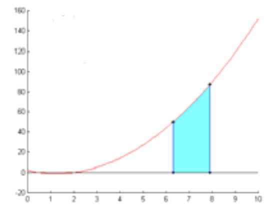
Numerical Integration

Ch 5 of Turner

CSS455 Winter 2012

Definite Integrals as Areas



$\int_a^b f(x)dx$ = area under f(x) from x = a to x = b.

Recall discussions in elementary calculus where you considered the definite integral of a function between two limits as the area under the curve of that function.

Approximate Numerical Method Interpolatory Quadrature

- First idea: approximate the function *f(x)* by an interpolant polynomial *p(x)* and then integrate the interpolant exactly.
- Second idea: Express the result as a sum over weighted values of *f(x)* at fixed predetermined values of *x*: an exact formula for the integral of *p*(*x*).

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = (b-a)\sum_{k=0}^{N-1} c_k f(x_k)$$

Weights are fixed for all quadratures of this type.

Constant for the integral limits.

x_k distributed evenly over the interval (a,b): *f* (x) is evaluated only at these points

---

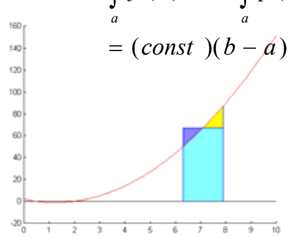
$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \sum_{k=0}^{2} c_k f(x_k)$$

- This is an exact formula for the integral of *p*(*x*).
- If *f(x)* is a polynomial of 2nd order or less, then *p*(*x*) is an exact fit and the integral of *f*(*x*) is exact: Degree of precision here is at least m=2.
- Exact formula for *f(x)* = 1, *x, or* x^2
- Since the *c's* are the same *regardless of the form of f(x),* we can write three independent equations for them, one based on each choice of *f(x).*
- *This will provide another general way to obtain the values of the weights.*

Zero order polynomial: Assume the area is equal to that of the rectangle formed by the line at the midpoint of the region.

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \int_a^b (const)\,dx =$$

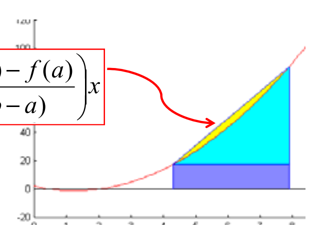
$$= (const)(b-a) = f\left(\tfrac{a+b}{2}\right)\cdot(b-a)$$



Interpolation is of zero order.

Trapezoid Approx: Assume the area is equal to that of the trapezoid formed by the line between the limits.

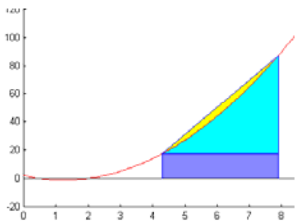
$$p(x) = f(a) + \left(\frac{f(b)-f(a)}{(b-a)}\right)x$$



$$\int_a^b f(x)dx \approx \int_a^b p(x)dx =$$

$$= f(a)\cdot(b-a) + \tfrac{1}{2}(b-a)\cdot[f(b)-f(a)]$$

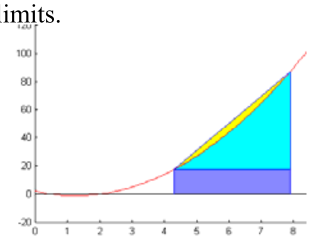
Trapezoid Approx: Assume the area is equal to that of the trapezoid formed by the line between the limits.



There is no cancellation of error here.

$$\int_a^b f(x)dx \approx f(a)\cdot(b-a)+\tfrac{1}{2}(b-a)\cdot[f(b)-f(a)]$$

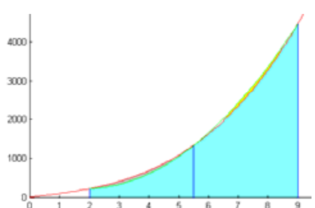
Trapezoidal Rule: Assume the area is equal to that of the trapezoid formed by the line between the limits.



Interpolant is of order one..

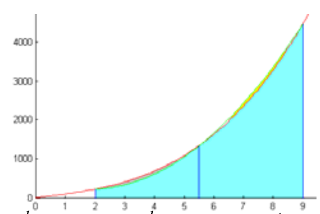
$$\int_a^b f(x)dx \approx \tfrac{1}{2}(b-a)\cdot[f(a)+f(b)]$$

Simpson's Rule: Assume the area is formed by a parabola interpolating the limits and midpoint.



