Numerical Integration Lesson 9.

Adriano FESTA

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica, L'Aquila

DISIM, L'Aquila, 20.05.2019

adriano.festa@univaq.it

Class outline

- Numerical Integration and Quadrature
- Newton-Cotes Formula
- Simpson Formula
- Parallelization
- Randomness in computers
- Montecarlo Methods

Next...

- We can interpolate $f(x)$
- Can we integrate $f(x)$?

Why?

- Often $f(x)$ is only known implicitly (known at a certain number of points)
- Often the anti-derivative of $f(x)$ is not known.

Integrals

We seek a solution to the following quantity

$$\int_a^b f(x)\,dx$$

The Fundamental Theorem of Calculus states that

$$\int_a^b f(x)\,dx = F(b) - F(a)$$

where F is the antiderivative of f. We don't know F, so we approximate the integral operation.

Integration

What is the integral $\int_a^b$?

- Let P be a partition of $[a, b]$ of $n + 1$ distinct and ordered points with $x_0 = a$ and $x_n = b$.
- For interval $[x_i, x_{i+1}]$ let m_i be a lower bound on $f(x)$
- For interval $[x_i, x_{i+1}]$ let M_i be an upper bound on $f(x)$
- Lower Sum:

$$L(f; P) = \sum_{i=0}^{n-1} m_i(x_{i+1} - x_i)$$

- Upper Sum:

$$U(f; P) = \sum_{i=0}^{n-1} M_i(x_{i+1} - x_i)$$

Integration

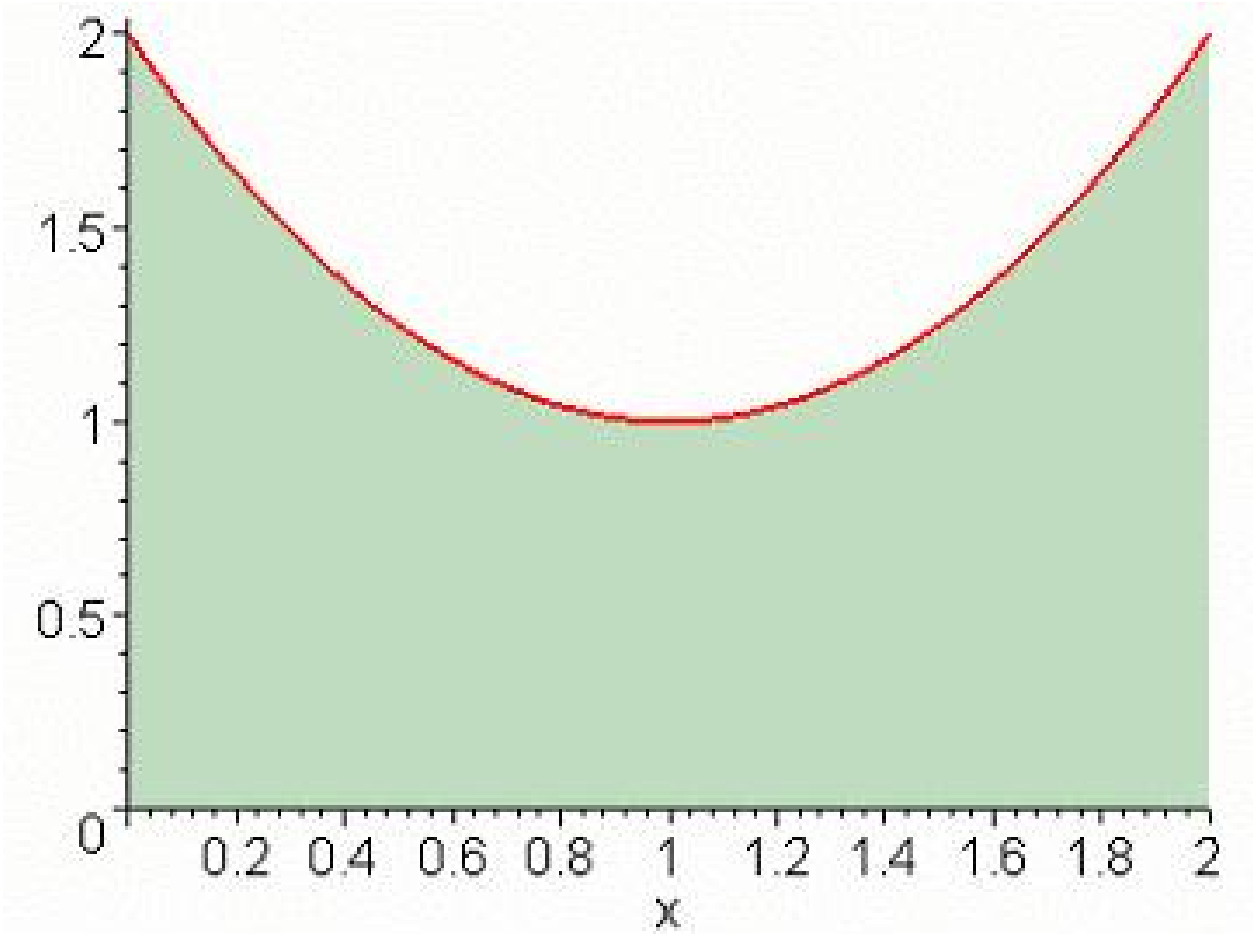
- The low sum always under-approximates the integral
- The upper sum always over-approximates the integral

