Numerical Analysis and Computing

Lecture Notes #7
— Numerical Differentiation and Integration —
Differentiation; Richardson's Extrapolation; Integration

Peter Blomgren,
⟨`blomgren.peter@gmail.com`⟩

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

http://terminus.sdsu.edu/

Fall 2014

Outline

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 $\mathbf{f'(x)}$ (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.
$\Rightarrow$ **smaller h not necessarily better numerically.**

Main Tools for Numerical Differentiation 1 of 2

In the discussion on Numerical Differentiation (and later Integration) we will rely on our old friend (nemesis?) — the Taylor expansions...

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* $\forall x \in (a,b)$, $\exists \xi(x) \in (\min(x_0,x), \max(x_0,x))$ *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 2 of 2

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 $L_{n,0}(x)$, $L_{n,1}(x)$, $\dots$, $L_{n,n}(x)$ to build the *Lagrange interpolating polynomial.* — A good tool for discussing polynomial behavior, but not necessarily for computing polynomial values (*c.f.* Newton's divided differences).

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**.

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

Using Higher Degree Polynomials to get Better Accuracy

Suppose $\{x_0, x_1, \dots, x_n\}$ 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.*

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

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 $f'(x_j)$:

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

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...

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

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

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

Example: 3-point Formulas, III/III

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

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

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

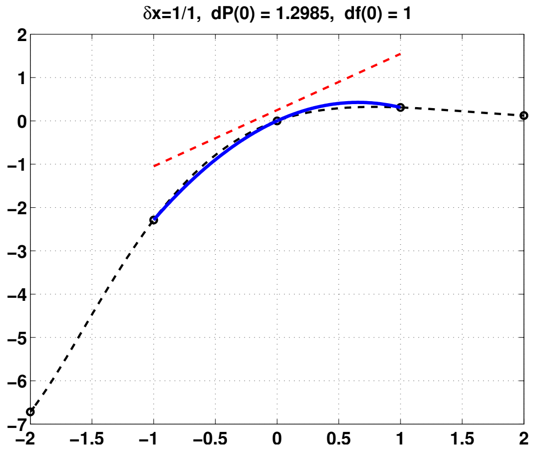
Note#2: The error is smallest in the second equation.

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

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

3-point Formulas: Illustration

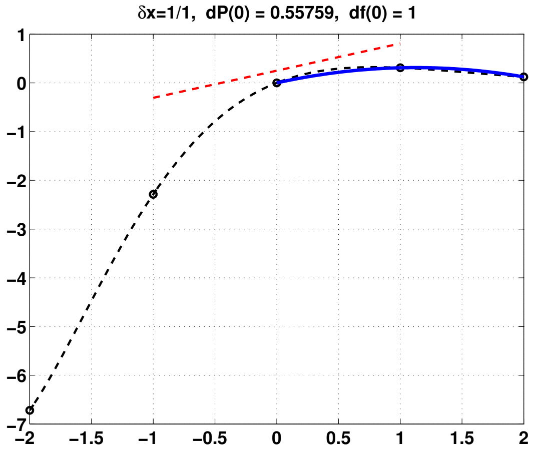
Centered Formula



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

3-point Formulas: Illustration

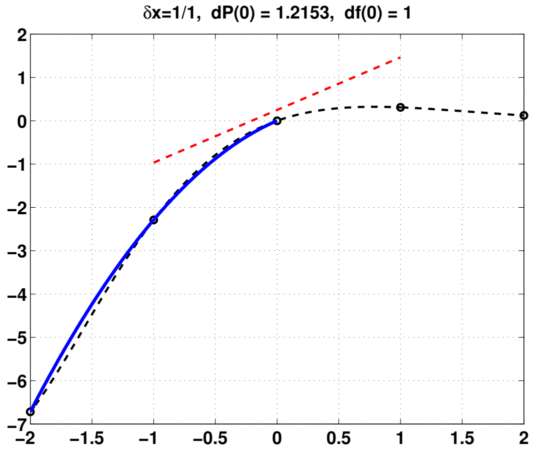
Forward Formula



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

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

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

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

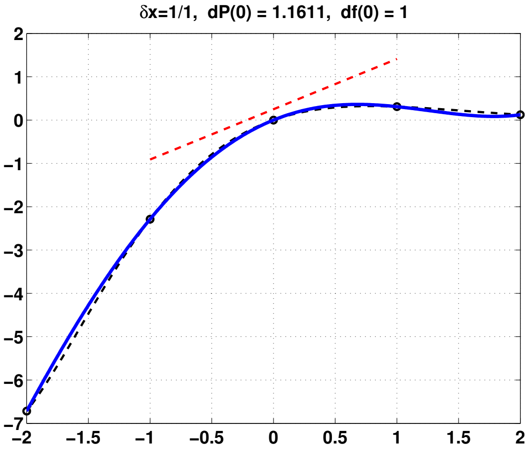
$$f'(x_0) = \frac{1}{12h}\Big[f(x_{-2})-8f(x_{-1})+8f(x_1)-f(x_2)\Big]$$