$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \int_a^b \left[\alpha+\beta x+\gamma x^2\right]dx$$

Simpson's Rule: Assume the area is formed by a parabola interpolating the limits and midpoint.



Interpolant is of order two. There is cancellation of error.

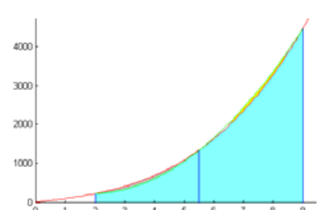
$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \left(\alpha x+\frac{\beta}{2}x^2+\frac{\gamma}{3}x^3\right]_a^b$$

$$\int_a^b p(x)dx = \int_a^b \left(\alpha+\beta x+\gamma x^2\right)dx = \alpha x+\frac{\beta}{2}x^2+\frac{\gamma}{3}x^3\Big|_a^b$$

- The coefficients $\{\alpha,\beta,\gamma\}$ depend upon the x_k and $f(x_k)$ values $\left\{a, f(a), b, f(b), \frac{a+b}{2}, f\left(\frac{a+b}{2}\right)\right\}$

The integral can be expressed in terms of these three data points (nodes) and the limits [a b].

Simpson's Rule: Assume the area is formed by a parabola interpolating the limits and midpoint.



This can be shown to be true by algebraic substitution

$$\int_a^b f(x)dx \approx \tfrac{1}{6}(b-a)\cdot\left[f(a)+4f\left(\tfrac{a+b}{2}\right)+f(b)\right]$$

Activity 12:

- For n=1,2,3 (linear,quadratic, cubic) degree polynomials you found that the integral of the interpolant took the form:

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \sum_{k=0}^{2} c_k f(x_k)$$

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \sum_{k=0}^{2} c_k f(x_k)$$

- Where the c_k's depend *only* upon the positions of the nodes: the actual values of x_k. These c_k's are called the "weights," and this procedure for integration is called "quadrature."

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx = \sum_{k=0}^{2} c_k f(x_k)$$

- This is an exact formula for the integral of $p(x)$.
- If $f(x)$ is a polynomial of 2nd order or less, then $p(x)$ is an exact fit and the integral of $f(x)$ is exact: Degree of precision here is at least m=2.
- Exact formula for $f(x) = 1, x,$ or x^2
- Since the *c's* are the same *regardless of the form of f(x),* we can write three independent equations for them, each one based on a special choice of *f(x).*
- *This will provide another general way to obtain the values of the weights.*

Three Eqns and Three unknowns

Select the three points $x = \left\{a, \frac{a+b}{2}, b\right\}$

To generate the first equation, choose $f(x) = 1$:

$$\int_a^b f(x)dx = \int_a^b (1)dx = x\Big|_a^b = b-a$$

$$\int_a^b f(x)dx = c_0 f(x_0) + c_1 f(x_1) + c_2 f(x_2) = c_0 + c_1 + c_2$$

$$c_0 + c_1 + c_2 = b - a, \quad \textbf{(eq 1)}$$

Equation 2: for $f(x) = x$ $\quad x = \left\{a, \frac{a+b}{2}, b\right\}$

$$\int_a^b f(x)dx = \int_a^b (x)dx = \tfrac{1}{2}x^2\Big|_a^b = \left(b^2 - a^2\right)\!\big/2$$

$$\int_a^b f(x)dx = c_0 f(x_0) + c_1 f(x_1) + c_2 f(x_2) = c_0 a + c_1\left(\tfrac{a+b}{2}\right) + c_2 b$$

$$ac_0 + \left(\tfrac{a+b}{2}\right)c_1 + bc_2 = \frac{b^2 - a^2}{2}, \quad \textbf{(eq 2)}$$

$$x = \left\{a, \frac{a+b}{2}, b\right\}$$

Equation #3: for $f(x) = x^2$

$$\int_a^b f(x)dx = \int_a^b x^2 dx = \tfrac{1}{3}x^3\Big|_a^b = \left(b^3 - a^3\right)\!\big/3$$

$$\int_a^b f(x)dx = c_0 f(x_0) + c_1 f(x_1) + c_2 f(x_2) = c_0 a^2 + c_1\left(\tfrac{a+b}{2}\right)^2 + c_2 b^2$$

$$a^2 c_0 + \left(\tfrac{a+b}{2}\right)^2 c_1 + b^2 c_2 = \frac{b^3 - a^3}{3}, \quad \textbf{(eq 3)}$$

System of three eqns

$$\begin{bmatrix} 1 & 1 & 1 \\ a & \frac{a+b}{2} & b \\ a^2 & \left(\frac{a+b}{2}\right)^2 & b^2 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} b-a \\ \left(b^2-a^2\right)/2 \\ \left(b^3-a^3\right)/3 \end{pmatrix}$$

Solve by methods of Chapter 7 or by explicit algebraic elimination.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} (b-a)/6 \\ 4(b-a)/6 \\ (b-a)/6 \end{pmatrix}$$

These are the weights we presented for Simpson's Rule, obtained by integrating the parabola fit to the three points.