$$L(f;P) \leq \int_a^b f(x)\,dx \leq U(f;P)$$

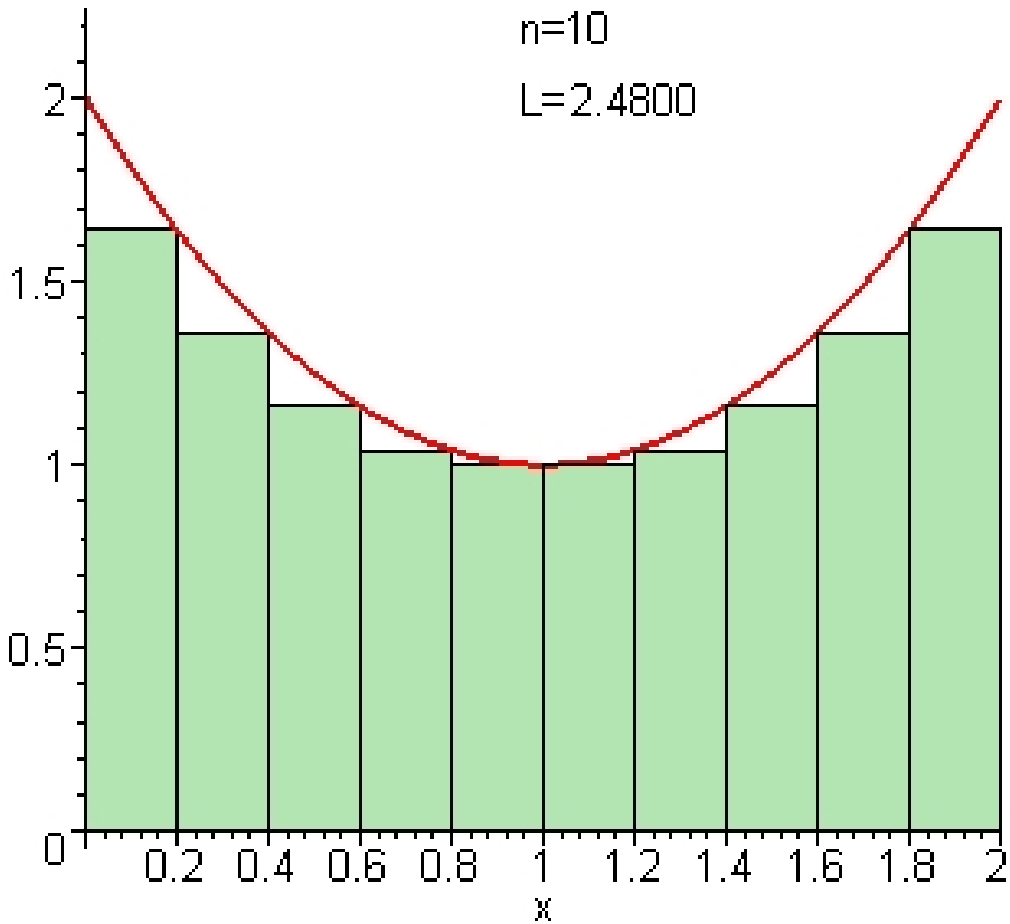
- In the limit, they are equal

$$\lim_{n\to\infty} L(f;P) = \int_a^b f(x)\,dx = \lim_{n\to\infty} U(f;P)$$

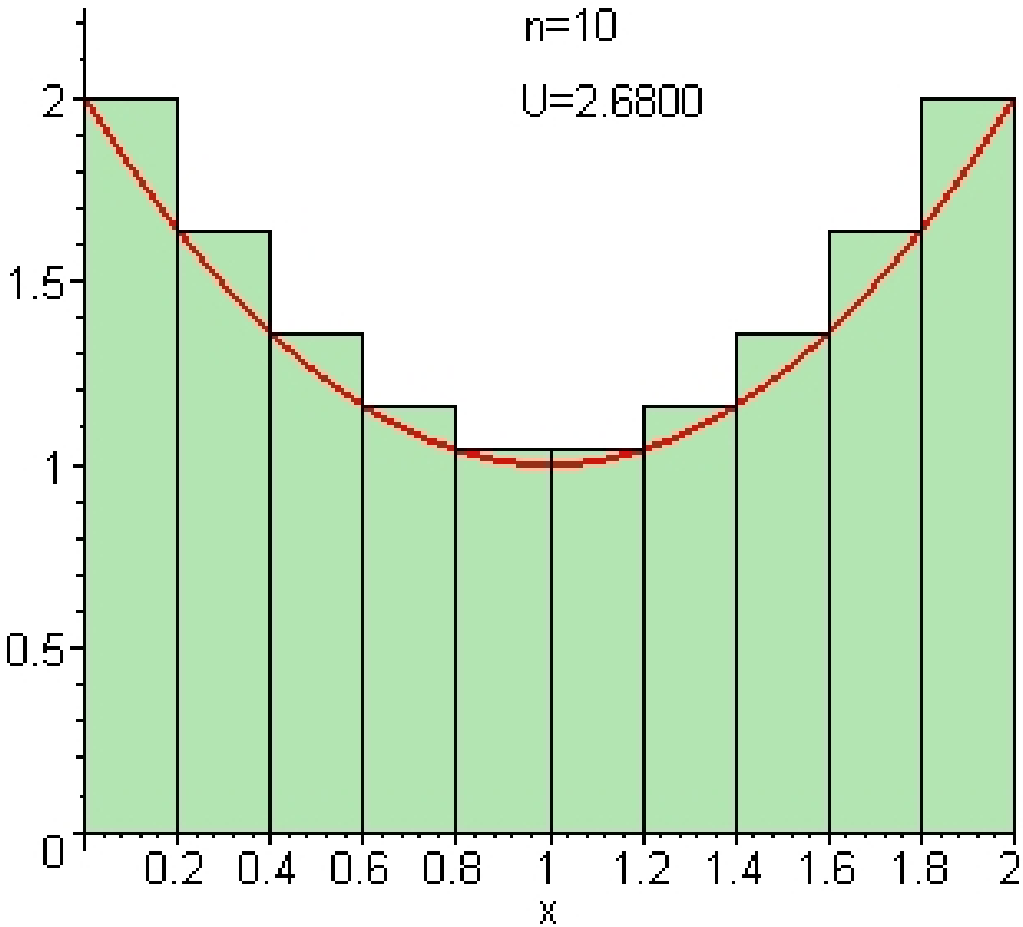
Graphically: Integral



Graphically: Lower sum



Graphically: Upper sum



Left-Riemann, Right-Riemann, Mid-Point

- The upper and lower bounds are often difficult to identify
- Use Left-Riemann, Right-Riemann, and Middle Riemann Sums
- Generally the Riemann sum is

$$S = \sum_{i=0}^{n-1} f(z_i)(x_{i+1} - x_i)$$

  for $x_i \leq z_i \leq x_{i+1}$
- $z_i = x_i$ is a Left Riemann Sum
- $z_i = x_{i+1}$ is a Right Riemann Sum
- $z_i = \frac{x_{i+1}+x_i}{2}$ is a Middle Riemann Sum

Goals

We have a way to compute integrals. Why aren't we done?
We may need to compute zillions of integrals or we may need to accurately compute integrals in very short times to meet a real-time requirement.

- Trapezoid Rule
- Composite Trapezoid Rule
- Simpson Rule
- Composite Simpson Rule
- Newton-Cotes in general
- Numerical MC