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

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

5-point Formulas: Illustration

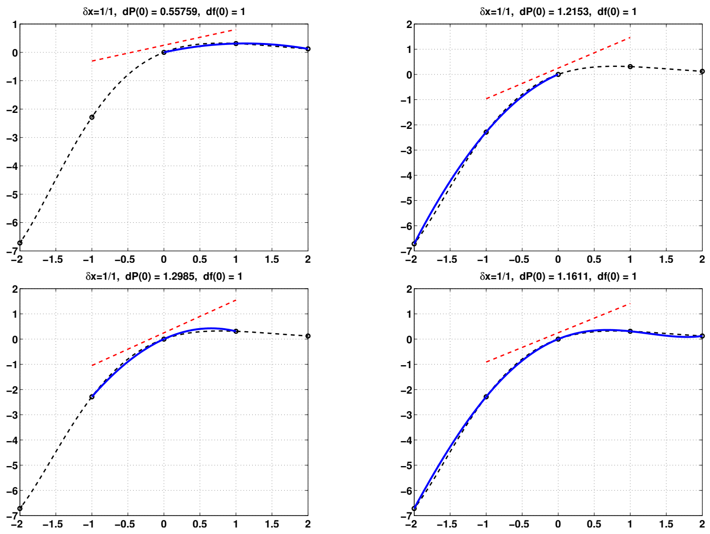
Centered Formula



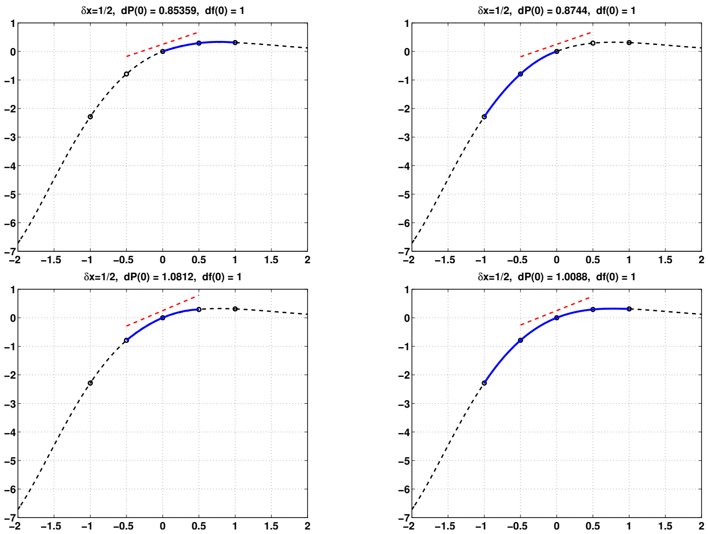
$$f'(x_0) = \frac{f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)}{12h} + \frac{h^4}{30} f^{(5)}(\xi)$$

3-point and 5-point Formulas

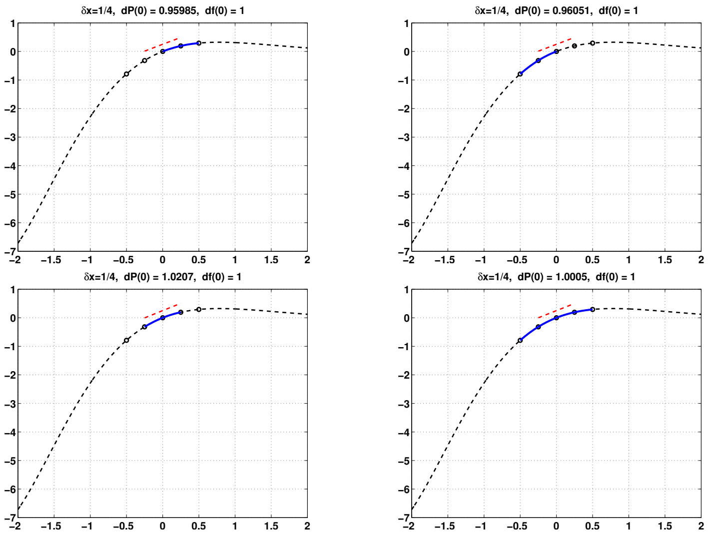
$\delta x = 1$



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

dx	3-Point Formulas			5-point
	Backward	Center	Forward	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

