0)(x_2-x_1)}\right] + \frac{f^{(3)}(\xi_j)}{6}\prod_{\substack{k=0\\k\neq j}}^{2}(x_j-x_k). \end{aligned}$$

Example: 3-point Formulas, II/III

When the points are equally spaced...

$$\begin{cases} f'(x_0) = \dfrac{1}{2h}\left[-3f(x_0) + 4f(x_1) - f(x_2)\right] + \dfrac{h^2}{3} f^{(3)}(\xi_0) \\ f'(x_1) = \dfrac{1}{2h}\left[-f(x_0) + f(x_2)\right] - \dfrac{h^2}{6} f^{(3)}(\xi_1) \\ f'(x_2) = \dfrac{1}{2h}\left[f(x_0) - 4f(x_1) + 3f(x_2)\right] + \dfrac{h^2}{3} f^{(3)}(\xi_2) \end{cases}$$

Use x_0 as the reference point — $x_k = x_0 + kh$:

$$\begin{cases} f'(x_0) = \dfrac{1}{2h}\left[-3f(x_0) + 4f(x_0+h) - f(x_0+2h)\right] + \dfrac{h^2}{3} f^{(3)}(\xi_0) \\ f'(x_0+h) = \dfrac{1}{2h}\left[-f(x_0) + f(x_0+2h)\right] - \dfrac{h^2}{6} f^{(3)}(\xi_1) \\ f'(x_0+2h) = \dfrac{1}{2h}\left[f(x_0) - 4f(x_0+h) + 3f(x_0+2h)\right] + \dfrac{h^2}{3} f^{(3)}(\xi_2) \end{cases}$$

Example: 3-point Formulas, III/III

$$\begin{cases} f'(x_0) = \frac{1}{2h}\left[-3f(x_0) + 4f(x_0+h) - f(x_0+2h)\right] + \frac{h^2}{3} f^{(3)}(\xi_0) \\ \mathbf{f'(x_0^*) = \frac{1}{2h}\left[-f(x_0^*-h) + f(x_0^*+h)\right] - \frac{h^2}{6} f^{(3)}(\xi_1)} \\ f'(x_0^+) = \frac{1}{2h}\left[f(x_0^+-2h) - 4f(x_0^+-h) + 3f(x_0^+)\right] + \frac{h^2}{3} f^{(3)}(\xi_2) \end{cases}$$

After the substitution $x_0 + h \to x_0^*$ in the second equation, and $x_0 + 2h \to x_0^+$ in the third equation.

Note#1: The third equation can be obtained from the first one by setting $h \to -h$.

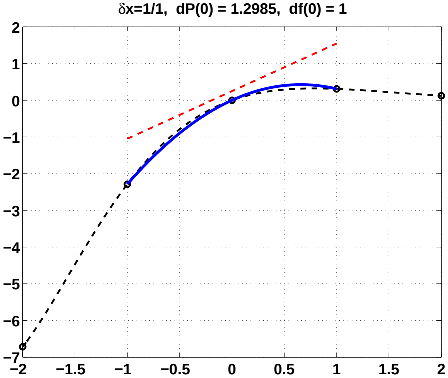
Note#2: The error is smallest in the second equation.

Note#3: The second equation is a two-sided approximation, the first and third one-sided approximations.

Note#4: We can drop the superscripts $^*, ^+$...

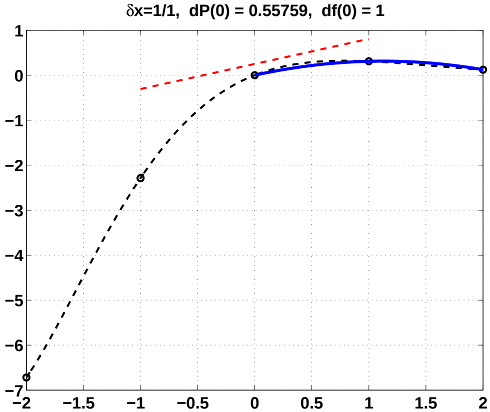
3-point Formulas: Illustration — Centered Formula

Consider $f(x) = e^{-x}\sin(x)$.