Trapezoid

Goal: Approximate

$$\int_a^b f(x)\,dx$$

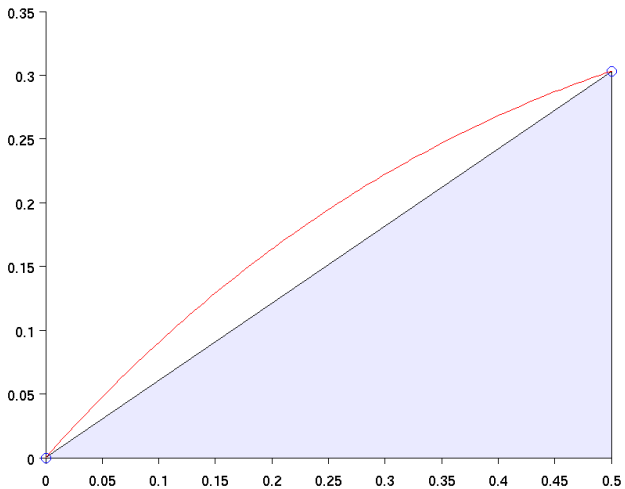
Goal: Approximate area under $f(x)$.

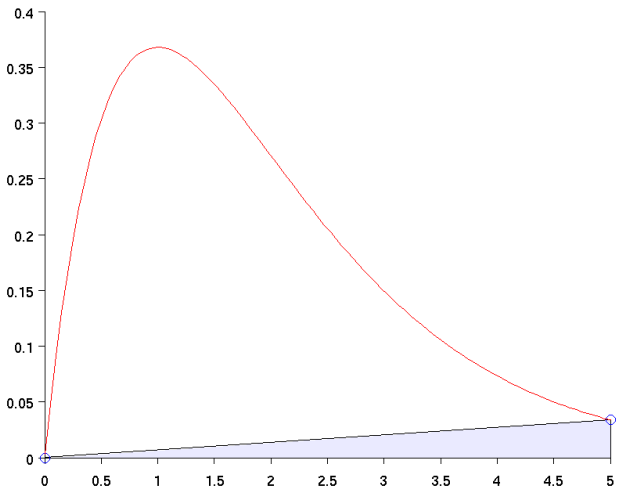
- Old Idea: Left,Right,Midpoint Riemann integration approximate $f(x)$ by a constant function and obtain the area under the constant function.
- New Idea: Trapezoid approximates $f(x)$ by a linear function (degree one polynomial) and obtains the area under the linear function.

Basic Trapezoid

Use endpoints $[a, b]$ to obtain a linear approximation to $f(x)$. The area under this function is the area of a trapezoid:

$$\int_a^b f(x)\,dx \approx \frac{1}{2}(b-a)(f(a)+f(b))$$





Basic Trapezoid

- Trapezoid Rule:

$$\int_{x_1}^{x_2} f(x)\,dx \approx \int_{x_1}^{x_2} P_1(x)\,dx = \frac{1}{2}(f_1 + f_2)h$$

$$\int_{x_1}^{x_2} f(x)\,dx \approx \frac{1}{2}(f_1 + f_2)h \text{ , where } f(x) = 15\,x^2$$

Example

$$\int_1^2 15\,x^2 \approx \frac{1}{2}(15 * 1^2 + 15 * 2^2) * 1$$

$$= \frac{1}{2}(15 + 60) = 37.5$$

- Analytical answer is $\int_1^2 15\,x^2 = 5\,x^3\Big|_1^2 = 40 - 5 = 35$.