System of three eqns

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} (b-a)/6 \\ 4(b-a)/6 \\ (b-a)/6 \end{pmatrix}$$

$$\int_b^b f(x)dx \approx \left(f(a) \quad f\left(\tfrac{a+b}{2}\right) \quad f(b)\right) \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix}$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right) f(a) + \left(\frac{4(b-a)}{6}\right) f\left(\tfrac{a+b}{2}\right) + \left(\frac{b-a}{6}\right) f(b)$$

Claim: exact for general quadratic f(x) (precise for m=2)

$$f(x) = \left(C + Dx + Ex^2\right)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right) f(a) + \left(\frac{4(b-a)}{6}\right) f\left(\tfrac{a+b}{2}\right) + \left(\frac{b-a}{6}\right) f(b)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right) f(a) + \left(\frac{4(b-a)}{6}\right) f\left(\tfrac{a+b}{2}\right) + \left(\frac{b-a}{6}\right) f(b)$$

To check for f(x) = C

$$\int_a^b Cdx = Cx\Big|_a^b = C(b-a)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right)\left(f(a) + 4f\left(\tfrac{a+b}{2}\right) + f(b)\right) =$$

$$= \left(\frac{b-a}{6}\right)(6C) = C(b-a)$$

Precise for m=0

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right) f(a) + \left(\frac{4(b-a)}{6}\right) f\left(\tfrac{a+b}{2}\right) + \left(\frac{b-a}{6}\right) f(b)$$

To check for f(x) = C + Dx

Precise for m=1

$$\int_a^b (C + Dx)dx = Cx\Big|_a^b + \tfrac{D}{2}x^2\Big|_a^b = C(b-a) + \frac{D}{2}\left(b^2 - a^2\right)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right)\left(f(a) + 4f\left(\tfrac{a+b}{2}\right) + f(b)\right) =$$

$$= \left(\frac{b-a}{6}\right)\left(C + Da + 4\left(C + \tfrac{D}{2}(a+b)\right) + C + Db\right) =$$

$$= \left(\frac{b-a}{6}\right)\left(6C + D(3a+3b)\right) = C(b-a) + \frac{D}{2}(b-a)(b+a)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right) f(a) + \left(\frac{4(b-a)}{6}\right) f\left(\tfrac{a+b}{2}\right) + \left(\frac{b-a}{6}\right) f(b)$$

To check for f(x) = C + Dx + Ex^2

$$\int_a^b \left(C + Dx + Ex^2\right)dx = Cx\Big|_a^b + \tfrac{D}{2}x^2\Big|_a^b + \tfrac{E}{3}x^3\Big|_a^b =$$

$$= C(b-a) + \frac{D}{2}\left(b^2 - a^2\right) + \frac{E}{3}\left(b^3 - a^3\right)$$

$$\int_a^b \left(C + Dx + Ex^2\right)dx = C(b-a) + \frac{D}{2}\left(b^2 - a^2\right) + \frac{E}{3}\left(b^3 - a^3\right)$$

Exact integral

Simpson rule integral

$$\int_a^b \left(C + Dx + Ex^2\right)dx = C(b-a) + \frac{D}{2}\left(b^2 - a^2\right) + \frac{E}{3}\left(b^3 - a^3\right)$$

$$\int_b^b f(x)dx \approx \left(\frac{b-a}{6}\right)\left(f(a) + 4f\left(\tfrac{a+b}{2}\right) + f(b)\right) =$$

$$= \left(\frac{b-a}{6}\right)\left(C + Da + Ea^2 + 4\left(C + \tfrac{D}{2}(a+b) + \tfrac{E}{4}(a+b)^2 + \right) + C + Db + Eb^2\right) =$$

$$= \left(\frac{b-a}{6}\right)\left(6C + D(3a+3b) + 2E\left(a^2 + b^2 + ab\right)\right) =$$