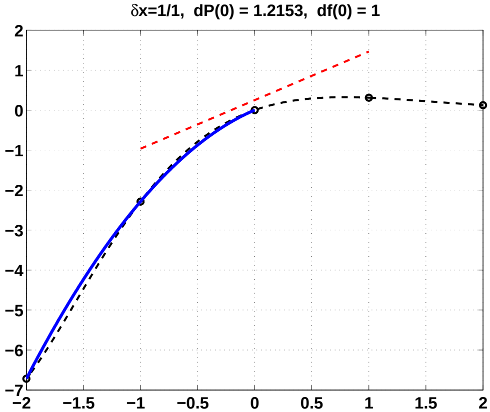
$$f'(x_0) = \frac{1}{2h}\left[-f(x_0 - h) + f(x_0 + h)\right] - \frac{h^2}{6} f^{(3)}(\xi_1)$$

3-point Formulas: Illustration Forward Formula



$$f'(x_0) = \frac{1}{2h}\left[-3f(x_0) + 4f(x_0+h) - f(x_0+2h)\right] + \frac{h^2}{3}f^{(3)}(\xi_0)$$

3-point Formulas: Illustration — Backward Formula



$$f'(x_0) = \frac{1}{2h}\left[f(x_0 - 2h) - 4f(x_0 - h) + 3f(x_0)\right] + \frac{h^2}{3} f^{(3)}(\xi_2)$$

5-point Formulas

If we want even better approximations we can go to 4-point, 5-point, 6-point, etc... formulas.

The most accurate (smallest error term) 5-point formula is:

$$f'(x_0) = \frac{f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)}{12h} + \frac{h^4}{30}f^{(5)}(\xi)$$

Sometimes (*e.g* for end-point approximations like the clamped splines), we need one-sided formulas

$$f'(x_0) = \frac{-25f(x_0)+48f(x_0+h)-36f(x_0+2h)+16f(x_0+3h)-3f(x_0+4h)}{12h} + \frac{h^4}{5}f^{(5)}(\xi).$$

5-Point Formulas

Reference

$$f'(x_0) = \frac{1}{12h}\Big[-25f(x_0) + 48f(x_1) - 36f(x_2) + 16f(x_3) - 3f(x_4)\Big]$$

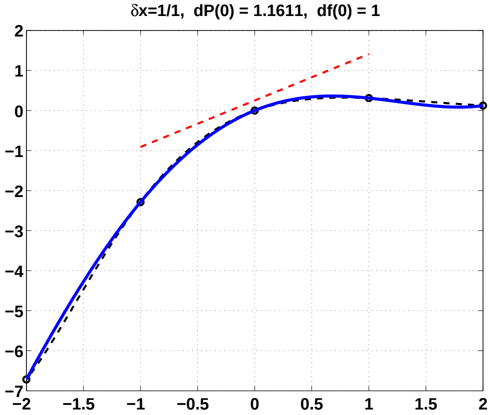
$$f'(x_0) = \frac{1}{12h}\Big[-3f(x_{-1}) - 10f(x_0) + 18f(x_1) - 6f(x_2) + f(x_3)\Big]$$

$$f'(x_0) = \frac{1}{12h}\Big[f(x_{-2}) - 8f(x_{-1}) + 8f(x_1) - f(x_2)\Big]$$

$$f'(x_0) = \frac{1}{12h}\Big[-f(x_{-3}) + 6f(x_{-2}) - 18f(x_{-1}) + 10(x_0) + 3f(x_1)\Big]$$

