Introduction to Numerical Integration

- The problem of finding $I = \int_a^b f(x)\, g(x)\, dx$ is closely related to both differentiation rules and to formulae for expanding $f(x)$ in $[a, b]$. If the expansion functions are nice, we can plug in expansion for f and do term by term integral. It divides into
- **NI.1,2:** $f(x)$ assumed known at prescribed points (cf. Lagrange interpolation)
- **NI.3:** We can choose points where $f(x)$ to be calculated (cf. Chebyshev expansion for interpolation)
- **NI.5:** Monte Carlo Integration
- **NI.5:** is most important as it is what you get in big problems with multidimensional integrals!
- **NI.4:** has remarks on use of parallel computers.

  A good compendium of rules will be found in M. Abramowitz and J. Stegun, "Handbook of Mathematical Functions."

NI.1: Newton-Cotes Formulae

- (i) We write

$$I = \int_a^b f(x)\, g(x)\, dx \qquad \text{(NI.1)}$$

  where $f(x)$ is assumed known at $x_0 \ldots x_n$ and $g(x)$ is sufficiently simple so that integrals (NI.3), involving it, can be calculated. Of course, we split off $g(x) \neq 1$ to make $f(x)$ smoother and more reasonably approximated as a polynomial.

- We assume the form:

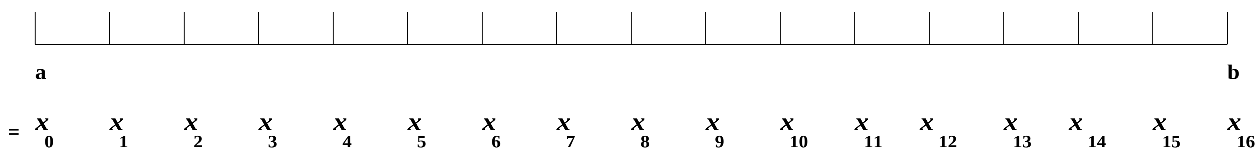
$$I = \sum_{r=0}^{n} w_r f(x_r) \qquad \text{(NI.2)}$$

  where the $n + 1$ w_r's are to be found by requiring I to be exact for all polynomials of degree $\leq n$.

- This is easy-for put

$$I_m = \int_a^b x^m g(x)\, dx \qquad \text{(NI.3)}$$

  which are calculable constants.



$$\int_a^b f(x) dx = \sum_i w_i f\left(x_i\right)$$

Linear Equations for Coefficients

- Our condition becomes

$$I_m = \sum_{r=0}^{n} w_r\, x_r^m \qquad : 0 \le m \le n \qquad \text{(NI.4)}$$

(linear nature of integral ensures that (NI.4) guarantees exactness of (NI.2) for all polynomials of degree $\le n$).

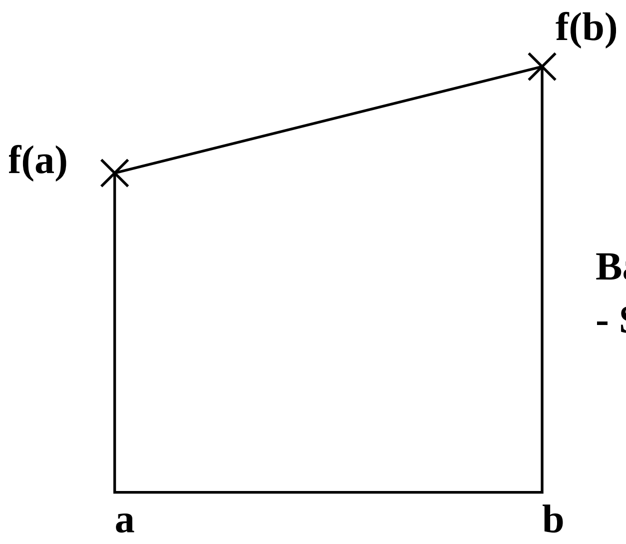
- We can write (NI.4) as:

$$A\underline{w} = \underline{b} \qquad \text{(NI.5)}$$

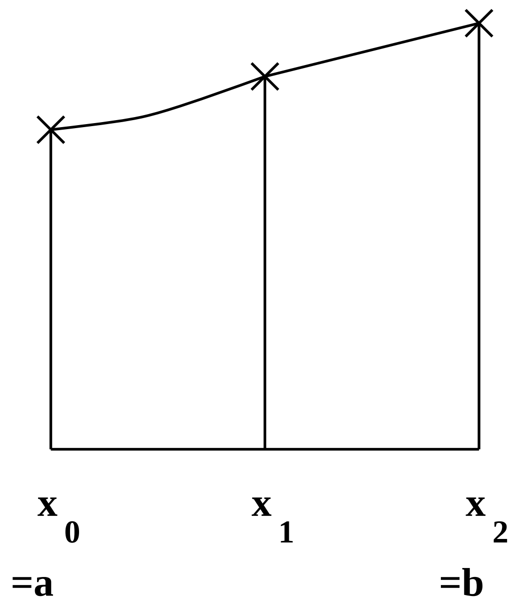
$\underline{w}$ is column vector of w_r's, $\underline{b}$ is column vector of I_m's and A is $(n+1) \times (n+1)$ matrix of coefficients $A_{mr} = (x_r)^m$. It is easy to show that A is nonsingular (if x_r distinct) and hence that (NI.5) can be solved and (NI.2) established for any $g(x)$ in (NI.1).

- Usually (NI.5) is solved—once and for all for given $g(x)$—and then (NI.2) can be used quickly for any $f(x)$ that crops up.

Examples of Newton Cotes Rules I

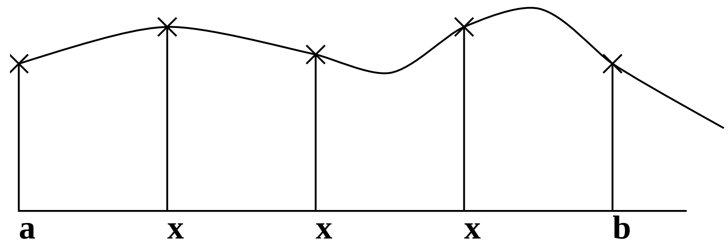


Basic Trapezoidal
- Straight line

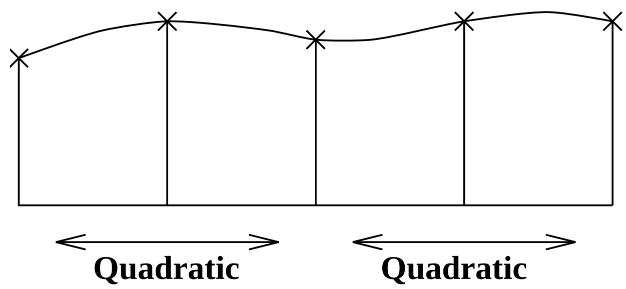


Basic Simpson
- Quadratic

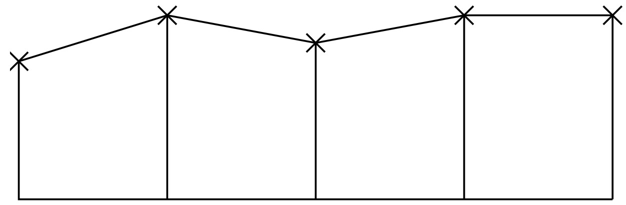
Examples of Newton Cotes Rules II



Newton Cotes
4th order
Polynomial



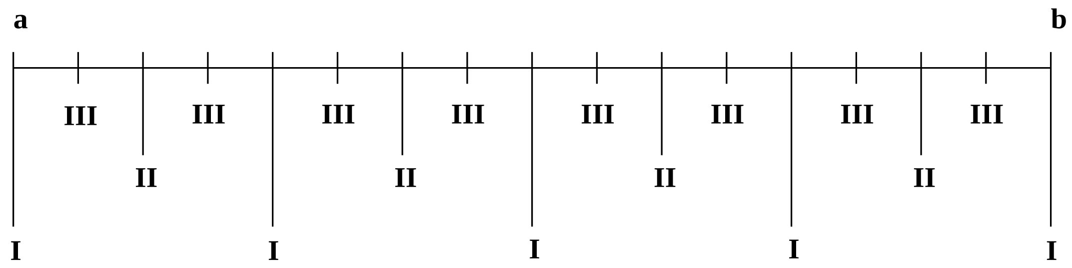
Simpson
Iterated
Quadratic



Trapezoidal
Iterated
Linear

Recursive Simpson (Romberg)

Step I followed by Step II
followed by Step III



Formulae for Examples:

Trapezoidal and Simpson's Rules

- **(a) Trapezoidal Rule**

$$n = 1,\ g(x) = 1 \qquad x_0 = a,\ x_1 = b$$

$$\int_a^b f(x)\,dx \approx \frac{h}{2}[+f(a) + f(b)] \quad h = b - a \qquad \text{(NI.6)}$$

- **(b) Simpson's Rule**

$$n = 2,\ g(x) = 1 \qquad x_0 = a,\ x_1 = \frac{1}{2}(a+b),\ x_2 = b$$

$$\int_a^b f(x)\,dx = \frac{h}{3}[f(x_0) + 4f(x_1) + f(x_2)] \qquad \text{(NI.7)}$$

where $h = \frac{1}{2}(b-a)$

Summary of Newton Cotes Rules

- Both these are equi-spaced (in x_i) rules with $g(x) = 1$ and the generalization to arbitrary n are called Newton-Cotes formulae (see Abramowitz and Stegun to find them explicitly written out).
- We can relax restrictions:
  - (a) x_i equidistant
  - (b) $x_0 = a$ and/or $x_n = b$
  - (c) $g(x) = 1$
- The reader need only solve (NI.5) to find relevant formulae for any n.
- You will find in the books, discussions of so called open formulae where $g(x) = 1$ as above, but one omits one or both of the end-points. The remaining n or $n-1$ points ($x_1 \ldots x_n$ or $x_0 \ldots x_{n-1}$ or $x_1 \ldots x_{n-1}$) are still equidistant.