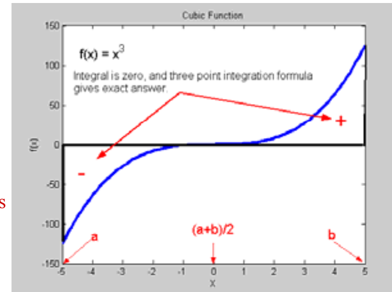
$$= C(b-a) + \tfrac{D}{2}(b-a)(b+a) + \tfrac{E}{3}(b-a)\left(a^2 + b^2 + ab\right)$$

Exact for m=2

What about f(x) cubic?

•The three-point quadratic interpolant p(x) cannot represent f(x) exactly.

•But the integration rule (Simpson's) does integrate f(x) exactly. Why?



Newton-Cotes Rules

- In each case the set of m uniformly spaced points is fit by an m-1 order interpolant p_{m-1}.
- The integral of that interpolant between the limits is the Newton-Cotes m-point rule for the approximation of the exact integral.

$$\int_a^b f(x)dx \approx \int_a^b p_{m-1}(x)dx = Q_{NC(m)}$$

Newton - Cotes rules

m	$Q_{NC(m)}$
1	$(b-a)f\left(\frac{a+b}{2}\right)$ *(mid-point)*
2	$\left(\frac{b-a}{2}\right)\left(f(a) + f(b)\right)$ *(trapezoid)*
3	$\left(\frac{b-a}{6}\right)\left(f(a) - 4f\left(\frac{a+b}{2}\right) + f(b)\right)$ *(Simpson's Rule)*
m	$(b-a)\sum_{k=0}^{m-1} c_k f(x_k)$ Dot product

NC Demos

- Look at WNC.m, which provides the weights.
- Then look at NCCosine.m which integrates for several Newton Cotes Rules:

$$\int_0^{\pi/2} \cos(x)dx = \sin(x)\Big|_0^{\pi/2} = 1$$

NC Demos

- Then look at NCCosSqrt.m which integrates for several Newton Cotes Rules:

$$\int_0^{\pi/2} \left(\cos(x) - \frac{1}{\sqrt{x + 0.01}}\right)dx =$$

$$= (\sin(x) - 2.*\mathrm{sqrt}(x + 0.010)\Big|_0^{\pi/2} =$$

$$= -1.314594462$$

NC Demos

- Then look at NCHumps.m which integrates for several Newton Cotes Rules:

$$\int_0^3 \left(\frac{1}{(x-0.3)^2+0.01} + \frac{1}{(x-0.9)^2+0.04} - 6 \right) dx =$$

$$= (-6x + 10\arctan(10x - 3.) + 5\arctan(5x - 4.5))\Big|_0^3 =$$

$$= 23.96807984$$

- Accuracy is a problem with a less well-suited function such as *humps(x)*.

```
clc
clear all
close all
%Newton Cotes integration
%Range is 0 to pi/2
fname = 'humps';
a=0;
b=3;
mlim=11;
for m = 2:mlim
    c = WNC(m);
    fx = zeros(1,m);
    x = linspace(a,b,m);
    fx = feval(fname,x);
    integ=(b-a)*(fx*c);
    disp (sprintf(' %2.0f points Int= %20.16f',m,integ))
end
```

2 points Int= -0.6927360139068717
3 points Int= -5.8516016598080371
4 points Int= 12.3647470310060380
5 points Int= 8.9068659991608801
6 points Int= 4.1083984982332478
7 points Int= 8.7613384746255925
8 points Int= 20.3119686539944090
9 points Int= 37.1667262265332570
10 points Int= 49.3473459511422020
11 points Int= 65.5474191610827010

Newton-Cotes Rules

- Error bounds can be calculated and turn out to be proportional to the size of the interval $(b-a)^{d+2}$, the magnitude of the $d+1$ derivative, and inversely proportional to the order of the interpolant $(m-1)^{d+2}$. *(d=m or m-1)*
- Formulas on page 127: need estimate of derivatives bound to be useful.