1 of 5

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

2 of 5

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:

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

Building High Accuracy Approximations 3 of 5

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 4 of 5

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

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

Therefore,

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

Building High Accuracy Approximations 5 of 5

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

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

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

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

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

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

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

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

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

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

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

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

Integration: Introduction — The *“Why?”*

After taking calculus, I thought I could differentiate and/or integrate every function...

Then came physics, mechanical engineering, etc...

The need for numerical integration was painfully obvious!

Sometimes (most of the time?), the anti-derivative is not available in closed form.

$$\int f(x)\,dx = \underbrace{F(x)}_{\text{Anti-Derivative}} + \mathcal{C}$$

Numerical Quadrature

The basic idea is to replace integration by clever summation:

$$\int_a^b f(x)\,dx \quad \rightarrow \quad \sum_{i=0}^{n} a_i f_i,$$

where $a \leq x_0 < x_1 < \cdots < x_n \leq b$, $f_i = f(x_i)$.

The coefficients a_i and the nodes x_i are to be selected.

Building Integration Schemes with Lagrange Polynomials

Given the nodes $\{x_0, x_1, \ldots, x_n\}$ we can use the **Lagrange interpolating polynomial**

$$P_n(x) = \sum_{i=0}^{n} f_i L_{n,i}(x), \quad \text{with error} \quad E_n(x) = \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

to obtain

$$\int_a^b f(x)\,dx = \underbrace{\int_a^b P_n(x)\,dx}_{\text{The Approximation}} + \underbrace{\int_a^b E_n(x)\,dx}_{\text{The Error Estimate}}$$

Identifying the Coefficients

$$\int_a^b P_n(x)\,dx = \int_a^b \sum_{i=0}^n f_i L_{n,i}(x)\,dx = \sum_{i=0}^n f_i \underbrace{\int_a^b L_{n,i}(x)\,dx}_{a_i} = \sum_{i=0}^n f_i a_i.$$

Hence we write

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^n a_i f_i$$

with error given by

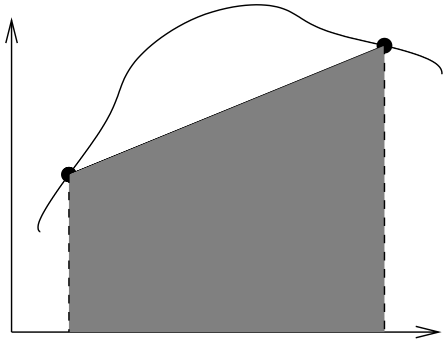
$$E(f) = \int_a^b E_n(x)\,dx = \int_a^b \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^n (x-x_i)\,dx.$$

Note: Can we change the order of integration $\int$ and summation $\sum$ as we did above? In this case where we are integrating a polynomial over a finite interval it is OK. For technical details see a class on real analysis (*e.g.* Math 534B).

Example #1: Trapezoidal Rule I/III

Let $a = x_0 < x_1 = b$, and use the linear interpolating polynomial

$$P_1(x) = f_0 \left[\frac{x-x_1}{x_0-x_1}\right] + f_1 \left[\frac{x-x_0}{x_1-x_0}\right].$$



Example #1: Trapezoidal Rule II/III

Then

$$\begin{aligned}\int_a^b f(x)\,dx &= \int_{x_0}^{x_1} \left[f_0 \left[\frac{x-x_1}{x_0-x_1}\right] + f_1 \left[\frac{x-x_0}{x_1-x_0}\right]\right] dx \\ &\quad + \frac{1}{2}\int_{x_0}^{x_1} f''(\xi(x))(x-x_0)(x-x_1)\,dx.\end{aligned}$$

The error term (use the Weighted Mean Value Theorem):

$$\int_{x_0}^{x_1} f''(\xi(x))(x-x_0)(x-x_1)\,dx = f''(\xi)\int_{x_0}^{x_1}(x-x_0)(x-x_1)\,dx$$

$$= f''(\xi)\left[\frac{x^3}{3} - \frac{x_1+x_0}{2}x^2 + x_0x_1x\right]_{x_0}^{x_1} = -\frac{h^3}{6}f''(\xi).$$

where $h = x_1 - x_0 = b - a$.