Errors in Newton Cotes Formulae

- (ii) The errors in (NI.6,7) are written in the form:

$$\begin{aligned} \text{Trapezoidal—(NI.6)} \quad & -h^3 f''(\xi)/12 \\ \text{Simpson—(NI.7)} \quad & -h^5 f^{iv}(\xi)/90 \end{aligned} \qquad \text{(NI.8)}$$

  where ξ is some value of x in $[a,b]$.

- Note that, as expected, trapezoidal formula is exact for linear functions (error $\propto f''(\xi)$ vanishes for $f(x) = \alpha x + \beta$).
- But Simpson's rule—which as derived is exact for quadratic $f(x)$—is unexpectedly exact for cubics.
- In other words, its error is particularly small and it is a widely used formula in its extended form:

$$\begin{aligned} \int_{x_0}^{x_{2n}} f(x)\,dx \; = \; & \frac{h}{3}[f_0 + 4f_1 + 2f_2 + 4f_3 + \ldots \\ & + \; \ldots + f_{2n}] - \frac{nh^5}{90} f^{(iv)}(\xi) \quad \text{(NI.9)} \end{aligned}$$

  where $f_i = f(x_i)$ and x_i are equidistant.

- One gets this formula by applying Simpson to intervals x_0 to x_2, x_2 to $x_4 \ldots x_{2n-2}$ to x_{2n} in succession.

Use of High Order Newton Cotes

- One could also generate a formula in terms of the same function values $f(x_i)$ by using the Newton-Cotes formula appropriate to $2n+1$ points.
- This gains an extra derivative in error just like Simpson, and error is $ch^{2n+3} f^{2n+2}(\xi)$ where c is some known number.
- In practice, this is not often used because
  - (a) the implied interpolation of $f(x)$ over a wide range by a high order polynomial can produce disastrous results when $f(x)$ not polynomial (e.g., has similarities in its value or derivatives).
  - On the other hand, local interpolation by $f(x)$ by a low order polynomial (used in (NI.9)) is much safer;
  - (b) it is much easier to estimate error in iterated Simpson of equation (NI.9) as we will describe in Section NI.2.

Strategies for Manipulating Integrals before using Standard Numerical Integration

- (iii) Points of Technique
- Note, when evaluating $I = \int_a^b f(x)\, g(x)\, dx$, one can choose between two strategies
  - (a) Change of variable. For example, $dy = g(x)\, dx$ converts integral to $g(y) = 1$ form.
  - (b) Newton-Cotes (or Gauss—see NI.3) formula with weight function $g(x)$.
- A third strategy is to change the integration range (e.g., to take b from ∞ to finite value).
- How do we choose between strategies and how to implement them?

Criteria for Choosing Manipulation

- In each case, one must simply use criterion.
  - Is $f(x)$ or $f(y)$ better approximated by a polynomial?
  - One can use same criterion both to choose $g(x)$ and/or subtract off some particularly nasty part (e.g., a singularity) of $f(x)$.
- For example

$$\begin{aligned} I &= \int_a^b f(x)\, g(x)\, dx \\ &= I_1 + I_2 \\ \text{where } f(x) &= f_1(x) + f_2(x) \\ \text{and } I_i &= \int_a^b f_i(x)\, g(x)\, dx \end{aligned}$$

- where $f_1(x)$ may be well approximated by a polynomial and $I_1 = \int_a^b f_1(x)\, g(x)\, dx$ could be done numerically.
- $f_2(x)$ may be a nasty function to integrate numerically (e.g., x^α with α non-integer) but chosen so that we can do $I_2 = \int_a^b f_2(x)\, g(x)\, dx$ analytically.

Dangers in Integrating

Non-Analytic Functions

- (iv) Further Examples of Practical Technique
- a) $I = \int_0^x x^\alpha$ exists for $\alpha > -1$. Namely, $I = \frac{x^{\alpha+1}}{\alpha+1}$, but $f(x) = x^\alpha$ is not a polynomial unless α is an integer.
- In typical integration formula

$$I = \frac{h}{3}\left[f(0) + 4f(h) + 2f(2h) + \ldots\right] + 0(h^4)$$

  The error is $0\left(h^4 f^{(iv)}\right)$.

- Clearly this numerical procedure doesn't work if $\alpha < 0$ even though integral well defined as $-1 < \alpha < 0 \Rightarrow f(0)$ infinite, but I well defined.
- We could use an "open" formula which does not involved $f(0)$ but even then, errors will be large. For example $\int_0^h f(x)\,dx$ has error something like $h^5\, f^{(iv)}(\zeta)$ where ζ in range $0 \le x \le h$. If $f(x) = x^\alpha$, $f^{(iv)}(\zeta) \sim \zeta^{\alpha-4}, \zeta \sim h$ implies that error is $h^{1+\alpha}$ for each interval.

Why Do We Have Problems with Non-Analytic Functions?

- What is the problem?
- $f(x) = x^\alpha$ is just not well approximated by a polynomial. Note this is true even when α is > 0, but not an integer.
- For example

$$\begin{aligned} f(x) &= x^2\,, \qquad f^{(1)}(x) = x\,, \qquad f^{(2)}(x) = 1 \\ f^{(3)\ldots} &= 0 \qquad \Rightarrow \text{errors small (actually zero)} \end{aligned}$$

- But for $\alpha = 1/2$

$$\begin{aligned} f(x) &= \sqrt{x} \\ f^{(1)} &= \frac{1}{2\sqrt{x}} \\ f^{(2)} &= \frac{-1}{4x^{3/2}} \qquad \Rightarrow \text{errors large} \end{aligned}$$

- $1/\sqrt{x}$ is certainly a dreadful polynomial, but $\sqrt{x}$ is not good either as derivative is $1/\sqrt{x}$ and so not well behaved.

Combining Analytic and Numerical Integration

- (b) Well, we don't need numerical integration to perform $\int f(x)\,dx$ with $f(x) = x^\alpha$ as we can do analytically.
- However, what about $\alpha = \pm 1/2$ examples.

$$\begin{aligned} f(x) &= \frac{1}{\sqrt{x}} \exp\left(-x^3\right) \\ \text{or} &= \frac{1}{\sqrt{x}} + \exp\left(-x^3\right). \end{aligned}$$

- There are three possible strategies which in a practical case might need all to be combined.
- (c) **Strategy I** for integrating non-polynomial approximatable functions.

$$f(x) = f_1(x) + f_2(x)$$

  - where for example

$$f_1(x) = \exp\left(-x^3\right) \qquad f_2 = \frac{1}{\sqrt{x}}$$

  - Do $f_1(x)$ numerically
  - Do $f_2(x)$ analytically

Cunning Choice of Weight Functions

- **Strategy II** is used when difficult function is multiplicative, not additive.
- Suppose

$$
\begin{aligned}
I &= \int f(x)dx \quad \text{where} \\
f(x) &= f_1(x)\, f_2(x) \quad \text{so that} \\
I &= \int w(x)\, f_2(x)
\end{aligned}
$$

with $f_1(x)$ viewed as a weight function $w(x)$.

- Use $f_1(x)$ as a prescribed weight function to derive special integration formulae for doing $\int w(x)$ polynomials. We described these earlier.
- This strategy applies to the case $f(x) = \frac{1}{\sqrt{x}} \exp\left(-x^3\right)$ with $w(x) = 1/\sqrt{x}$.

Change of Variable Before Numerical Integration—I

- **Strategy III**
  - We can also use change of variables in $\int f(x)$ with $f(x) = \frac{1}{\sqrt{x}} \exp\left(-x^3\right)$.
  - Namely, if

$$\begin{aligned} \frac{1}{\sqrt{x}}\, dx &= dy \\ y &= 2\sqrt{x} \\ \frac{1}{\sqrt{x}} \exp\left(-x^3\right)\, dx &= \int \exp\left(-\left(\frac{y}{2}\right)^6\right)\, dy \end{aligned}$$

  - which can be done reliably numerically by approximating integrand by a polynomial, as in standard formulae.

Change of Variable Before Numerical Integration—II

- Let us consider implications of the transformation $x \to y = 2\sqrt{x}$.
- Integration formulae for x (which assume integrand $f(x) \sim$ polynomial) $\Rightarrow$ equally spaced (or distribution of Gaussian formulae) grid points.
- But we have seen y is a better variable and we will rather get equally spaced points in y.
- Now $dx = \sqrt{x}\, dy$. A fixed spacing $dy \Rightarrow$ spacing in x space $\propto \sqrt{x}$.
- That is, x points are concentrated near the origin.
- This is intuitively reasonable as the integrand $\frac{1}{\sqrt{x}} \exp\left(-x^3\right)$ is rapidly varying near the origin.

How Do You Choose Correct Strategy?

- d) General Advice
  - The user must "stare" at his or her function and decide on "correct" choice of strategy.
  - There will be many correct choices.
  - One can mix strategies and also may need different strategies in different regions.
  - Consider

$$I = \int_0^\infty \frac{1}{\sqrt{x}} \left[\exp\left(-x^3\right) + \exp(-x)\right] dx = I_1 + I_2$$

- with I_1 given by

$$I_1 = \int_0^{x\sim 1} \frac{1}{\sqrt{x}} \left[\exp\left(-x^3\right) + \exp(-x)\right] dx.$$

- Use change of variables $y = 2\sqrt{x}$ <u>or</u> weight $w(x) = 1/\sqrt{x}$ formula.

Hybrid Strategy Example (continued)

- The rest of the integrand

$$I_2 = \int_1^\infty f(x)\,dx$$

$$f(x) = \frac{1}{\sqrt{x}}\left[\exp\left(-x^3\right) + \exp(-x)\right] \text{ again.}$$

- Here, the problem is "infinite range" of integral.
- So we let either

$$\begin{aligned} y &= \exp(-x/2) \qquad x = (-2\log y) \\ dy &= -\frac{1}{2}\exp(-x/2)\,dx \\ I_2 &= \int_0^{1/\sqrt{e}} \frac{2}{\sqrt{(-2\log y)}}\left(y + \frac{1}{y}\,\exp\left((2\log y)^3\right)\right)\,dy \end{aligned}$$

- or

$$\begin{aligned} y &= \tfrac{1}{x} \\ I_2 &= \int_0^1 \left(\frac{1}{y}\right)^{3/2}\left[\exp\left(-\frac{1}{y^3}\right) + \exp\left(-\frac{1}{y}\right)\right]\,dy \end{aligned}$$

Note these integrals are well behaved.