Page 127 error formulas

$$\int_a^b f(x)dx - (b-a)f((a+b)/2) = \frac{(b-a)h^2}{24} f''(\xi_M) \qquad (5.10)$$

where $h = b - a$,

$$\int_a^b f(x)dx - \frac{b-a}{2}[f(a) + f(b)] = -\frac{(b-a)h^2}{12} f''(\xi_T) \qquad (5.11)$$

where again $h = b - a$, and

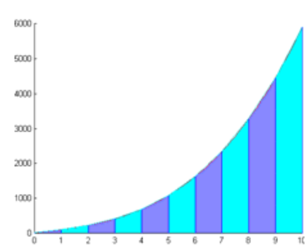
$$\int_a^b f(x)dx - \frac{h}{3}[f(a) + 4f(a+h) + f(b)] = -\frac{(b-a)h^4}{180} f^{(4)}(\xi_S) \qquad (5.12)$$

where $h = (b-a)/2$. Here the points ξ_M, ξ_T, ξ_S are 'mean value points' in the interval (a, b).

Improve Accuracy

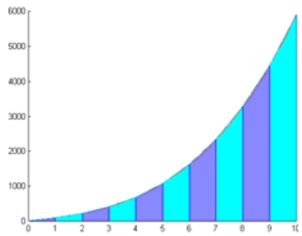
- **Gaussian quadrature:** choose positions of nodes and weights to give maximum precision: order is much higher. m nodes can yield 2m-1 order precision on the interval [-1, 1].
- **Composite Rules:** To obtain high accuracy, divide the integral up into small regions that can be *individually* treated with accuracy (at reasonable values of *m*)
- The contributions are then summed.

Composite Rules



- The overall error is now also **inversely** proportional to *N.*
- The error can be recursively reduced by increasing the number of intervals used.
- Greater improvement can be achieved with adaptive spacing of the intervals.

Composite Rules



Activity 13, Part I

Consider Simpson errors (m=3)

For the simple (single panel) case, it can be shown that the error is given by:

$$E(S_1) = -\tfrac{(b-a)h^4}{180} f^{[4]}(\xi_2) \text{ where } h = {(b\text{-}a)}/{2}$$

For the N-panel case (N, m=3), the error approximation can be shown to be:

$$E(S_N) = -\tfrac{(b-a)(h)^4}{180} f^{[4]}(\xi_2) \text{ where } h = {(b-a)}/{2N}$$

Consider Simpson errors (m=3)

For the N-panel case (N, m=3), the error approximation can be shown to be:

$$E(S_N) = -\tfrac{(b-a)(h)^4}{180} f^{[4]}(\xi_2) \text{ where } h = {(b-a)}/{2N}$$

$$E(S_{2N}) = -\tfrac{(b-a)(h/2)^4}{180} f^{[4]}(\xi_2) \text{ where } h = {(b-a)}/{2N}$$

$$E_{2N} \approx \left(\frac{1}{16}\right) E_N$$

$$E_{2N} \approx \left(\frac{1}{16}\right) E_N$$

$$16\left(I_{exact} - S_{2N}\right) \approx \left(I_{exact} - S_N\right)$$

$$15\left(I_{exact} - S_{2N}\right) \approx S_{2N} - S_N$$

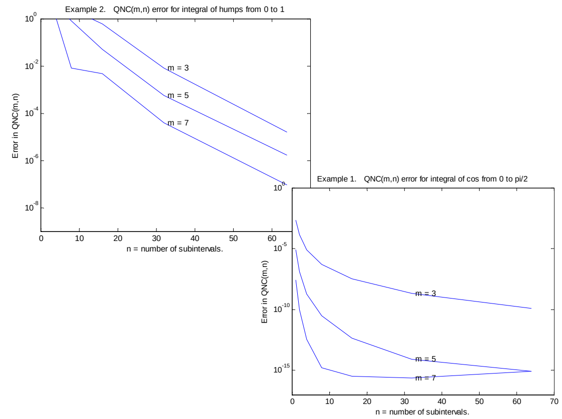
$$\left(I_{exact} - S_{2N}\right) \approx {\left(S_{2N} - S_N\right)}/{15}$$

Review single panel results

Run NCCos and NChumps demos.

Activity 13 part II

Activity 13 part III



Adaptive Composite rules

- Distribute the nodes unevenly to improve precision, instead of just decreasing them uniformly with each iteration.
- How would you decide how to do that? *Think of an iterative procedure…*

Matlab Integration of *humps* with adaptive procedure *Quad (m=3)*.

```
[int,count] = quad(fname,a,b,tol,1);
```

Improper Integrals

- Divide the improper integral into two pieces, one of which is to be neglected and the other can be evaluated numerically:

$$I = \int_a^\infty f(x)dx = \int_a^b f(x)dx + \int_b^\infty f(x)dx$$

•Divide up the acceptable error ε in half and find a value "b" that gives a bound for the second integral that is ≤ ε/2.