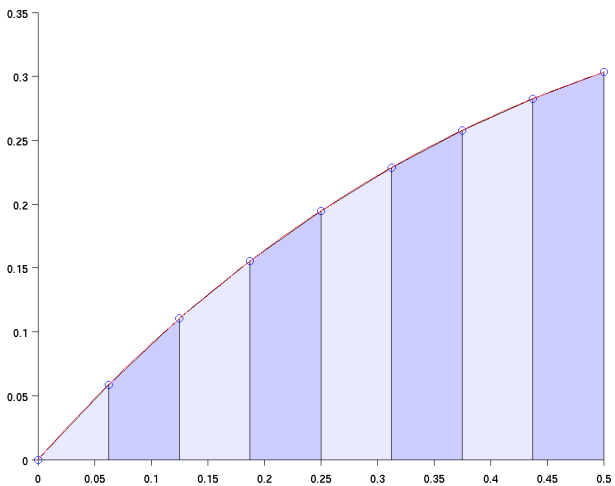
Composite Trapezoid

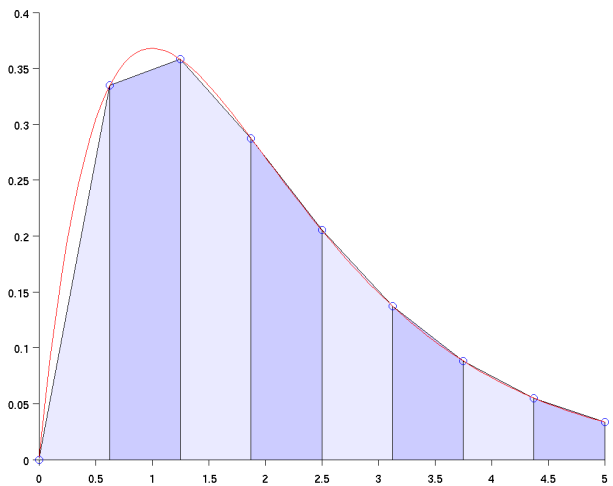
Obviously a naive linear approximation won't cut it.

Consider a partition $P = \{x_0 = a < \ldots x_n = b\}$ of $[a, b]$.

In each interval $[x_i, x_{i+1}]$ use the basic Trapezoid:

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n-1} \frac{1}{2}(x_{i+1} - x_i)(f(x_i) + f(x_{i+1}))$$





Composite Trapezoid

- With uniform spacing of P, $h_i = x_{i+1} - x_i = h$ is constant

$$T(f;P) = \int_a^b f(x)\,dx \approx \frac{h}{2}\sum_{i=0}^{n-1} f(x_i) + f(x_{i+1})$$

- This becomes

$$T(f;P) = \int_a^b f(x)\,dx \approx \frac{h}{2}\left(f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n)\right)$$

Pseudocode:

$h = (b-a)/n$
$sum = (f(a) + f(b))/2$
for $i = 1$ to $n-1$
  $sum = sum + f(x_i)$
end
$sum = sum \cdot h$

Example

Test composite trapezoid for

$$\int_0^5 xe^{-x}$$

Question: What is the order of accuracy?

Look at h^p with

$$p \approx \frac{log(err^{(k)}/err^{(k-1)})}{log(h^{(k)}/h^{(k-1)})}$$

Accuracy

So composite Trapezoid appears to be order 2. Why? Look first a basic Trapezoid:

$$\int_a^b f(x)\,dx \approx \frac{1}{2}(b-a)(f(a)+f(b))$$

Looking at the error

$$\begin{aligned}
E &= \int_a^b f(x)\,dx - \int_a^b \frac{(b-x)f(a)}{b-a} + \frac{(x-a)f(b)}{b-a}\,dx \\
&= \int_a^b \frac{(b-a)f(x)-(b-x)f(a)-(x-a)f(b)}{b-a}\,dx \\
&= \int_a^b (x-a)(x-b)f[a,b,x]\,dx \\
&= f[a,b,\xi]\int_a^b (x-a)(x-b)\,dx \text{(MVT)} \\
&= \frac{f''(\eta)}{2}\left(-\frac{1}{6}(b-a)^3\right) \\
&= -\frac{(b-a)^3 f''(\eta)}{12}
\end{aligned}$$

Accuracy

What about Composite Trapezoid?

$$T(f;P) = \int_a^b f(x)\,dx \approx \frac{h}{2}\sum_{i=0}^{n-1} f(x_i) + f(x_{i+1})$$

The error in each interval $[x_i, x_{i+1}]$ is

$$E_i = -\frac{h^3 f''(\eta_i)}{12}$$

So the total error is

$$\begin{aligned}\sum_{i=0}^{n-1} E_i &= \sum_{i=0}^{n-1} -\frac{h^3 f''(\eta_i)}{12} \\ &= -n\frac{h^3 f''(\eta)}{12}\,(IVT) \\ &= -\frac{(b-a)h^2 f''(\eta)}{12}\end{aligned}$$

Example

How many points should be used to ensure the composite Trapezoid rule is accurate to 10^{-6} for $\int_0^1 e^{-x^2}\,dx$? Need

$$\frac{(b-a)h^2 f''(\eta)}{12} \leq 10^{-6}$$

How big is $f''(x)$?

$$f(x) = e^{-x^2}$$
$$f'(x) = -2xe^{-x^2}$$
$$f''(x) = -2e^{-x^2} + 4x^2e^{-x^2}$$
$$f'''(x) = 12xe^{-x^2} - 8x^3e{-x^2}$$

So f''' is always positive. So f'' is monotone increasing and thus take on a maximum at an endpoint: $f''(0) = 2$. Then bound

$$\frac{(b-a)2h^2}{12} \leq 10^{-6}$$

Or

$$h^2 \leq 6 \times 10^{-6} \quad \Rightarrow \quad \sqrt{(1/6)}10^3 \leq n$$

or $n > 410$.