NI.2: Romberg or Adaptive Integration

- A very convenient way of evaluating integrals is to repeatedly apply Simpson's rule—halving interval each time.

a								b	
x				x				x	Step 1
•		x		•		x		•	2
•	x	•	x	•	x	•	x	•	3
				→ x					

- The point is that, as diagram indicates, one need only evaluate half as many (x in diagram) functions as you expect because you use old values (• in figure).
- As the error is proportional to nh^5 and n doubles and h halves each go, error should go down by $\approx 1/16$ each iteration and convergence is fast. So, in practice, one halves h until change in integral between steps is less than some assigned value.

NI.3: Gaussian Integration

- i) We are evaluating the same formula

$$I = \int_a^b g(x)\, f(x)\, dx \qquad \text{(NI.1)}$$

in terms of the same ansatz

$$I = \sum_{r=0}^{n} w_r\, f(x_r) \qquad \text{(NI.2)}$$

- Now we choose both w_r and x_r (a grand total of $2n+2$ parameters) to make (NI.2) exact for polynomials of degree $\leq 2n+1$. ($2n+2$ equations analogous to (NI.3)).
- With Newton-Cotes, one only chooses $n+1$ parameters w_r viewing x_r as fixed.
- Thus, Newton-Cotes formulae are exact for polynomials of degree $\leq n$.

Some Mathematics Underlying Gaussian Integration

- This can be ignored!
- Let $P_n(x)$ be the polynomials orthogonal with respect to weight function $g(x)$

$$\int_a^b g(x)\, P_n(x)\, P_m(x)\, dx = h_n\, \delta_{nm} \qquad \text{(NI.10)}$$

  h_n is a known irrelevant constant.
- Let $x_i^{(n)}$ be the i th zero of $P_n(x)$.
- Define $K_n(x, y)$ by the formula

$$K_n(x, y) = \sum_{k=0}^{n} P_k(x)\, \frac{P_k(y)}{h_k} \qquad \text{(NI.11)}$$

- If $\Pi_n(x)$ is any polynomial of degree n, then one can easily show:

$$\int_a^b g(x)\, K_n(x, y)\, \Pi_n(x)\, dx = \Pi_n(y) \qquad \text{(NI.12)}$$

- It is not difficult to see that (NI.2) is satisfied by

$$x_i = x_i^{(n+1)}, \quad w_i = \frac{1}{K_{n+1}\left(x_i^{(n+1)};\, x_i^{(n+1)}\right)} \qquad \text{(NI.13)}$$

More Mathematics Deriving Gaussian Integration Formulae

- To convince yourself, choose a particular complete set of polynomials of degree $2n+1$.
    - a) Firstly, $P_{n+1}(x)$ multiplied into 1, x, x^2, $\ldots x^n$
    - For these, I is zero because $x_i = x_i^{(n+1)}$ are at zeros of $P_{n+1}(x)$.
    - b) Secondly,

      $$K_{n+1}\left(x_i^{(n+1)}, x\right)\left(x - x_0^{(n+1)}\right)\left(x - x_1^{(n+1)}\right)$$
      $$\ldots\left(x - x_{i-1}^{(n+1)}\right)\left(x - x_{i+1}^{(n+1)}\right)\ldots\left(x - x_n^{(n+1)}\right)$$

      Note: Omit $\left(x - x_i^{(n+1)}\right)$ factor
    - By construction, the only term that contributes to I for this polynomial is that with $r = i$ and this gives formula for w_i.
- This is quick derivation of Gaussian Integration. Key point is that x_i are zeroes of the orthogonal polynomial $P_{n+1}(x)$.

Basic Gaussian Integration

- For $g(x) = 1$, we get normal Gauss formulae with:

$$x_i = i^{th} \quad \text{zero of} \quad P_{n+1}(x)$$

$$w_i = \frac{2}{[1 - x_i^2]} \left[P'_{n+1}(x_i)\right]^{-2} \qquad \text{(NI.14)}$$

  $P_{n+1}(x)$ is Legendre Polynomial of degree $n + 1$.

- Clearly, Gaussian integration is best when $f(x)$ is very hard to evaluate, and one must minimize value of n.
- However, it is very hard to estimate error in calculation as the simple iterative scheme described in Section NI.2 for Simpson's rule cannot work.
- Not only is hard to evaluate weights and x_i's as n iterates $\rightarrow \infty$, but you have to recalculate all n functions each time. Note: There is no overlap in the x_i values as we double grid points ($n \rightarrow 2n + 1$).

When Should You Use Gaussian?

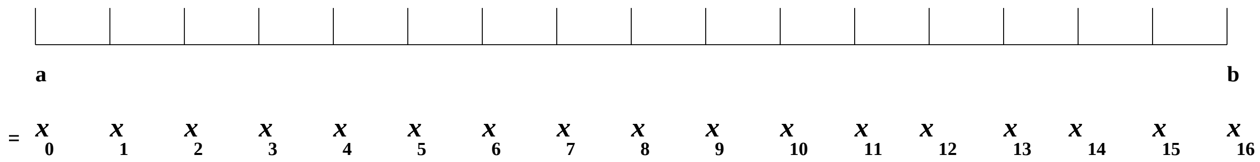
- (iv) Comparing Gauss with fixed grid point method, Gauss "saves" a factor of two by doubling the polynomial order for fixed number of function evaluations. However, this advantage is lost when compared to Romberg as latter also gains a factor of two as reuses points. Romberg is further much easier to code.
- (v) So use Romberg if you have a few integrals which are needed to prescribed accuracy. Use Gaussian Integration when you have a lot of integrals with similar $f(x)$ and the right value of n can be determined before hand.

Three Choices of x_i in Numerical Integration

Express $\quad I = \sum_{r=0}^{n} w_r f\left(x_r\right)$

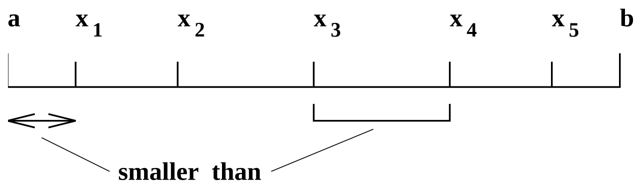
as $\quad \int_a^b f(x)dx = \sum_i w_i f\left(x_i\right)$

Newton-Cotes



Gaussian Integration

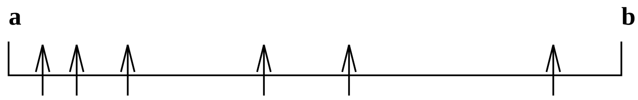
Unequally spaced prescribed points



exact for polynomials of degree $\leq$ 13

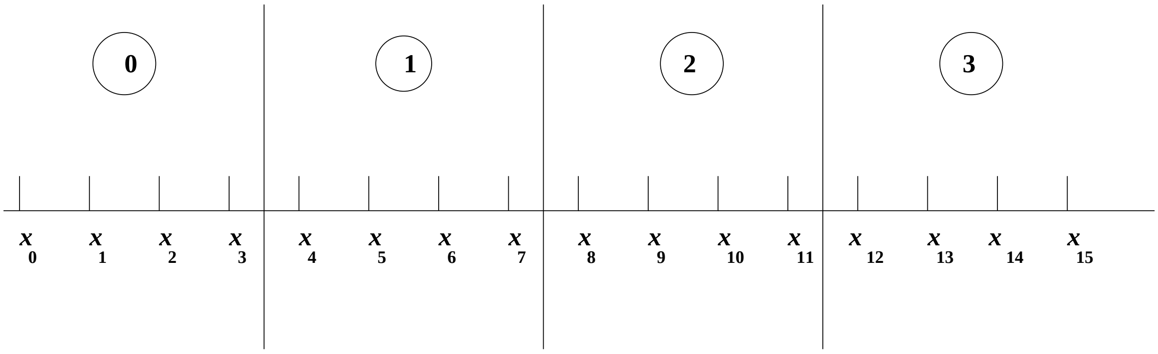
i.e., points clustered near end points

Monte Carlo Integration



Random Points

NI.4: Parallel Computing and Integration



- If have $n+1$ point integration ($n = 15$ above) divide into N_{proc} ($N_{\mathrm{proc}} = 4$ above) equal parts.
- $M = (n+1)/N_{\mathrm{proc}}$ ($= 4$ above) integration points in each processor. Each processor does an internal sum.

$$
\begin{aligned}
I_l &= \sum_{i=B_l}^{F_l} w_i\, f(x_i) \qquad l = 0 \ldots N_{\mathrm{proc}-1} \\
\text{index } B_l &= Ml \\
F_l &= B_l + M - 1
\end{aligned}
$$

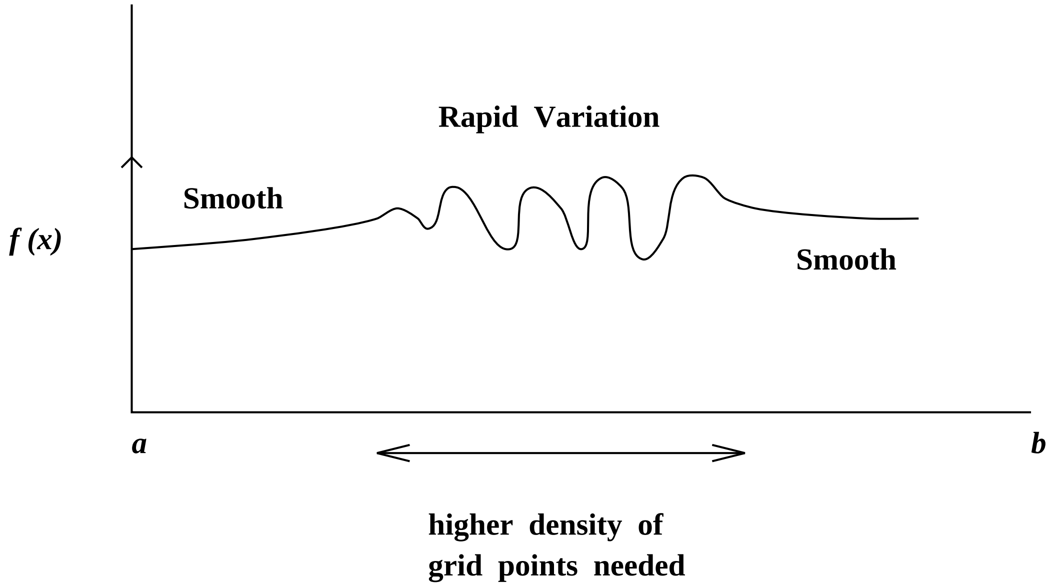
- Then total integral is

$$I = I_0 + I_1 \ldots + I_{N_{\mathrm{proc}}-1}$$

- There is ONLY communication in one place—a global COMBINE (reduction).