•Evaluate the first integral with an error bound of ε/2

Numerical Differention

- Recall the fundamental definition:

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

This would be true for negative or positive *h*.

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

Activity: Part IV

- With your partner, determine the approximate value of the derivative of *cos(x)* at *x*= pi/2, and calculate the error for the values of *h assigned to your team.*

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

Exact answer: f'(pi/2) = -1.0

Finite Difference Approximation

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h} = f[x_0, x_0+h]$$

- Two sources of error:
  - Inherent in using finite difference approx: proportional to magnitude of *h*
  - Numerical due to subtraction of two similar terms: inversely proportional to magnitude of *h*

Trade off here!

Truncation or calculus error

- Expand $f(x_0+h)$ in Taylor Series

$$f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{3!}f'''(x_0) + \ldots$$

$$f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(\theta)$$

Solve for derivative:

$$f'(x_0) = \frac{f(x_0+h)-f(x_0)}{h} - \frac{h}{2}f''(\theta)$$

1st order accurate

2nd derivative gives error magnitude

Numerical error

- Similar error in $f(x_0)$ and $f(x_0+h)$: δ

$$\hat{f}(x_0) - f(x_0) \approx \delta$$

$$\hat{f}(x_0+h) - f(x_0+h) \approx \delta$$

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

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

Total computational error in finite difference: $2\delta/h$

$$f'(x_0) - \hat{f}'(x_0) \approx \frac{\delta+\delta}{h} = \frac{2\delta}{h}$$

Total Error

- $E = ch + 2\delta/h$ $\{c = (1/2)f''(\theta)\}$
- Minimum error found by differentiating:
  - $dE/dh = c - 2\delta/h^2 = 0$
  - Solution $h = \sqrt{2\delta / c}$

•There is a value of *h* for which the error in this approximation to the derivative is a minimum.

•Need to form estimate *c* of truncation error

Other divided difference Formulae

- Negative (backward) divided difference

$$f'(x_0) \approx \frac{f(x_0)-f(x_0-h)}{h} = f[x_0-h, x_0]$$

Average of positive and negative: Two-sided difference

$$f'(x_0) \approx \tfrac{1}{2}\left(\frac{f(x_0)-f(x_0-h)}{h} + \frac{f(x_0+h)-f(x_0)}{h}\right)$$

$$f'(x_0) \approx \left(\frac{f(x_0+h)-f(x_0-h)}{2h}\right)$$

Two-sided error is smaller

- Taylor Series:

$$f(x_0+h)=f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\frac{h^3}{3!}f'''(x_0)+\ldots$$

$$f(x_0-h)=f(x_0)-hf'(x_0)+\frac{h^2}{2}f''(x_0)-\frac{h^3}{3!}f'''(x_0)+\ldots$$

Subtract second from first:

$$f(x_0+h)-f(x_0-h)=2hf'(x_0)+\frac{2h^3}{3!}f'''(x_0)+\ldots$$

Two-sided error is smaller

- Solve for f'(x):

$$2hf'(x_0)=f(x_0+h)-f(x_0-h)-\frac{2h^3}{3!}f'''(x_0)+\ldots$$

$$f'(x_0)=\frac{f(x_0+h)-f(x_0-h)}{2h}-\frac{h^2}{6}f'''(\xi)$$

The central difference formula above is 2nd order accurate, with the 3rd derivative expressing the truncation error.

2nd Derivative

$$f''(x_0)\approx\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}$$

If we were to approximate this derivative as a finite difference in a set of grid points, it would require three points , x_0, and one on either side.

2nd Partial Derivative

$$\frac{\partial^2 f(x_0,y_0)}{\partial x^2}\approx\frac{f(x_0+h,y_0)-2f(x_0,y_0)+f(x_0-h,y_0)}{h^2}$$

If we were to approximate this partial derivative as a finite difference in a set of grid points, it would require three points , x_0, and one on either side.