How do we improve Trapezoid?

- instead of a linear approximation, use a quadratic approximation
- $\Rightarrow$ Simpson's Rule

Simpson

- consider $\int_a^b f(x)\,dx$
- partition $P = \{a, a+h, b = a+2h\}$
- or $\int_0^{2h} f(x)\,dx$
- partition $P = \{0, h, 2h\}$
- replace $f(x)$ by a quadratic $p(x)$:

$$\begin{aligned} f(x) &\approx p(x) \\ &= f(0) + \frac{f(h) - f(0)}{2}x + \frac{f(2h) - 2f(h) - f(0)}{2h^2}x(x-h) \\ &= \text{newton form} \end{aligned}$$

- integrate $\int_0^{2h} p(x)\,dx$:

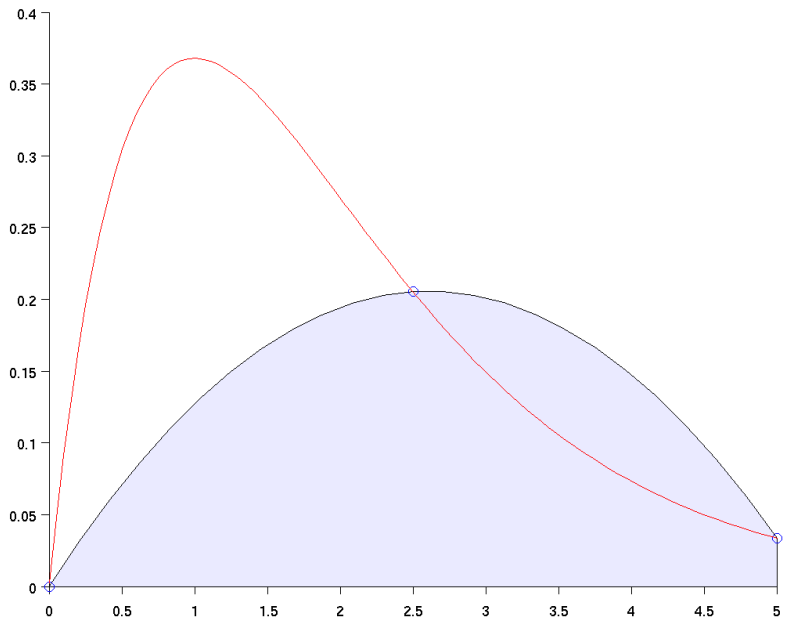
$$\begin{aligned} \int_0^{2h} f(x)\,dx &\approx \int_0^{2h} p(x)\,dx \\ &= \frac{h}{3}\left[f(0) + 4f(h) + f(2h)\right] \end{aligned}$$

Simpson

Since $b - a = 2h$ we have

basic Simpson's Rule

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



Composite Simpson

Over a uniform partition $P = x_0, x_1, \ldots, x_n$, use Basic Simpson's Rule over each subinterval $[x_{2i}, x_{2i+2}]$

$$\int_a^b f(x)\,dx = \sum_{i=0}^{n/2-1} \int_{x_{2i}}^{x_{2i+2}} f(x)\,dx$$

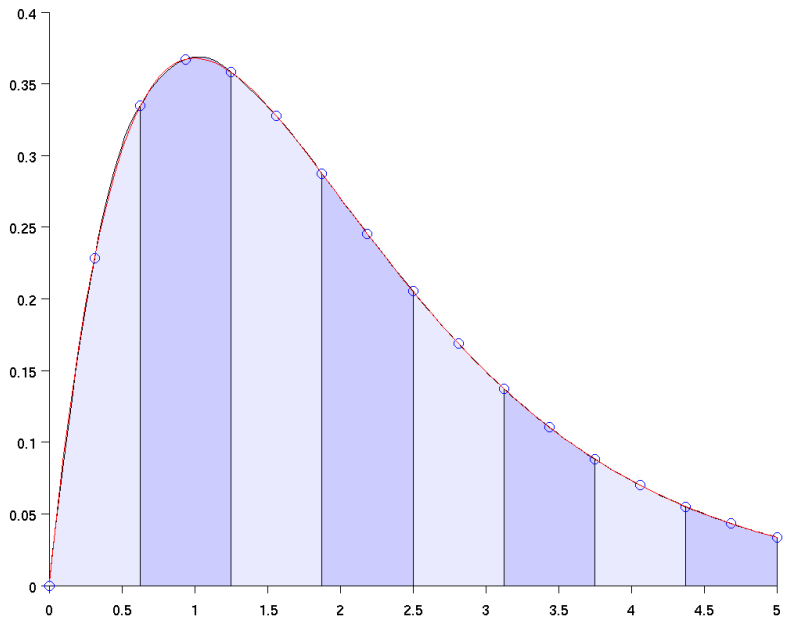
$$= \sum_{i=0}^{n/2-1} \frac{2h}{6}\left[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})\right]$$

$$= \frac{h}{3}\left[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \cdots + 4f(x_{n-1}) + f(x_n)\right]$$

Simpson

Composite Simpson's Rule

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[f(a) + f(b) + 4\sum_{i=1}^{n/2} f(a+(2i-1)h) + 2\sum_{i=1}^{n/2-1} f(a+2ih)\right]$$



Simpson

How accurate is Simpson? Recall composite Trapezoid:

$$I - T(f, P) = -\frac{1}{12}(b-a)h^2 f''(\xi) - \mathcal{O}(h^2)$$

Prediction? $\mathcal{O}(h^2)$? $\mathcal{O}(h^3)$? $\mathcal{O}(h^4)$?