Issues for Parallel Computing

- i) If N_{proc} does not exactly divide $n+1$, then one (or more) processors have less work than the others—minor load imbalance and slightly tedious programming.
- ii) Adaptive algorithms (e.g., Romberg) could lead to load imbalance if one introduces new integration points selectively (so that one gets more integration points in one processor than another).



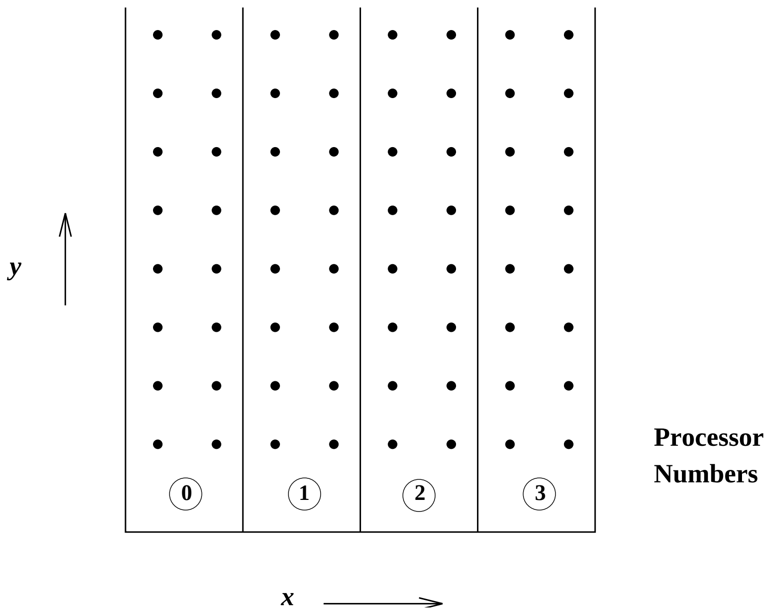
- Parallel random numbers were discussed in statistics module.

Two-Dimensional Integrals

- iii) For most one-dimensional integrals would not need parallel computing as rarely is this a big calculation.
- However, in multidimensional integrals, one can need more computing power.

$$\int_a^b dx \quad \int_c^d dy \, f(x, y)$$

- In first term, divide the x points between processors, as illustrated, for one-dimensional case. In second term, do internally to each processor.



- iv) We will return to multidimensional case when we discuss Monte Carlo methods, which are a more general and useful approach.

NI.5: Monte Carlo Integration

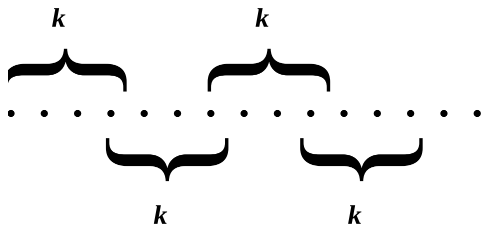
- (i) "All" integration methods can be written as

$$I = \int_a^b g(x)\, f(x)\, dx = \sum_{i=0}^{n} w_i f\,(x_i)$$

- where the $n+1$ points x_i and $n+1$ weights w_i are determined in sundry ways to optimize various properties, as described earlier.
- If we increase n, the dependence of error is

Error $\propto$	$\frac{1}{n^2}$	iterated Trapezoidal
	$\frac{1}{n^4}$	iterated Simpson
	$\frac{1}{n^{2m}}$	iterated m-order Gauss

- with iterated groups of k points.



k = 2 **Trapezoidal**

k = 3 **Simpson**

k = m **Gauss**

Basic Formulation of Integral as a Mean (Expectation Value)

- Taking $g(x) = 1$ for convenience, we write our basic integral as

$$
\begin{aligned}
I &= (b-a)\int_{-\infty}^{+\infty} f(x)\,p(x)\,dx \\
p(x) &= \frac{1}{(b-a)} \qquad a \le x \le b \\
&= 0 \quad \text{outside} \quad [a,b]
\end{aligned}
$$

- where $p(x)$ has carefully been constructed to be a properly normalized probability distribution for a random variable x which is uniformly distributed in the range $a \le x \le b$.

Basic Monte Carlo Approach to Integration

- Now consider the Monte Carlo integration method, which writes $I = \langle \tilde{f} \rangle$.

$$\langle \tilde{f} \rangle = \int_a^b \frac{\tilde{f}(x)\,dx}{b-a} \qquad \tilde{f} = (b-a)\,f \qquad \text{(NI.15)}$$

  I is mean of $\tilde{f}(x)$ where x is uniformly distributed random variable between a and b.

- Defining

$$\sigma^2 = \int_a^b \frac{\left[\tilde{f}(x) - I\right]^2 \, dx}{b-a} \qquad \text{(NI.16)}$$

- We know from central limit theorem that

$$\frac{1}{n} \sum_{i=1}^{n} \tilde{f}(x_i) \to I \qquad \text{(NI.17)}$$

  with error $\frac{\sigma}{\sqrt{n}}$.

- (NI.15) is of the same form as our previous formulae: all the w_i are equal, while the positions x_i are random. However, it seems a bad idea as error $\to 0$ much slower than even iterated trapezoidal.

Why Monte Carlo Methods Are Best in Multidimensional Integrals

- But wait! Consider an integral in k dimensions – for other methods, we must make a grid in each direction with $n^{1/k}$ points for each coordinate.
- Thus, error for Trapezoidal is $\propto 1/n^{2/k}$ which has a slower n dependence as k increases.
- However, Monte Carlo method still keeps its $1/\sqrt{n}$ behavior and for $k \geq 5$, converges faster than Trapezoidal method.
- The other methods appear better than Monte Carlo until larger k.
- However, in practice, this is not so because there the implicit condition $n^{1/k} >> 1$ before the error formulae even apply.
- For instance, $n^{1/k} = 10, k = 5$ gives large (10^5) values of n while $n^{1/k} = 1000$ gives ridiculous values for n (10^{15}) with $k = 5$.
- This example shows that the problem of multidimensional integrals is not a trivial generalization of that in one dimension.

Best Multidimensional Integration Formulae

- You can pose general problem of best method in k dimensions: the answer is not known in general and is probably not either Monte Carlo <u>or</u> Gaussian for each coordinate.
- Once k is bigger than around 3, Monte Carlo is usually best pragmatic choice. The distribution of points in multidimensional integrals is contrasted in figures MC.1 and MC.2.

Distribution of Points in Two-dimensional Integral Done by Newton-Cotes Style Formulae

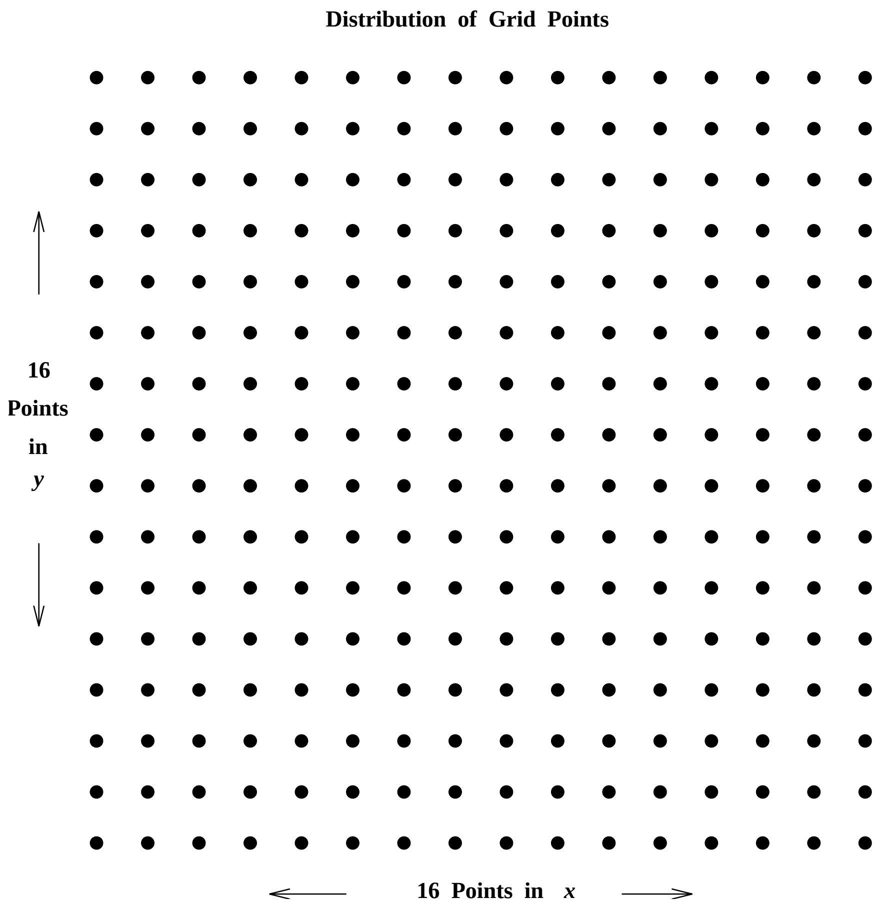


Figure MC.1:

Two-dimensional integral done by Newton-Cotes style formulae

Distribution of Points in Two-dimensional Integral Done by Monte Carlo

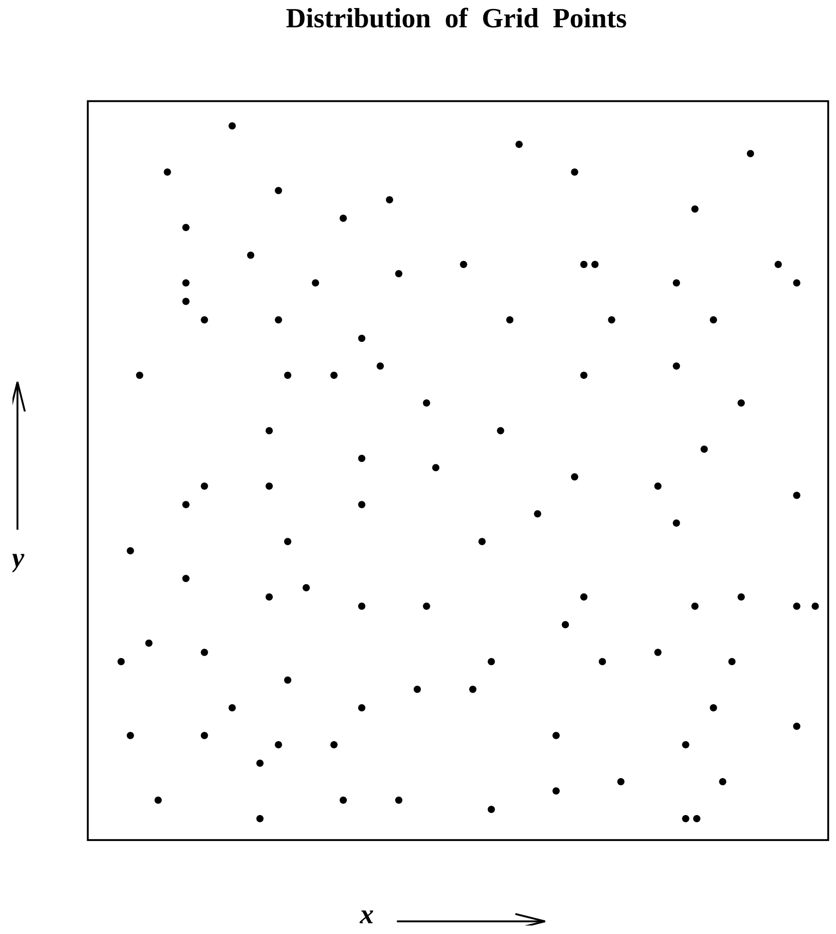
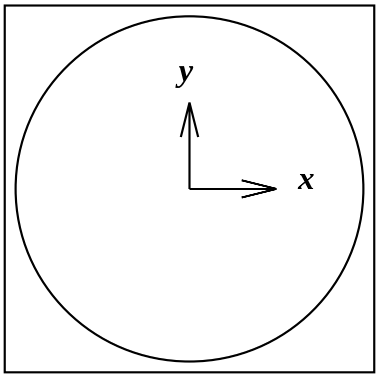


Figure MC.2:

Two-dimensional integral done by Monte Carlo

Use of Bounding Boxes to Calculate π — I

- (ii) Complicated Domains
- One advantage of the Monte Carlo method is that it can very easily take account of complicated integration domains.
- We illustrate this first with a simple example—calculation of π.



- Suppose we wished to calculate the integral over the inside of the circle

$$I = \int_{x^2 = y^2 \leq 1} g(x, y)\, dx\, dy.$$

- One way of doing this is to put

$$\begin{aligned} g'(x, y) \quad &= g(x, y) \qquad & x^2 + y^2 \leq 1 \\ &= 0 & x^2 + y^2 > 1 \end{aligned}$$

Use of Bounding Boxes to Calculate π — II

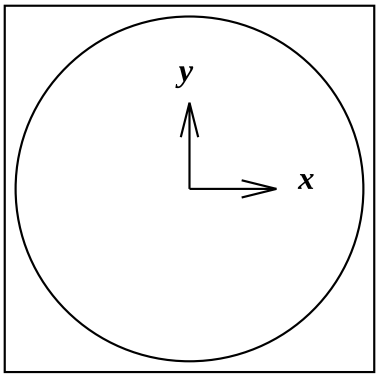
- And calculate

$$I = \int_{-1}^{1} dx \int_{-1}^{1} dy\, g'(x, y)$$

  which is an elementary Monte Carlo integral with x and y generated independently between -1 and 1.

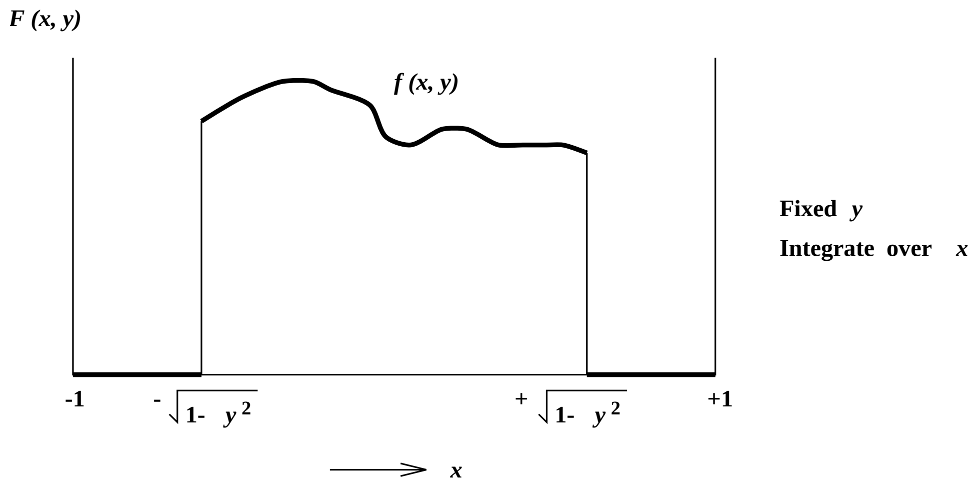
- Actually, the brute force method (generate x uniformly between -1 and 1, y uniformly between $\pm\sqrt{1-x^2}$) is more efficient—in the language of (NI.16) this second method has a smaller variance.

- However, use of bounding boxes is an important technique for it is often impossible to find ranges analytically while it is easy to decide if a given (x, y) lies within range of integration.



Use of Bounding Boxes in Complicated Geometries — I

- Example: Integrate $g(x,y)$ over region $h(x,y) \geq 0$.
  $$h(x,y) = x^3 + 3xy^2 + 23y^3$$
  or something complicated like this.
- It is indeed easy to decide if $h(x,y) \geq 0$ or < 0 for a given value (x,y).
- However, it is hard to solve equation $x^3 + 3xy^2 + 23y^3 = 0$ (the boundary of domain).
- Note you cannot use Simpson/Gauss/Trapezoidal to integrate
  $$\int_{-1}^{+1} \int_{-1}^{+1} dx\, dy\, F(x,y)$$
  where
  $$\begin{aligned} F(x,y) &= f(x,y) & x^2 + y^2 \leq 1 \\ &= 0 & x^2 + y^2 > 1 \end{aligned}$$



Use of Bounding Boxes in Complicated Geometries — II

- $F(x, y)$ is discontinuous at $x = \pm\sqrt{1 - y^2}$. This violates assumptions in analytic methods which require $F(x, y)$ be approximated by a linear (trapezoidal), quadratic (Simpson) or higher order polynomial.
- This assumption is violated by a function $F(x, y)$ which is discontinuous. Even if $f(x, y)$ happens to be zero at $x^2 + y^2 = 1$, it would rarely have required zero derivatives (of higher order for Gauss methods).
- On the other hand, Monte Carlo method only requires finite standard deviation. If has NO requirement on continuity of values; derivatives need not even exist.
- Thus, only Monte Carlo methods can use this "bounding box" approach to complicated integration domains. In practice, the latter are very common.

IMPORTANCE Sampling

Basic Theory

- (iii) Of course, it is obvious from (NI.16) that Monte Carlo integration gets better as standard deviation σ gets smaller.
- Using definition of σ^2 (or remembering its name "variance"), it follows that method is best when $f(x)$ is slowly varying over domain.
- For instance, if $f(x) \equiv$ constant, $\sigma = 0$ and method is exact.
- Importance sampling ("sampling where integrand important") is a technique for reducing variation of $f(x)$.
- Thus if

$$
\begin{array}{lrcl}
 & I & = & \int f(x)\,dx \\
\text{change variables} & y & = & q(x)\,, \quad dy = q'(x)\,dx \\
\text{put} & \dfrac{f(x)}{q'(x)} & = & F(y) \\
\text{then} & I & = & \int F(y)\,dy.
\end{array}
$$

- We now apply Monte Carlo method generating y uniformly.

Choice of Importance Sampling Weight Function — I

- We try to choose $q'(x)$ so $f(x)/q'(x)$ has a smaller standard deviation than $f(x)$.
- The best choice is

$$q'(x) = f(x)$$

- which implies
  - 1. $F(y)$ is constant with zero standard deviation
  - 2. Probability of getting a particular x value is $\propto q'(x) = f(x)$. That is, one "chooses" x values to congregate where function f is large.
- This is, of course, not realistic as to find $q'(x) = f(x)$ requires solving

$$q(x) = \int_a^x f(x)\, dx$$

(i.e., solving indefinite integral and that's harder than definite integral which is $q(b)$).