In a 2D problem, where we have *f(x,y),* we may need both 2nd order partial derivatives. We would then be using the central point and points one either side and above and below it for both derivatives. (5 pts total)

$$\frac{\partial^2 f(x_0,y_0)}{\partial y^2}\approx\frac{f(x_0,y_0+h)-2f(x_0,y_0)+f(x_0,y_0-h)}{h^2}$$

laPlace equation

$$\nabla^2F(x,y)=\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)F(x,y)=0$$

With boundary conditions at edges of region.

$$\left(\frac{f(x_0+h,y_0)-2f(x_0,y_0)+f(x_0-h,y_0)}{h^2}+\frac{f(x_0,y_0+h)-2f(x_0,y_0)+f(x_0,y_0-h)}{h^2}\right)F(x,y)=0$$

$$\left(\frac{f(x_0+h,y_0)-4f(x_0,y_0)+f(x_0-h,y_0)+f(x_0,y_0+h)+f(x_0,y_0-h)}{h^2}\right)F(x,y)=0$$

These finite difference equations for the laPlace equation look just like the equations we solved in the rat maze. (The boundary conditions would be the food pattern.)
This is motivation for efficient algorithms for very large sets of equations (fine grids of points) in that type of problem.

Optimization

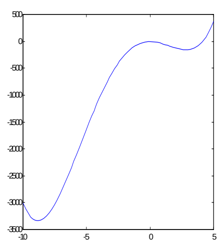
General Task

- Given a function of several variables, $f(x_1,x_2,\ldots,x_n)$, find its minimum value subject to a set of constraints:
  $g(x_1,x_2,\ldots,x_n) = 0$ and $h(x_1,x_2,\ldots,x_n) \leq 0$
  - *f* is scalar, *g* and *h* may or may not be.
  - *feasible point* is any point within domain of *f* that satisfies the constraints.
  - *unconstrained optimization* is an important subclass.

General Task in 1D

- Given a function of a single variable, f(x), find its minimum value(s).
- Constraint is commonly placed on acceptable range of the independent variable x.
- *feasible point* is any point within domain of *f* that satisfies the constraints.

- **Global minimum** occurs at x^* such that $f(x^*) \leq f(x)$ for <u>all</u> feasible points *x*.
- **Local minimum** occurs at x^* such that $f(x^*) \leq f(x)$ for <u>all</u> feasible points *x* <u>in the neighborhood of x^*</u>.



Extrema occur when derivatives vanish (stationary points)

- **1-Dimensional f(x)**
- At minimum or maximum of *f(x)*, the derivative of *f* vanishes:

$$\left.\frac{df}{dx} = 0\right|_{x=x^*}$$

- **Multidimensional**
- At extrema, all partial derivatives vanish (the gradient = 0)

$$\frac{\partial f}{\partial x_1} = \frac{\partial f}{\partial x_2} = \ldots = \frac{\partial f}{\partial x_n} = 0$$

$$\nabla f = 0 = \left(\frac{\partial f}{\partial x_1}\right)\mathbf{e}_1 + \left(\frac{\partial f}{\partial x_2}\right)\mathbf{e}_2 + \ldots + \left(\frac{\partial f}{\partial x_n}\right)\mathbf{e}_n$$

- As a first step, graph the function *f(x)* over the interesting domain.
- Root finding and minimization are related: we could search for a root of derivative instead of a minimum of parent function.
- *Unimodal:* Single minimum in domain, and function is strictly increasing on one side and decreasing on other side of minimum.

Find a bracket containing extremum

- From starting point x_0 explore points distributed as
  $x_k = x_{k-1} + 2^{k-1}h$ *(see text for examples)*
- When three successive points satisfy:
  $x < y < z$ and
  $f(y) < f(x), f(z)$, then the extremum is bracketed.
- Derivative of *f(x)* can be used to speed process.

Golden Section Search $\tau = \frac{2}{1+\sqrt{5}}$

- Pick two points within unimodal interval [a,b] such that $x1 = a + (1-\tau)(b-a)$ and $x2=a+ \tau(b-a)$, with $\tau= 0.618$.

a ______ x1 ______ x2 ______ b

- If $f(x_2) < f(x_1)$, then min *f* must lie in [x1,b]

______ a ______ x1 ______ x2 ______ b

Linear Convergence - failsafe

- ______ a ______ x1 ______ x2 ______ b

Now if $f(x1) < f(x_2)$

______ a ______ x1 ______ x2 ______ b ______

- Repeat until [a,b] are as small as desired.
- *The choice of τ causes coincidence of abscissas so that only one new evaluation is needed.*

Parabolic Interpolation

- Convergence is improved by parabolic (quadratic) interpolation.
- Evaluate the function at three points: $f(x_1)$, $f(x_2)$, $f(x_3)$.
- Fit these three points to the parabolic form: $g(x) = ax^2 + bx +c$. by solving 3 eqns for three unknowns a,b,c. *or determine the divided difference polynomial, which is equivalent.*
- Find the minimum of g(x) *analytically* , and use that value of x as one of the new triple of x-values.