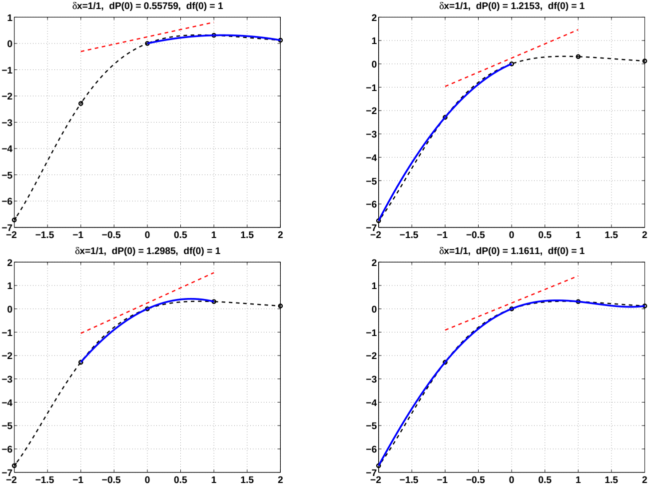
$$f'(x_0) = \frac{1}{12h}\Big[3f(x_{-4}) - 16f(x_{-3}) + 36f(x_{-2}) - 48(x_{-1}) + 25f(x_0)\Big]$$

5-point Formulas: Illustration Centered Formula

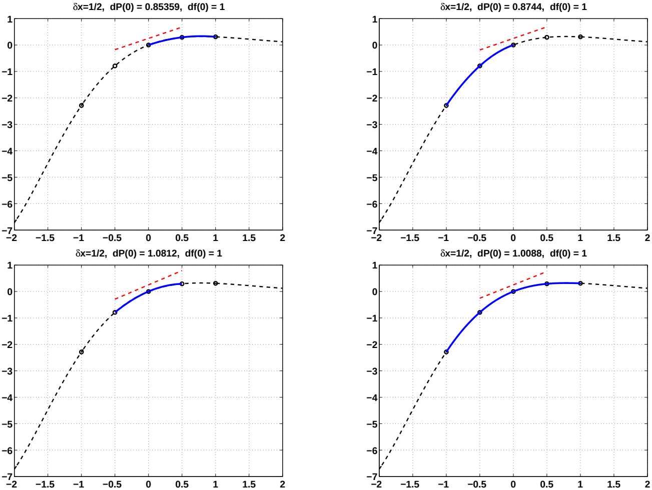


$$f'(x_0) = \frac{f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)}{12h} + \frac{h^4}{30}f^{(5)}(\xi)$$

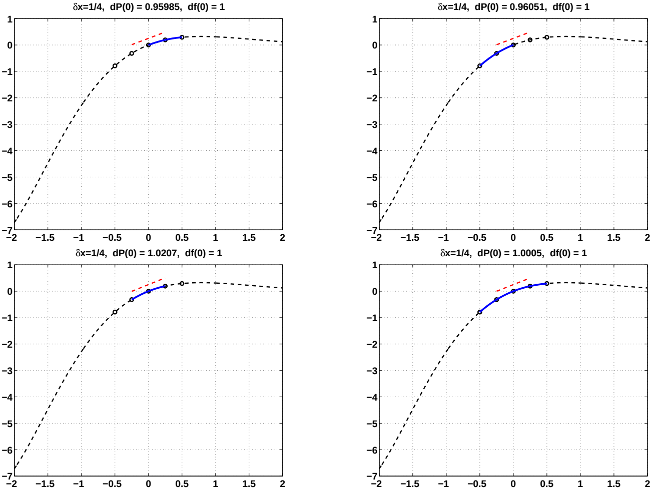
3-point and 5-point Formulas $\delta x = 1$



3-point and 5-point Formulas $\delta x = 1/2$



3-point and 5-point Formulas $\delta x = 1/4$



3-point and 5-point Formulas | Summary

For the example: $f(x) = e^{-x}\sin(x)$ around $x = 0$.

dx	3-Point Formulas Backward	3-Point Formulas Center	3-Point Formulas Forward	5-point Formula
1	1.2153	1.2985	0.55759	1.1611
1/2	0.8744	1.0812	0.8536	1.0088
1/4	0.96051	1.0207	0.95985	1.0005

Table: "Clearly" the centered 3-point formula beats out the backward and forward formulas; but the 5-point formula is big winner here.