Why is composite Simpson $\mathcal{O}(h^4)$?

Taylor Series:

$$f(a+h) = f + hf' + \frac{1}{2!}h^2 f'' + \frac{1}{3!}h^3 f''' + \frac{1}{4!}h^4 f^{(4)} + \frac{1}{5!}h^5 f^{(5)} + \dots$$

$$f(a+2h) = f + 2hf' + 2h^2 f'' + \frac{4}{3}h^3 f''' + \frac{2}{3}h^4 f^{(4)} + \frac{4}{15}h^5 f^{(5)} + \dots$$

This gives

$$\frac{h}{3}\left[f(a) + 4f(a+h) + f(b)\right] = 2hf + 2h^2 f' + \frac{4}{3}h^3 f'' + \frac{2}{3}h^4 f''' + \frac{5}{18}h^5 f^{(4)}$$

Integrating the Taylor Series expansion of $f(x)$ exactly gives

$$\int_a^b f(x)\,dx = 2hf + 2h^2 f' + \frac{4}{3}h^3 f'' + \frac{2}{3}h^4 f''' + \frac{4}{15}h^5 f^{(4)}$$

So basic Simpson's Rule gives an error of

$$-\frac{1}{90}\left(\frac{b-a}{2}\right)^5 f^{(4)}(\xi)$$

Why is composite Simpson $\mathcal{O}(h^4)$?

basic Simpson's Rule:

$$-\frac{1}{90}\left(\frac{b-a}{2}\right)^5 f^{(4)}(\xi)$$

Over $n/2$ subintervals $[x_{2i}, x_{2i+2}]$ becomes:

$$\begin{aligned} err &= \sum_{i=1}^{n/2} -\frac{1}{90}\left(\frac{x_{2i+2}-x_{2i}}{2}\right)^5 f^{(4)}(\xi_i) = -\frac{1}{90}\sum_{i=1}^{n/2}\left(\frac{2h}{2}\right)^5 f^{(4)}(\xi_i) \\ &= -\frac{1}{90}\frac{n}{2}h^5 f^{(4)}(\xi) = -\frac{1}{180}\frac{(b-a)}{h}h^5 f^{(4)}(\xi) \\ &= -\frac{b-a}{180}h^4 f^{(4)}(\xi) \end{aligned}$$

Composite Simpson's Rule

$$-\frac{b-a}{180}h^4 f^{(4)}(\xi)$$

We "gain" two orders over Trapezoid

Can we generalize?

Summary:

- left/right Riemann: approximate $f(x)$ by 0-degree $p(x)$ and integrate
- Trapezoid: approximate $f(x)$ by 1-degree $p(x)$ and integrate
- Simpson: approximate $f(x)$ by 2-degree $p(x)$ and integrate

Degree of Precision

- If the integration rule has zero error when integrating any polynomial of degree $\leq r$
- If the error is nonzero for some polynomial of degree $r+1$,

then, the rule has *degree of precision* equal to r.

(basic) Newton-Cotes rules:

name	n	formula
Trapezoid	1	$\frac{(b-a)}{2}\left[f(a)+f(b)\right]$
Simpson's $1/3$	2	$\frac{(b-a)}{6}\left[f(a)+4f(\frac{a+b}{2})+f(b)\right]$
Simpson's $3/8$	3	$\frac{(b-a)}{8}\left[f(a)+3f(a+h)+3f(b-h)+f(b)\right]$
Boole's	4	$\frac{(b-a)}{90}\left[7f(a)+32f(a+h)+12f(\frac{a+b}{2})+32f(b-h)+7f(b)\right]$

Error bounds for composite Newton-Cotes:

name	n	error	h
Trapezoid	1	$-\frac{(b-a)h^2}{12}f''(\xi)$	$h=b-a$
Simpson's $1/3$	2	$-\frac{(b-a)h^4}{90}f^{(4)}(\xi)$	$h=(b-a)/2$
Simpson's $3/8$	3	$-\frac{(b-a)h^4}{80}f^{(4)}(\xi)$	$h=(b-a)/3$
Boole's	4	$-\frac{2(b-a)h^6}{945}f^{(6)}(\xi)$	$h=(b-a)/4$

Randomness and MC

expected value and variance

- expected value: average value of the variable

$$E[x] = \sum_{j=1}^{n} x_j p_j$$

- variance: variation from the average

$$\sigma^2[x] = E[(x - E[x])^2] = E[x^2] - E[x]^2$$

throwing a die

- expected value: $E[x] = (1 + 2 + \cdots + 6)/6 = 3.5$
- variance: $\sigma^2[x] = 2.916$

Estimated $E[x]$

- to estimate the expected value, choose a set of random values based on the probability and average the results

$$E[x] = \frac{1}{N} \sum_{j=1}^{N} x_i$$

- bigger N gives better estimates

throwing a die

- 3 rolls: $3, 1, 6 \rightarrow E[x] \approx (3 + 1 + 6)/3 = 3.33$
- 9 rolls: $3, 1, 6, 2, 5, 3, 4, 6, 2 \rightarrow E[x] \approx (3+1+6+2+5+3+4+6+2)/9 = 3.51$

law of large numbers

- by taking N to ∞, the error between the estimate an the expected value is statistically zero. That is, the estimate will converge to the correct value