- Convergence is not guaranteed within the unimodal interval, but when converging it will do so superlinearly.
- Matlab *fminbnd* uses a golden section to get started and then finishes up with parabolic interpolation.
- **Precision:** The function *f* is fairly insensitive to changes in *x* at the minimum. The value of *x* will be determined to less precision than *f*, typically about $(\varepsilon)^{1/2}$
- Matlab *Minimizer1D.m* for $f(x) = x^2 -2x +2$

Functions of several variables

- A Function of several variables can be minimized with the library call to *fminsearch*.
- Convergence can be slow here due to the large number of degrees of freedom in exploring a multidimensional surface rather than a line as is the case with f(x).
- For the case of two dimensions F(x,y), use of contour plots to generalize the surface can be useful.

Contour Plots

- For the function F(x,y), each contour on the surface represents a curve of constant value of F(x,y).
- Compare to topographic contour maps, where the curves are constant elevation as a function of latitude and longitude.
- Matlab has convenient contouring functions.

Conto…

Set up a grid in both x and y directions. *meshgrid* will make such a grid from the vectors.

```
%Contour demo
clear all
clc
close all
%set up a grid, evaluate the function on the
%  grid and then contour it
xlo=-3;
xhi = 3;
ylo =-4;
yhi = 4;
x = linspace(xlo,xhi,7);
y = linspace (ylo,yhi,9);
[X Y] = meshgrid(x,y);
%Display the grids
% X values increase left to right
%Y values increase top to bottom
%  The (1,1) element of the grid arrays
%  is the upper left element.
X
Y
```

Creates two (9 x 7) arrays

X contains the x value of each point;

Y contains its y-value.

The X and Y grids

```
X =

    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1
    -3    -2    -1
    -3    -2    -1
    -3    -2    -1
    -3    -2    -1
```

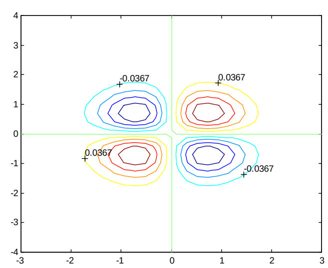
```
Y =

    -4    -4    -4    -4    -4    -4    -4
    -3    -3    -3    -3    -3    -3    -3
    -2    -2    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1    -1    -1
     0     0     0     0     0     0     0
     1     1     1     1     1     1     1
     2     2     2     2     2     2     2
     3     3     3     3     3     3     3
     4     4     4     4     4     4     4
```

Evaluate the Function on this grid

```
[X Y] = meshgrid(x,y);
X
Y
%Evaluate the function on this grid
F = (Y.*X).*(exp(-(X.^2 + Y.^2)));
figure
c =Contour(x,y,F,9);
clabel(c,'manual')
```

$$f(x,y) = yx \cdot e^{-(x^2+y^2)}$$

Contour plot with manual labeling



The y-axis has been given the usual orientation, even though the y-grid had the smallest value of y at the top (1st row)

Contouring Summary

- Establish *x* and *y* vectors with the grid definition.
- Use these vectors as arguments to *meshgrid* to establish the rectangular X and Y grids.
- Use X and Y with pointwise operators to evaluate the function *f(x,y)* over these grids.
- Plot contours with *contour*.

Minimiza…

```
XMIN =

   1.000000000125413e+000   1.000000000422316e+000


Fmin =

   2.956615764144878e-018
```

```
%using fmins from matlab
%first look at the functi
clear
close all
clc
x1 = linspace(-1.5,1.5,120);
x2 = linspace(-.5,1.5,120);
[X,Y]=meshgrid(x1,x2);
Fvec = vectorize(inline('100*(y-x*x)^2 + (1-x)^2','x','y'));
F = Fvec(X,Y);
v = [300,200,100,60,30,20,10,5,3,2,1,.1];
[C,h]=contour(X,Y,F,v);
clabel(C,h);
%clabel (C,'manual');
pause
tol=1.E-8;
xguess = input('enter x1,x2 as row vector guess')
options = optimset ('display', 'Iter', 'TolX', tol);
[XMIN Fmin] = fminsearch(inline('100*(x(2)-x(1)*x(1))^2 + (1-x(1))^2'),xguess,...
        options)
```

examples

- d-function
- rosenbrock