Choice of Importance Sampling Weight Function — II

- In general, one chooses $q'(x)$ so that
  - a) $q'(x)$ is "approximately" $f(x)$
  - b) $\int q'(x)$ can be performed easily.
- Example: $f(x) = e^{-x} \quad (\otimes$ a mess) where mess slowly varying.
  - Choose

$$\begin{aligned} q'(x) &= e^{-x} \\ q(x) &= 1 - e^{-x} \end{aligned}$$

  - Then $F(y) =$ "a mess".

Monte Carlo Approach to Discrete Integrals

- (iv) Discrete Integrals
- One can apply Monte Carlo methods to sums as well as continuous integrals. For example, suppose we have to calculate

$$\int dx_a \, dx_b \, dx_c \, \ldots \sum_{l=1}^{2} f_l \, (x_a, x_b, x_c \ldots)$$

- Then generate using normal Monte Carlo approach with random numbers to generate x_a, x_b, x_c.
- Now, generate another random number r with $0 \leq r \leq 1$ uniformly and then code

$$\begin{aligned} \text{if} \;\; (r < 0.5) \; l &= 1 \\ \text{else} \;\; l &= 2 \end{aligned}$$

- and obvious modifications for other discrete summations.

Why Use Monte Carlo for Summations?

- This can have important consequences. Suppose our integral involves

$$S = \sum_{l=1}^{1000} f_l\left(x_a, x_b, x_c\right)$$

- Then if we direct calculate the sum, it has computational cost, which is 1000 times that of calculating f_l.
- Suppose we generate r and set $l_r = 1 + \mathrm{INT}(1000 \times r)$ ($\mathrm{INT}(y)$ gives you integer part of floating point number y).
- And then let $S = 1000 \times f_{l_r}\left(x_a, x_b, x_c\right)$ where x_a, x_b, x_c chosen by continuous Monte Carlo procedure.

Example of using Monte Carlo for Summations

- Then computational complexity of whole problem is reduced by a factor of 1000! The price, of course, is that error is increased by variance (over l) of f_l.
- This variance is added in to variance of anything else in integrand including variance of f_l over x_a, x_b, x_c which is already present.
- Thus, we can increase total number of points n (where x_a, x_b, x_c, l are chosen separately n times) by 1000 and reduce total error by $\sqrt{1000} \sim 32$ if we generate l randomly rather than using full calculation of sum.
- This strategy is often a big win!
- See following discussion for a related comment.

The Wrong Way to Perform Multiple Monte Carlo

- (v) When one does multiple integrals $\int dx \int dy\, f(x,y)$ by Monte Carlo and chooses a set of n points $\underline{x}_i = (x_i, y_i)$ where for each i, one chooses separately x_i, y_i randomly. This gives $\underline{x}_i$ distributed randomly over the plane.
- It would be very wrong to view as in Newton Cotes as an integral over x of $F(x) = \int dy\, f(x,y)$ where we think of doing x and y integrals by Monte Carlo.
- This incorrect approach gives a set of points x_i, y_i where one first chooses x_1 randomly, then fixing this value of x_1, chooses $n_y \sim \sqrt{n}$ random values of y; then change to a new random x and so on.
- This scenario gives use to a set (x_i, y_i) where

$$\begin{aligned} x_1 &= x_2 \ldots = x_{n_y} \\ x_{n_y+1} &= x_{2n_y} \text{ etc.} \end{aligned}$$

- In this case error is proportional to $1/\sqrt{n_x} \sim 1/n^{1/4}$ and NOT $1/\sqrt{n}$ where all x_i and all y_i are chosen independently.

Stock Market Example of Multiple Monte Carlos — I

- You may think of this as obvious, but let us study its consequences in a generalized example. We wish to make money on the stock market using a Monte Carlo model of their behavior on certain variables $I = \int f\,(x_1, x_2 \ldots)\; dx_1 \ldots$.
- We need an accuracy of say 1%. Then we need a number of events n_f where $\sigma_f / n_f^{1/2} \sim 0.1 \langle f \rangle$ where desired integral is up to a constant $\langle f \rangle = \mu_f$. Then

$$n_f = 10000 \left(\frac{\sigma_f}{\mu_f} \right)^2$$

- Now suppose your friend had another set of variables whose Monte Carlo $J = \int g\,(y_1, y_2, \ldots)\; dy_1, dy_2$ required

$$n_g = 10000 \left(\frac{\sigma_g}{\mu_g} \right)^2$$

Stock Market Example of Multiple Monte Carlos — II

- Now we realize that we need to combine the two approaches and integrate over both x_i and y_j!
- Then naively this is hopeless as it requires

$$n = n_f \cdot n_g = 10^8 \left(\frac{\sigma_f \, \sigma_g}{\mu_f \, \mu_g} \right)^2 \text{ points.}$$

- But this is regarding joint Monte Carlo independently, but rather we should think about problem as

$$K = \int f(x_1 \ldots) \, g(y_1 \ldots) \, dx_1 \ldots dy_1 \ldots$$

where fresh x_i, y_j are chosen each point.

Number of Points Needed in Joint Monte Carlo

- Then assuming special case written above where integrand has product form $f\,g$ and f and g are independent, we need

$$n = n_k = 10^4 \left(\frac{\sigma_{fg}}{\mu_f\,\mu_g}\right)^2 \text{ points}$$

$$\sigma_{fg}^2 = \sigma_f^2\,\sigma_g^2 + \mu_f^2\,\sigma_g^2 + \mu_g^2\,\sigma_f^2$$

  as can easily be proven.

$$\frac{\sigma_{fg}^2}{\mu_f^2\,\mu_g^2} \sim \frac{\sigma_f^2}{\mu_f^2} + \frac{\sigma_g^2}{\mu_g^2}.$$

- We see that we need NOT $n_k = n_f \cdot n_g$ points, but rather $n_k = n_f + n_g$ points!
- The exact calculation requires one to relax the unrealistic assumption that f and g are independent, but the point is general and the estimate of n_k is reasonably accurate!

Accept/Reject Method for Generating General Probability Distributions

- (vi) Accept/Reject Method
- This is an important practical method for generating random numbers according to particular probability distributions, which cannot be integrated so that $\int_{x_1}^{x} p(x)$ can be used as described earlier.
- We wish to generate x according to a given probability distribution $p(x)$ in range $x_1 \leq x \leq x_2$.
- Put
$$p_{\max} = \max_x p(x)$$
Let $w_{\max}$ be (any number) $\geq p_{\max}$.
- Generate x uniformly in $[x_1 x_2]$. (*)
Generate r uniformly in [01].
$$\text{If } r > \frac{p(x)}{w_{\max}} \text{ reject } x \text{ and start again at (*)}$$
$$r \leq \frac{p(x)}{w_{\max}} \text{ accept } x$$

Estimate of Maximum in Accept/Reject Method

- The resultant accepted x is indeed distributed according to $p(x)$.
- Often, it is hard to find $p_{\max}$—remember we might apply method to $p(x_1, x_2, \ldots x_r)$ with a very complicated formula for p. One can estimate a $w_{\max}$ sometimes.
- The method works for any $w_{\max} \geq p_{\max}$ but if $w_{\max}$ is chosen as twice the real $p_{\max}$, then one unnecessarily rejects half of the possible events.
- The method works, but is inefficient.
- Actually, one can use $w_{\max} < p_{\max}$ if either
  - weight events by $p(x)/w_{\max}$ if $p(x) > w_{\max}$
  - just smile and accept error if doesn't happen often

Introduction to Metropolis Method

- (vii) Metropolis Method
- This very important method was introduced by N. Metropolis in *Journal of Chemical Physics*, **21**, 1087 (1953).
- It is used in nearly all large scale statistical physics simulations.
- We are again doing "importance" sampling for $\int p(\underline{x})\, f(\underline{x})\, d\underline{x}$ where we wish to choose $\underline{x}$ according to $p(\underline{x})$.
- In the simplest physics applications $p(x) = \exp\left(-(\text{Energy} = E(\underline{x}))/kT\right)$ and is (due to exponential and large energy $E(\underline{x})$) typically tiny!
- Direct application of Monte Carlo without importance sampling or use of "accept/reject" method will "never" find interesting values of $\underline{x}$ with $p(\underline{x}) \sim p_{\max}$.

The Metropolis Procedure

- The method starts with a point $\underline{x}_i$ which can choose at random or with some prejudice. We now wish to generate the next point $\underline{x}_j$.
- Choose $\underline{x}_t$ uniformly in a neighborhood of $\underline{x}_i$, i.e., $-\Delta\underline{x} + \underline{x}_i \leq \underline{x}_t \leq \Delta x + \underline{x}_i$.
- Then examine $p\left(\underline{x}_i\right)$ and $p\left(\underline{x}_t\right)$
- if $p\left(\underline{x}_t\right) \geq p\left(\underline{x}_i\right)$, accept $\underline{x}_t$ and put $\underline{x}_{i+1} = \underline{x}_t$ and
- if $p\left(\underline{x}_t\right) < p\left(\underline{x}_i\right)$, generate a new random number r uniformly in $[0, 1]$.
  - if $r < p\left(\underline{x}_t\right))/p\left(x_i\right) < 1$, set $\underline{x}_{i+1} = \underline{x}_t$
  - if $r \geq p\left(\underline{x}_t\right))/p\left(x_i\right) < 1$, set $\underline{x}_{i+1} = \underline{x}_i$
- $\underline{x}_{i+1}$ is next random point (note it may be identical to $\underline{x}_i$). Eventually, this produces a set of points distributed "according to $p(x)$".