Example #1: Trapezoidal Rule III/III

Hence,

$$\int_a^b f(x)\,dx = \left[f_0\left[\frac{(x-x_1)^2}{2(x_0-x_1)}\right] + f_1\left[\frac{(x-x_0)^2}{2(x_1-x_0)}\right]\right]_{x_0}^{x_1} - \frac{h^3}{12}f''(\xi)$$

$$= \frac{(x_1-x_0)}{2}\,[f_0+f_1] - \frac{h^3}{12}f''(\xi)$$

$$\boxed{\int_a^b \mathbf{f(x)\,dx} = \mathbf{h}\left[\frac{\mathbf{f(x_0)+f(x_1)}}{\mathbf{2}}\right] - \frac{\mathbf{h^3}}{\mathbf{12}}f''(\xi), \qquad h = b-a.}$$

Example #2a: Simpson's Rule (sub-optimal error bound)

Let $x_0 = a$, $x_1 = \frac{a+b}{2}$, $x_2 = b$, let $h = \frac{b-a}{2}$ and use the **quadratic interpolating polynomial**

$$\int_a^b f(x)dx = \int_{x_0}^{x_2} \left[f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \right.$$

$$\left. + f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \right] dx$$

$$+ \int_{x_0}^{x_2} \frac{(x-x_0)(x-x_1)(x-x_2)}{6} f^{(3)}(\xi(x))\, dx \ \ldots$$

$$\boxed{\int_a^b \mathbf{f(x)\,dx} = \mathbf{h}\left[\frac{\mathbf{f(x_0) + 4f(x_1) + f(x_2)}}{\mathbf{3}}\right] + \mathcal{O}(\mathbf{h^4 f^{(3)}}(\xi)).}$$

Example #2b: Simpson's Rule (optimal error bound)

The optimal error bound for Simpson's rule can be obtained by Taylor expanding $f(x)$ about the mid-point x_1:

$$f(x) = f(x_1) + f'(x_1)(x-x_1) + \frac{f''(x_1)}{2}(x-x_1)^2 + \frac{f'''(x_1)}{6}(x-x_1)^3 + \frac{f^{(4)}(\xi(x))}{24}(x-x_1)^4,$$

then formally integrating this expression, to get:

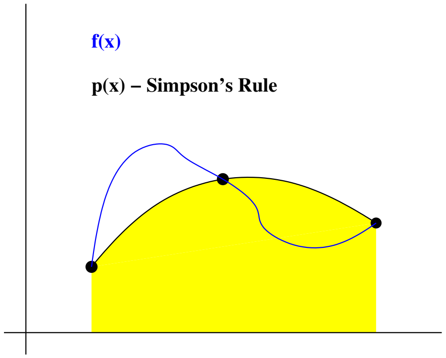
$$\int_a^b \left[f(x_1) + f'(x_1)(x-x_1) + \frac{f''(x_1)}{2}(x-x_1)^2 + \frac{f'''(x_1)}{6}(x-x_1)^3 + \frac{f^{(4)}(\xi(x))}{24}(x-x_1)^4 \right] dx.$$

After use of the weighted mean value theorem, and the approximation $f''(x_1) = \frac{1}{h^2}[f(x_0) - 2f(x_1) + f(x_2)] - \frac{h^2}{12}f^{(4)}(\xi)$, and a whole lot of algebra (see BF $^{\text{8th / 9th}}$ pp. 189–190 / 195–196) we end up with

$$\int_{x_0}^{x_2} f(x)\,dx = \mathbf{h}\left[\frac{\mathbf{f(x_0) + 4f(x_1) + f(x_2)}}{\mathbf{3}}\right] - \frac{\mathbf{h^5}}{\mathbf{90}}\mathbf{f^{(4)}}(\xi).$$

Example #2: Simpson's Rule

$$\boxed{\int_a^b \mathbf{f(x)\,dx} = \mathbf{h}\left[\frac{\mathbf{f(x_0) + 4f(x_1) + f(x_2)}}{\mathbf{3}}\right] + \mathcal{O}(\mathbf{h^5 f^{(4)}}(\xi)).}$$



Integration Examples

$f(x)$	$[a,b]$	$\int_a^b f(x)dx$	Trapezoidal	Error	Simpson	Error
x	$[0,1]$	1/2	0.5	0	0.5	0
x^2	$[0,1]$	1/3	0.5	0.16667	0.33333	0
x^3	$[0,1]$	1/4	0.5	0.25000	0.25000	0
x^4	$[0,1]$	1/5	0.5	0.30000	0.20833	0.0083333
e^x	$[0,1]$	$e-1$	1.8591	0.14086	1.7189	0.0005793

The Trapezoidal rule gives exact solutions for linear functions. — The error terms contains a second derivative.