$$P(E[x] = lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} x_i) = 1$$

continuous extensions

- expected value

$$E[x] = \int_a^b x\rho(x)\,dx$$

$$E[g(x)] = \int_a^b g(x)\rho(x)\,dx$$

- variance

$$\sigma^2[x] = \int_a^b (x - E[x])^2\rho(x)\,dx$$

$$\sigma^2[g(x)] = \int_a^b (g(x) - E[g(x)])^2\rho(x)\,dx$$

- estimating the expected value

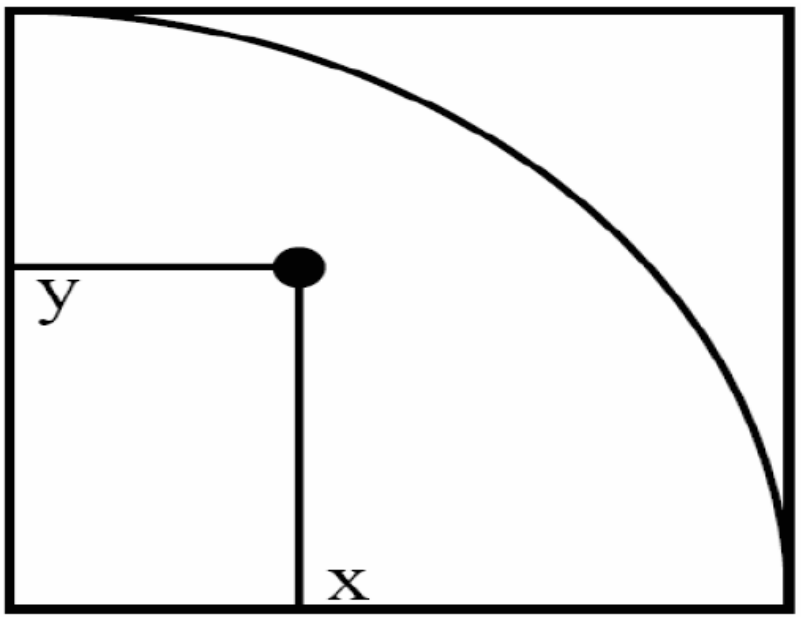
$$E[g(x)] \approx \frac{1}{N}\sum_{i=1}^{N} g(x_i)$$

2d: example computing π

Use the unit square $[0, 1]^2$ with a quarter-circle

$$f(x, y) = \begin{cases} 1 & (x, y) \in \ circle \\ 0 & else \end{cases}$$

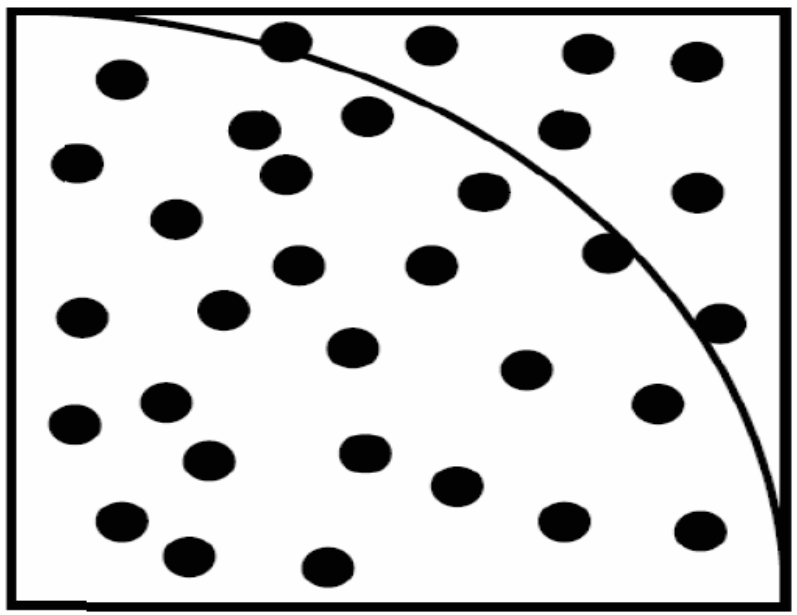
$$A_{quarter-circle} = \int_0^1 \int_0^1 f(x, y)\, dxdy = \frac{pi}{4}$$



2d: example computing π

Estimate the area of the circle by randomly evaluating $f(x, y)$

$$A_{quarter-circle} \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i, y_i)$$



2d: example computing π

By definition

$$A_{quarter-circle} = \pi/4$$

so

$$\pi \approx \frac{4}{N} \sum_{i=1}^{N} f(x_i, y_i)$$

2d: example computing π, algorithm

Pseudocode:

```
input N
call rand in 2d
for i=1:N
    sum = sum + f(x_i, y_i)
end
sum = 4 * sum/N
```

2d: example computing π, algorithm

The expected value of the error is $\mathcal{O}(\frac{1}{\sqrt{N}})$

- convergence does not depend on dimension
- deterministic integration is very hard in higher dimensions
- deterministic integration is very hard for complicated domains
- Trapezoid rule in dimension d: $\mathcal{O}(\frac{1}{N^{2/d}})$

CLT: Central Limit Theorem

The distribution of an average is close to being normal, even when the distribution from which the average is computed is not normal.

What?