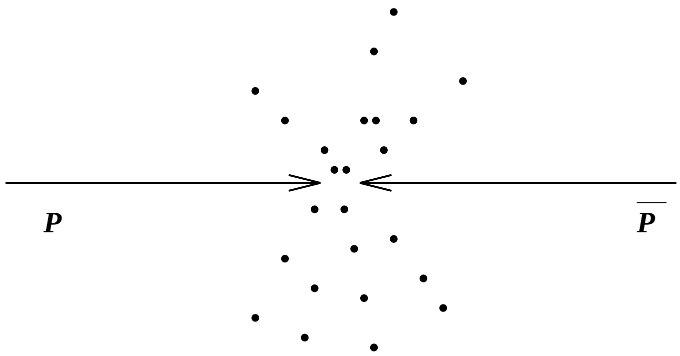
Why Metropolis Method Works

- Reasons:
  - 1. The neighborhoods $\Delta \underline{x} - \underline{x}_i \leq \underline{x}_t \leq \underline{x}_i + \Delta \underline{x}$ eventually cover the whole space.
  - 2. Take any two points $\underline{x}_i$, $\underline{x}_j$. Suppose $p\left(\underline{x}_i\right) > p\left(\underline{x}_j\right)$, i, j label states or points in system.
  - The Metropolis Method is such that probabilities of transition are $Pr(i \rightarrow j)$, which is $p\left(x_j\right)/p\left(x_i\right)$ and $Pr(j \rightarrow i)$ is 1. That is,
  $$\frac{Pr(i \rightarrow j)}{Pr(j \rightarrow i)} = \frac{p\left(x_j\right)}{p\left(x_i\right)}$$
  - 3. In equilibrium,
  $$N_i\, Pr(i \rightarrow j) = N_j\, Pr(j \rightarrow i).$$
  - 4. Therefore,
  $$\frac{N_i}{N_j} \propto \frac{p\left(x_i\right)}{p\left(x_j\right)}$$
  - 5. Here N_i and N_j are number (density) of points generated at $\underline{x}_i$ and $\underline{x}_j$.

Monte Carlo Examples

Example 1: An Experimental Physics Application

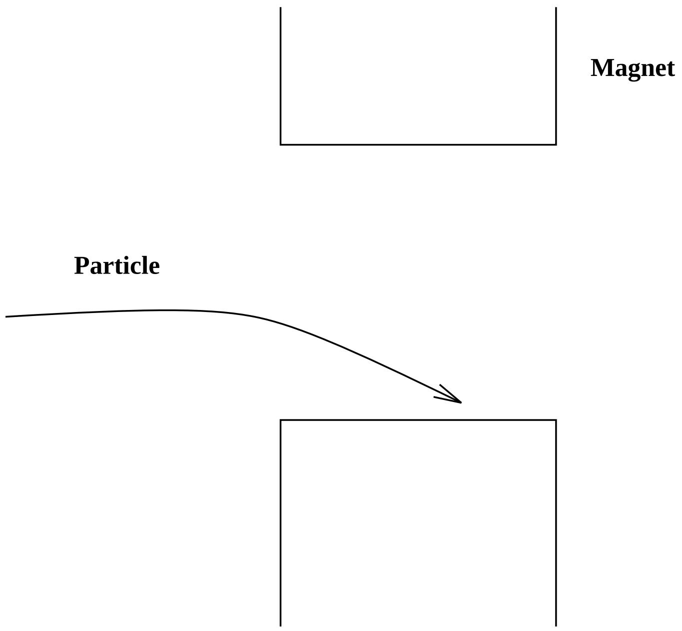
- The SSC near Dallas was to have been built at a cost of $\gtrsim$\$10B. Our experiment has looked for a Higgs Particle and after a one-year run, it has seen no candidates?
- What does this mean for the underlying knowledge of physics?
- Theory:



- Lots of Particles (including Higgs).

A High Energy Experiment Scenario

- Each particle is seen in a huge detector with some probability.



- Some particles miss apparatus. Others hit chambers/bend into magnets.
- All “multiple scatter”.
- Many produce secondary “showers” (i.e., final detection is a “mess”).
- Further theory is
  - a) An approximation;
  - b) Hard to calculate.
  - However it is “just” an integral.

An Experimental Physics Monte Carlo

- Chance of seeing a Higgs

$$
\begin{aligned}
= \int_{*} & \text{(Cross-Section(= Probability) to Produce Higgs)} \\
\otimes \int_{**} & \text{(Particles Interacting with Experimental Detector)} \qquad \text{(NI.18)}
\end{aligned}
$$

  - ***** represents all relevant theoretical parameters
  - ****** represents all possible things that happen to particles as they traverse real world for ~ 50 nanoseconds (i.e., for around 50 feet as speed of light is a nanosecond per foot!).

- This is a "Double" Monte Carlo—each Monte Carlo is a multiple integral.
- Note that "Second Monte Carlo" has lots of "non-differentiable functions" as particles are lost/detected with probability 0,1 respectively at sharp geometric boundaries.

Double Monte Carlo's Again — I

- A digression on double integrals, which we already discussed in NI.4 in general terms

$$I = \int f(x)\,dx \int g(x,y)\,dy$$

- Each integral $\int$ could be several dimensions, but write as one. If

$$\begin{aligned} \int f(x)\,dx &= I_1 \quad \text{needs } N_1 \text{ points} \\ \int g(x,y)\,dy &= I_2 \quad \text{needs } N_2 \text{ points} \end{aligned}$$

- Then for analytic method (such as Romberg), the calculation of I would need (approximately) $N_1 \cdot N_2$ points. Computer time is "multiplicative".
- What about Monte Carlo! Fractional error

$$\frac{\delta I_i}{I_i} = \left(\frac{\sigma_i}{I_i}\right) / \sqrt{N_i}$$

Double Monte Carlo's Again — II

- We can estimate this error assuming f and g are independent.

$$\begin{aligned}
\sigma_{fg}^2 &= \int \left(f^2(x)\, g^2(y) - \langle f\rangle^2 \langle g\rangle^2\right)\, dx\, dy \\
&= \int f^2(x)\, dx \int g^2(y)\, dy - \langle f\rangle^2 \langle g\rangle^2 \\
&= \left(\sigma_f^2 + \langle f\rangle^2\right) \left(\sigma_g^2 + \langle g\rangle^2\right) - \langle f\rangle^2 \langle g^2\rangle \\
\frac{\sigma_{fg}^2}{\langle f\rangle^2 \langle g\rangle^2} &= \frac{\sigma_f^2}{\langle f\rangle^2} + \frac{\sigma_g^2}{\langle g\rangle^2} \\
\left(\frac{\delta I}{I}\right)^2 &= \left(\frac{\delta I_1}{I_1}\right)^2 + \left(\frac{\delta I_2}{I_2}\right)^2
\end{aligned}$$

- NOTE: We add, not multiply errors.
- Suppose we use $N_1 = N_2 = N$ events.
- Therefore, use of $2N$ events gets the same fractional error in I as in I_1, I_2, if these each use N events and have $\sigma_f/\langle f\rangle = \sigma_g/\langle g\rangle$.
- So do NOT think of (NI.18) as a Monte Carlo over theory and for each theory (model) a Monte Carlo over experiment.
- Rather, (NI.18) is a single Monte Carlo where "random x" plus "random y" form a single random point in multidimensional space.

A Monte Carlo Event

- We call the resultant point "an event" in analogy to what happens in real world. Monte Carlo Integrals are "just" averages over events. Call
  - "f" = Cross-section to produce Higgs
  - "g" = Particles interacting with experimental detector.
- In the real world, each event has "unit weight" (God does perfect importance sampling).
- In the Monte Carlo,
  - g is typically 0 or 1, as it is in real world,
  - f is a widely varying function reflecting theoretical expectation (model).
- Calculating g is very time consuming as need to track many (100's at SSC) particles through apparatus.
- Also, one typically has several different g's to calculate for each f.
- These different g's are the responses of different parts of apparatus.

Uniform Weight Events

- This seems a terrible "waste" if weight $f(\underline{x})$ (particular event) small compared to average, i.e.,

$$\frac{f\left(\underline{x}\right)}{\langle f \rangle} \text{ small.}$$

- So we use "accept-reject" method, accepting event $\underline{x}$ with probability

$$\frac{f\left(\underline{x}\right)}{\max_{\underline{x}} f\left(\underline{x}\right)\big|_{\text{guess/correct}}}$$

- In practice, often hard to find and one guesses the maximum—often incorrectly (i.e., less than true maximum).

What Happens If You Miscalculate Maximum Weight in Accept/Reject Approach

- It is possible to cope with an $f\left(\underline{x}\right) >$ guessed $\max_{\underline{x}} f(x)$ in two ways.
  - 1. Weight event with factor $w\left(\underline{x}\right)$ in integrand
$$w\left(\underline{x}\right) = \frac{f\left(\underline{x}\right)}{\max_{\underline{x}} f\left(\underline{x}\right)\big|_{\text{guess}}}$$
  - 2. Accept
$$\frac{f\left(\underline{x}\right)}{\max_{\underline{x}} f\left(\underline{x}\right)\big|_{\text{guess}}}$$
    copies of $\underline{x}$.
- This approach (2.) is possible even if not an integer using probability to accept > 1 particles!

Example 2: Parallel Computing for "Event" Monte Carlos

- The easiest way to use parallel computing for Monte Carlo is to generate separate events on separate processors.
- This just needs independent random number sequences on each node (we discussed in statistics module).
- These "independent" events are naturally implemented in HPF with an INDEPENDENT DO loop.
- This is often called the "farm" concept.
- A key hardware feature is that little communication is needed between processors.
- One needs to accumulate the central limit theorem sums such as $\Sigma_{\text{events } x}$ "$f(x)\,g(y)$".
- This "reduction" included in both MPI (message passing) and HPF.
- Note "loosely-coupled" systems (networks of workstations) are suitable as little communication.