Higher Order Derivatives

We can derive approximations for higher order derivatives in the same way. — Fit a kth degree polynomial to a cluster of points $\{x_i, f(x_i)\}_{i=n}^{n+k+1}$, and compute the appropriate derivative of the polynomial in the point of interest.

The standard centered approximation of the second derivative is given by

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} + \mathcal{O}(h^2)$$

Wrapping Up Numerical Differentiation

We now have the tools to build high-order accurate approximations to the derivative.

We will use these tools and similar techniques in building integration schemes in the following lectures.

Also, these approximations are the backbone of finite difference methods for numerical solution of differential equations (*see* `Math 542`, and `Math 693b`).

Next, we develop a general tool for combining low-order accurate approximations (to derivatives, integrals, anything! (almost))... in order to hierarchically constructing higher order approximations.

Richardson's Extrapolation

What it is: A general method for generating high-accuracy results using low-order formulas.

Applicable when: The approximation technique has an error term of predictable form, *e.g.*

$$M - N_j(h) = \sum_{k=j}^{\infty} E_k h^k,$$

where M is the unknown value we are trying to approximate, and $N_j(h)$ the approximation (which has an error $\mathcal{O}(h^j)$.)

Procedure: Use two approximations of the same order, but with *different* h; *e.g.* $N_j(h)$ and $N_j(h/2)$. Combine the two approximations in such a way that the error terms of order h^j cancel.

Building High Accuracy Approximations

Consider two first order approximations to M:

$$M - N_1(h) = \sum_{k=1}^{\infty} E_k h^k,$$

and

$$M - N_1(h/2) = \sum_{k=1}^{\infty} E_k \frac{h^k}{2^k}.$$

If we let $\mathbf{N_2(h) = 2N_1(h/2) - N_1(h)}$, then

$$M - N_2(h) = \underbrace{2E_1\frac{h}{2} - E_1 h}_{0} + \sum_{k=2}^{n} E_k^{(2)} h^k,$$

where

$$E_k^{(2)} = E_k\left(\frac{1}{2^{k-1}} - 1\right).$$

Hence, $N_2(h)$ is now a **second order approximation** to M.

Building High Accuracy Approximations

We can play the game again, and combine $N_2(h)$ with $N_2(h/2)$ to get a third-order accurate approximation, etc.

$$N_3(h) = \frac{4N_2(h/2) - N_2(h)}{3} = N_2(h/2) + \frac{N_2(h/2) - N_2(h)}{3}$$

$$N_4(h) = N_3(h/2) + \frac{N_3(h/2) - N_3(h)}{7}$$

$$N_5(h) = N_4(h/2) + \frac{N_4(h/2) - N_4(h)}{2^4 - 1}$$

In general, combining two jth order approximations to get a $(j+1)$st order approximation:

$$\mathbf{N_{j+1}(h) = N_j(h/2) + \frac{N_j(h/2) - N_j(h)}{2^j - 1}}$$

Building High Accuracy Approximations

Let's derive the general update formula. Given,

$$\begin{aligned} M - N_j(h) &= E_j h^j + \mathcal{O}\left(h^{j+1}\right) \\ M - N_j(h/2) &= E_j \frac{h^j}{2^j} + \mathcal{O}\left(h^{j+1}\right) \end{aligned}$$

We let

$$N_{j+1}(h) = \alpha_j N_j(h) + \beta_j N_j(h/2)$$

However, if we want $N_{j+1}(h)$ to approximate M, we must have $\alpha_j + \beta_j = 1$. Therefore

$$M - N_{j+1}(h) = \alpha_j E_j h^j + (1 - \alpha_j) E_j \frac{h^j}{2^j} + \mathcal{O}\left(h^{j+1}\right)$$

Building High Accuracy Approximations

Now,

$$M - N_{j+1}(h) = E_j h^j \left[\alpha_j + (1-\alpha_j)\frac{1}{2^j} \right] + \mathcal{O}\left(h^{j+1}\right)$$

We want to select α_j so that the expression in the bracket is zero.