- Let $x_1, \ldots, x_n$ be some independent random variables from any PDF (probability density function)
- Consider the sum $S_n = x_1 + \cdots + x_n$
- The expected value is $n\mu$ and the standard deviation is $\sigma\sqrt{n}$
- That is $\frac{S_n - n\mu}{\sigma\sqrt{n}}$ approaches the normal distribution
- What? The sample mean has an error of $\sigma/\sqrt{n}$

`http://youtube.com/watch?v=XAuMfxWg6eI`

Stochastic Simulation

- Stochastic simulation mimics physical behavior through random simulations
- ...also known as Monte Carlo method
- Usefulness:
  - nondeterministic processes
  - deterministic models that are too complicated to model
  - deterministic models with high dimensionality

Stochastic Simulation

- Two requirements for MC:
  - knowing which probability distributions are needed
  - generating sufficient random numbers
- The probability distribution depends on the problem (theoretical or empirical evidence)
- The probability distribution can be approximated well by simulating a large number of trials

Randomness

- Randomness $\approx$ unpredictability
- One view: a sequence is random if it has no shorter description
- Physical processes, such as flipping a coin or tossing dice, are deterministic with enough information about the governing equations and initial conditions.
- But even for deterministic systems, sensitivity to the initial conditions can render the behavior practically unpredictable.
- we need random simulation methods

Repeatability

- With unpredictability, true randomness is not repeatable
- ...but lack of repeatability makes testing/debugging difficult
- So we want repeatability, but also independence of the trials

```
>> rand('seed',1234)
>> rand(10,1)
```

Pseudorandom Numbers

Computer algorithms for random number generations are deterministic

- ...but may have long periodicity (a long time until an apparent pattern emerges)
- These sequences are labeled *pseudorandom*
- Pseudorandom sequences are predictable and reproducible (this is mostly good)

Random Number Generators

Properties of a good random number generator:

Random pattern: passes statistical tests of randomness

Long period: long time before repeating

Efficiency: executes rapidly and with low storage

Repeatability: same sequence is generated using same initial states

Portability: same sequences are generated on different architectures

Random Number Generators

- Early attempts relied on complexity to ensure randomness
- "midsquare" method: square each member of a sequence and take the middle portion of the results as the next member of the sequence
- ...simple methods with a statistical basis are preferable

Linear Congruential Generators

- Congruential random number generators are of the form:

$$x_k = (ax_{k-1} + c)(\mod M)$$

  where a and c are integers given as input.
- x_0 is called the *seed*
- Integer M is the largest integer representable (e.g. $2^{31} - 1$)
- Quality depends on a and c. The period will be at most M.

Example

Let $a = 13$, $c = 0$, $m = 31$, and $x_0 = 1$.

$$1,\ 13,\ 14,\ 27,\ 10,\ 6,\ \ldots$$

This is a permutation of integers from $1, \ldots, 30$, so the period is $m - 1$.

```
randgui('rand')
randgui('randssp')
```

History

- IBM used Scientific Subroutine Package (SSP) in the 1960's the mainframes.
- Their random generator, `rnd` used $a = 65539$, $c = 0$, and $m = 2^{31}$.
- arithmetic mod 2^{31} is done quickly with 32 bit words.
- multiplication can be done quickly with $a = 2^{16} + 3$ with a shift and short add.
- Notice (mod m):

$$x_{k+2} = 6x_{k+1} - 9x_k$$

...strong correlation among three successive integers

History

- Matlab used $a = 7^5$, $c = 0$, and $m = 2^{31} - 1$ for a while
- period is $m - 1$.
- this is no longer sufficient

```
randgui('randssp')
```

what's used?

Two popular methods:
1. Method of Marsaglia (period $\approx 2^{1430}$).

Initialize $x_0, \ldots, x_3$ and c to random values given a seed

Let $s = 2111111111 x_{n-4} + 1492 x_{n-3} 1776 x_{n-2} + 5115 x_{n-1} + c$

Compute $x_n = s \mod 2^{32}$

$c = floor(s/2^{32})$

2. *rand()* in Unix uses $a = 1103515245$, $c = 12345$, $m = 2^{31}$.

Even the Marsaglia method produces points in $n - D$ on only a small number of hyperplanes.

In general, the digits in random numbers are not themselves random...some patterns reoccur much more often.

Linear Congruential Generators

- sensitive to a and c
- be careful with supplied random functions on your system
- period is M
- standard division is necessary if generating floating points in $[0, 1)$.

Typical LCG values

Source	m	a	c
Numerical Recipes	2^{32}	1664525	1013904223
Borland C/C++	2^{32}	22695477	1
glibc (GCC)	2^{32}	1103515245	12345
ANSI C: Watcom C, Digital Mars, etc	2^{32}	1103515245	12345
Borland Delphi, Virtual Pascal	2^{32}	134775813	1
MS Visual C++	2^{32}	214013	2531011
Apple CarbonLib	$2^{31} - 1$	16807	0

Fibonacci

- produce floating-point random numbers directly using differences, sums, or products.
- Typical subtractive generator:

$$x_k = x_{k-17} - x_5$$

  with "lags" of 17 and 5.
- Lags much be chosen very carefully
- negative results need fixing
- more storage needed than congruential generators
- no division needed
- very very good statistical properties
- long periods since repetition does not imply a period

Sampling over intervals

If we need a uniform distribution over $[a, b)$, then we modify x_k on $[0, 1)$ by

$$(b - a)x_k + a$$

Non-uniform distributions

- sampling nonuniform distributions is much more difficult
- if the cumulative distribution function is invertible, then we can generate the non-uniform sample from the uniform:

$$f(t) = \lambda e^{-\lambda t}, \quad t > 0$$

thus

$$y_k = -\log(1 - x_k)/\lambda$$

where x_k is uniform in $[0, 1)$.

- ...not so easy in general

Quasi-Random Sequences

- For some applications, reasonable uniform coverage of the sample is more important than the "randomness"
- True random samples often exhibit clumping
- Perfectly uniform samples uses a uniform grid, but does not scale well at high dimensions
- quasi-random sequences attempt randomness while maintaining coverage

Quasi-Random Sequences

- quasi random sequences are not random, but give random appearance
- by design, the points avoid each other, resulting in no clumping