Simpson's rule gives exact solutions for polynomials of degree less than 4. — The error term contains a fourth derivative.

Degree of Accuracy (Precision)

Definition (Degree of Accuracy)

The **Degree of Accuracy**, or **precision**, of a quadrature formula is the largest positive integer n such that the formula is exact for x^k $\forall k = 0, 1, \dots, n$.

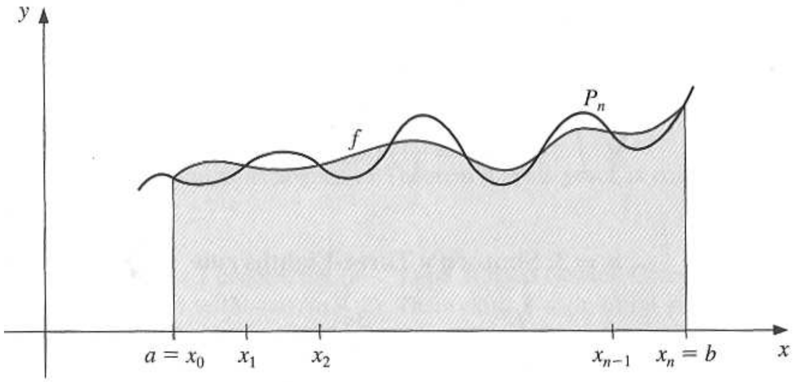
With this definition:

Scheme	Degree of Accuracy
Trapezoidal	1
Simpson's	3

Trapezoidal and Simpson's are examples of a class of methods known as **Newton-Cotes formulas**.

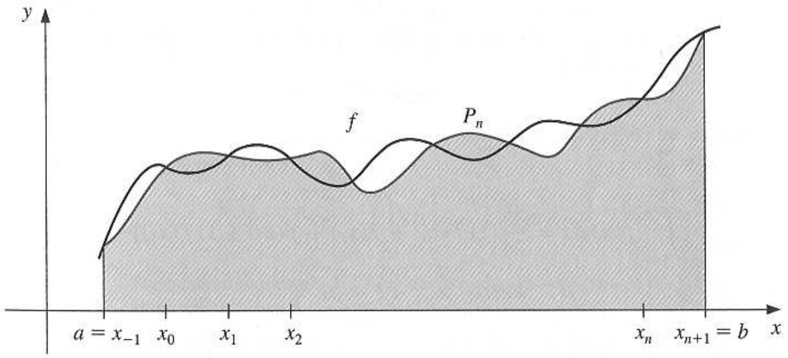
Newton-Cotes Formulas — Two Types Closed

Closed The $(n+1)$ point closed NCF uses nodes $x_i = x_0 + ih$, $i = 0, 1, \dots, n$, where $x_0 = a$, $x_n = b$ and $h = (b-a)/n$. It is called closed since the endpoints are included as nodes.



Newton-Cotes Formulas — Two Types Open

Open The $(n+1)$ point open NCF uses nodes $x_i = x_0 + ih$, $i = 0, 1, \dots, n$ where $h = (b-a)/(n+2)$ and $x_0 = a + h$, $x_n = b - h$. (We label $x_{-1} = a$, $x_{n+1} = b$.)



Closed Newton-Cotes Formulas

The approximation is

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} a_i f(x_i),$$

where

$$a_i = \int_{x_0}^{x_n} L_{n,i}(x)\,dx = \int_{x_0}^{x_n} \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}\,dx.$$

Note: The Lagrange polynomial $L_{n,i}(x)$ models a function which takes the value 0 at all x_j $(j \neq i)$, and 1 at x_i. Hence, the coefficient a_i captures the integral of a function which is 1 in x_i and zero in the other node points.

Closed Newton-Cotes Formulas — Error

Theorem

Suppose that $\sum_{i=0}^{n} a_i f(x_i)$ *denotes the* $(n+1)$ *point closed Newton-Cotes formula with* $x_0 = a$, $x_n = b$, *and* $h = (b-a)/n$. *Then there exists* $\xi \in (a,b)$ *for which*

$$\int_a^b f(x)dx = \sum_{i=0}^{n} a_i f(x_i) + \frac{\mathbf{h^{n+3}f^{(n+2)}}(\xi)}{(n+2)!}\int_0^n t^2(t-1)\cdots(t-n)dt,$$

if n *is even and* $f \in C^{n+2}[a,b]$, *and*

$$\int_a^b f(x)dx = \sum_{i=0}^{n} a_i f(x_i) + \frac{\mathbf{h^{n+2}f^{(n+1)}}(\xi)}{(n+1)!}\int_0^n t(t-1)\cdots(t-n)dt,$$

if n *is odd and* $f \in C^{n+1}[a,b]$.