This gives

$$\alpha_j = \frac{-1}{2^j - 1}, \qquad 1 - \alpha_j = \frac{2^j}{2^j - 1} = \frac{(2^j - 1) + 1}{2^j - 1} = 1 + \frac{1}{2^j - 1}$$

Therefore,

$$N_{j+1}(h) = N_j(h/2) + \frac{N_j(h/2) - N_j(h)}{2^j - 1}$$

Building High Accuracy Approximations

The following table illustrates how we can use Richardson's extrapolation to build a 5th order approximation, using five 1st order approximations:

$\mathcal{O}(\mathbf{h})$	$\mathcal{O}(\mathbf{h^2})$	$\mathcal{O}(\mathbf{h^3})$	$\mathcal{O}(\mathbf{h^4})$	$\mathcal{O}(\mathbf{h^5})$
$N_1(h)$				
$N_1(h/2)$	$N_2(h)$			
$N_1(h/4)$	$N_2(h/2)$	$N_3(h)$		
$N_1(h/8)$	$N_2(h/4)$	$N_3(h/2)$	$N_4(h)$	
$N_1(h/16)$	$N_2(h/8)$	$N_3(h/4)$	$N_4(h/2)$	$N_5(h)$
$\uparrow$ ***Measurements***	$\uparrow$	***Extrapolations***		$\uparrow$

Example (*c.f.* slide#13, and slide#17)

The centered difference formula approximating $f'(x_0)$ can be expressed:

$$f'(x_0) = \underbrace{\frac{f(x+h) - f(x-h)}{2h}}_{N_2(h)} - \underbrace{\frac{h^2}{6} f'''(\xi) + \mathcal{O}(h^4)}_{\text{error term}}$$

In order to eliminate the h^2 part of the error, we let our new approximation be

$$N_3(h) = N_2(h/2) + \frac{N_2(h/2) - N_2(h)}{3}.$$

$$\begin{aligned} N_3(2h) &= \frac{f(x+h)-f(x-h)}{2h} + \frac{\frac{f(x+h)-f(x-h)}{2h} - \frac{f(x+2h)-f(x-2h)}{4h}}{3} \\ &= \frac{8f(x+h)-8f(x-h)}{6h} - \frac{f(x+2h)-f(x-2h)}{6h} \\ &= \mathbf{\frac{1}{12h}\left[f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)\right]}. \end{aligned}$$

Example, $f(x) = x^2 e^x$.

x	f(x)
1.70	15.8197
1.80	19.6009
1.90	24.1361
2.00	29.5562
2.10	36.0128
2.20	43.6811
2.30	52.7634

$$f'(x) = (2x + x^2)e^x,$$
$$f'(2) = 8e^2 = 59.112.$$

$$\frac{f(2.1)-f(2.0)}{0.1} = 64.566. \text{ (Fwd Difference, 2pt)}$$

$$\frac{f(2.1)-f(1.9)}{0.2} = 59.384. \text{ (Ctr Difference, 3pt)}$$

$$\frac{f(2.2)-f(1.8)}{0.4} = 60.201. \text{ (Ctr Difference)}$$

$$(4 * 59.384 - 60.201)/3 = 59.111. \text{ (Richardson)}$$

$$\frac{f(1.8)-8f(1.9)+8f(2.1)-f(2.2)}{1.2} = 59.111. \text{ (5pt)}$$