Note that when n is an even integer, the degree of precision is $(n+1)$. When n is odd, the degree of precision is only n.

Closed Newton-Cotes Formulas — Examples

n = 2: Simpson's Rule

$$\frac{h}{3}\Big[f(x_0) + 4f(x_1) + f(x_2)\Big] - \frac{h^5}{90}f^{(4)}(\xi)$$

n = 3: Simpson's $\frac{3}{8}$-Rule

$$\frac{3h}{8}\Big[f(x_0) + 3f(x_1) + 3f(x_2) + f(x_3)\Big] - \frac{3h^5}{80}f^{(4)}(\xi)$$

n = 4: Boole's Rule

$$\frac{2h}{45}\Big[7f(x_0) + 32f(x_1) + 12f(x_2) + 32f(x_3) + 7f(x_4)\Big] - \frac{8h^7}{945}f^{(6)}(\xi)$$

Open Newton-Cotes Formulas

The approximation is

$$\int_a^b f(x)\,dx = \int_{x_{-1}}^{x_{n+1}} f(x)\,dx \approx \sum_{i=0}^{n} a_i f(x_i),$$

where

$$a_i = \int_{x_{-1}}^{x_{n+1}} L_{n,i}(x)\,dx = \int_{x_0}^{x_n} \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)}\,dx.$$

Open Newton-Cotes Formulas — Error

Theorem

Suppose that $\sum_{i=0}^{n} a_i f(x_i)$ *denotes the* $(n+1)$ *point open Newton-Cotes formula with* $x_{-1} = a$, $x_{n+1} = b$, *and* $h = (b-a)/(n+2)$. *Then there exists* $\xi \in (a,b)$ *for which*

$$\int_a^b f(x)dx = \sum_{i=0}^{n} a_i f(x_i) + \frac{h^{n+3}f^{(n+2)}(\xi)}{(n+2)!}\int_{-1}^{n+1} t^2(t-1)\cdots(t-n)dt,$$

if n *is even and* $f \in C^{n+2}[a,b]$, *and*

$$\int_a^b f(x)dx = \sum_{i=0}^{n} a_i f(x_i) + \frac{h^{n+2}f^{(n+1)}(\xi)}{(n+1)!}\int_{-1}^{n+1} t(t-1)\cdots(t-n)dt,$$

if n *is odd and* $f \in C^{n+1}[a,b]$.

Note that when n is an even integer, the degree of precision is $(n+1)$. When n is odd, the degree of precision is only n.

Open Newton-Cotes Formulas — Examples

n = 0 : $$2hf(x_0) + \frac{h^3}{3} f''(\xi)$$

n = 1 : $$\frac{3h}{2}\Big[f(x_0) + f(x_1)\Big] + \frac{3h^3}{4} f''(\xi)$$

n = 2 : $$\frac{4h}{3}\Big[2f(x_0) - f(x_1) + 2f(x_2)\Big] + \frac{14h^5}{45} f^{(4)}(\xi)$$

n = 3 : $$\frac{5h}{24}\Big[11f(x_0) + f(x_1) + f(x_2) + 11f(x_3)\Big] + \frac{95h^5}{144} f^{(4)}(\xi)$$

Divide and Conquer!

Say you want to compute:

$$\int_0^{100} f(x)\,dx.$$

Is it a Good Idea$^{\text{TM}}$ to directly apply your favorite Newton-Cotes formula to this integral?!?

No!

With the closed 5-point NCF, we have $h = 25$ and $h^5/90 \sim 10^5$ so even with a bound on $f^{(6)}(\xi)$ the error will be large.

Better: Apply the closed 5-point NCF to the integrals

$$\int_{4i}^{4(i+1)} f(x)\,dx, \quad i = 0, 1, \ldots, 24$$

then sum. **"Composite Numerical Integration."** (next time)

Homework #6 `http://webwork.sdsu.edu`

- Will open on 10/15/2014 at 09:30am PDT.
- Will close no earlier than 10/24/2014 at 09:00pm PDT.