Example 3: Lattice Monte Carlo

Theoretical Physics

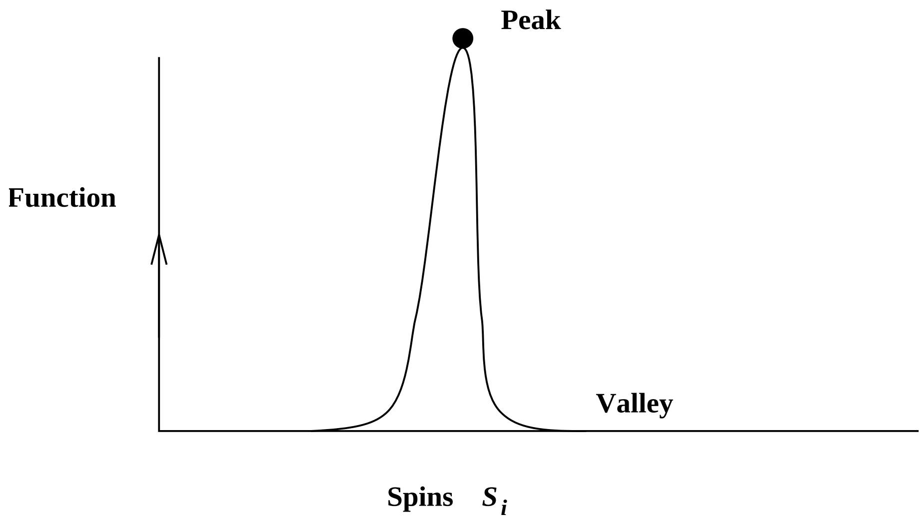
- A typical "theoretical" calculation in condensed matter or high energy physics involves integrals of form
$$I = \int \exp\left[\frac{-H}{kT}\right] \Theta \, ds_1, \ldots ds_n$$
where s_i is spin (gluon field) at site i.
- Typically, the sites are in a regular lattice in two, three, or four dimensions.
- This integral would involve solve $10^4 \rightarrow 10^6$ dimensions and so must be evaluated by Monte Carlo.
- Θ is not a very rapidly varying function (typically a polynomial in the s_i).
- Also, we want to do integral for several different choices for Θ.
- The Hamiltonian H is proportional to size of system and so the weighting function $\exp[-H/kT]$ is very small and one must be very careful to choose one's integration points appropriately.

Choice of Points in Lattice Monte Carlo

- Assume you have a particular choice $s_1^{(0)} \ldots s_n^{(0)}$ with $H^{(0)} = H\left(s_1^{(0)} \ldots s_n^{(0)}\right)$.
- Remember $\exp[-H/kT]$ is a tiny number (as H is energy of full system H/kT is approximately the number of objects in system) and more importantly, varies rapidly.
- Thus, we cannot choose points "uniformly".
- Rather, we assume point $\underline{s}^{(1)}$ "near" $\underline{s}^{(0)}$. So that $\exp[-H/kT]$ only changes a little. This can be achieved by only changing one of the spins at a time, i.e., we move $s_k^{(0)} \rightarrow s_k^{(1)}$ and $s_i^{(0)} i \neq k$ unchanged.
- Typically, this produces a change $\Delta H \sim kT$ as kT is energy per site (up to numerical constants).
- Remember classical result of energy of $1/2kT$ per degree of freedom.
- So change in $\exp[-H/kT]$ is roughly $\exp[I\, kT/kT]$ (i.e., modest).
- Thus, we will not move off "peak" of integrand.

Pictorial View of Lattice Monte Carlo Integrands

- Essential point is that integral looks like multidimensional version of figure below, with a single peak becoming a ridge in many dimensions.
- Our strategy ensures that we move along ridge keeping integrand large.
- The naive strategy of uniform s_i will always choose s_i in the valley with "zero" integrand.



Metropolis and Heat Bath Methods

- One usually considers one of two methods to change s_k.
- (a) **Metropolis** — generate $s_k^{(1)}$ uniformly about $s_k^{(0)}$ and accept or reject change based on

$$r = \exp\left[-\left[\frac{H\left(\ldots s_k^{(1)} \ldots\right)}{kT} - \frac{H\left(\ldots s_k^{(0)} \ldots\right)}{kT}\right]\right]$$

  - if $r > 1$ (energy decreased), accept $s_k^{(1)}$
  - if $r < 1$ (energy increased), accept $s_k^{(1)}$ with probability r.
- (b) **Heat Bath** — sometimes one can use "exact" method to generate a variable according to a given probability distribution.
- That is, if distribution function

$$F_k(x) = \int ds_k \exp\left(-\frac{H\left(\ldots s_k \ldots\right)}{kT}\right)$$

  can be calculated and inverted.
- Here, we view H as a function of s_k with other spins fixed.

Calculation of Observables

- In each case, (a) or (b), spins are generated according to distribution $\exp[-H/kT]$ and so values of observables are calculated as

$$\langle \Theta \rangle = \frac{1}{\underset{=N}{\underset{configurations}{\#spin}}} \sum_{l=1}^{N} \Theta \left(s_1^{(l)} \ldots s_m^{(l)} \right)$$

- If we had not been careful about generating the spin values, but say chosen them uniformly, then we must use

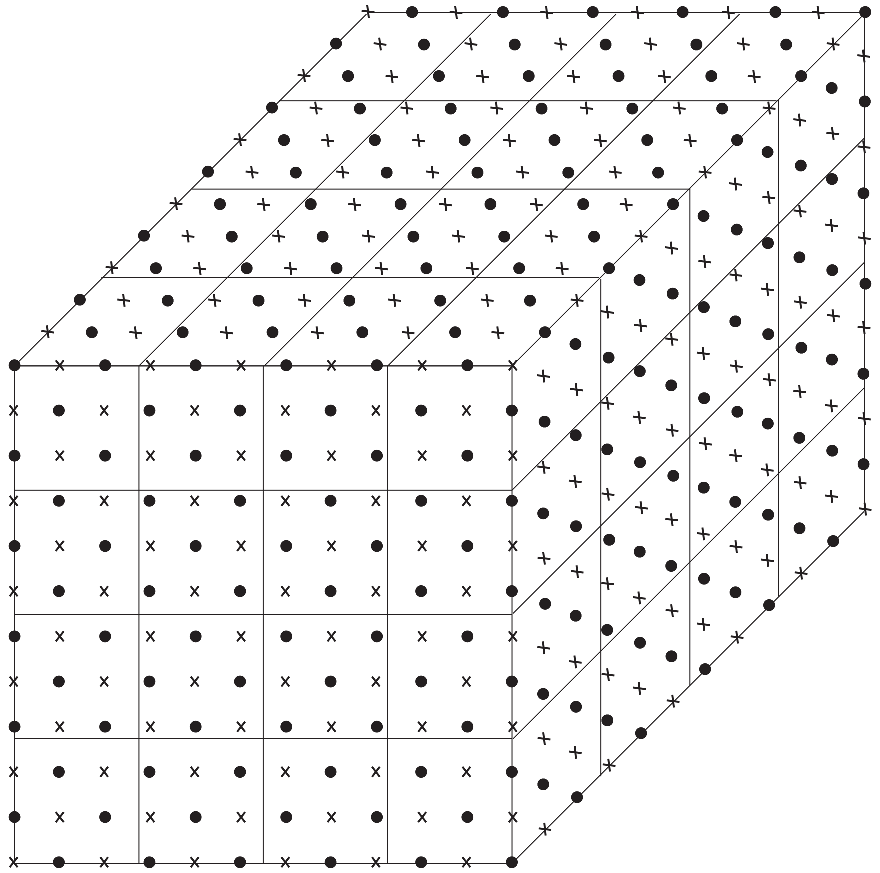
$$\langle \Theta \rangle = \frac{\sum_{l=1}^{N} \exp\left[-H[s^{(l)}]/kT\right] \, \Theta\left(s^{(l)} \ldots\right)}{\sum_{l=1}^{N} \exp\left[-H[s^{(l)}]/kT\right]}$$

- This is correct, but numerically hopeless. The wide variation in exponential (remember H proportional to size) would give a huge standard deviation in $\langle \Theta \rangle$.

Example 4: Parallel Computing for Lattice Theory

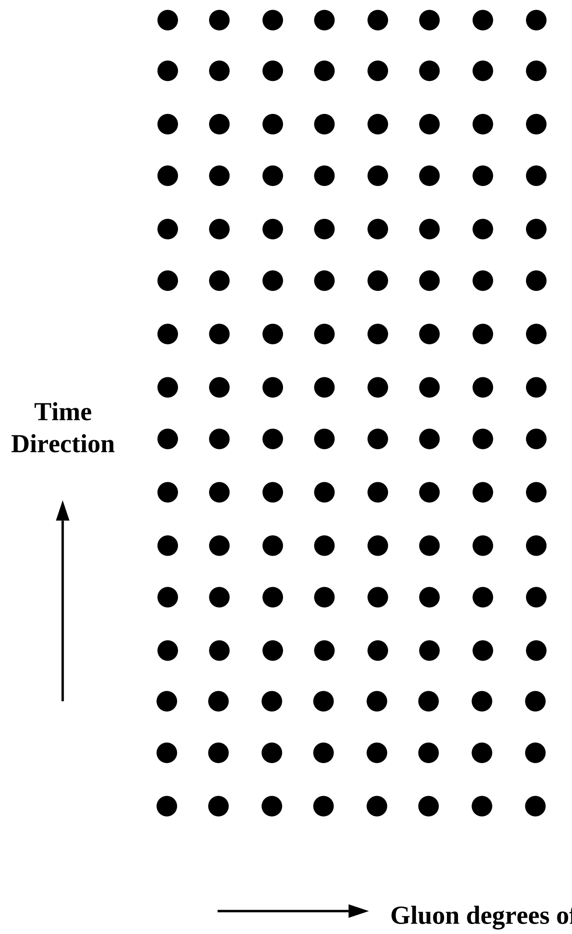
- One can in some cases do Lattice Theory as in "event" case with separate lattice's on each node of farm.
- Parallelism comes from the different events (here states of system) gotten with different random numbers on each node.
- However, this is not usually possible as system is too big to fit on a node.
- Rather, we take (two, three or four dimensional) lattice and domain decompose (i.e., get data parallelism from space on which "spins" defined).
- This leads to parallel issues similar to partial differential equations.
- The decomposition is illustrated in the following two figures.

A 12^3 Problem Lattice Decomposed Onto a 64-node Machine Arranged as a 4^3 Machine Lattice



A 12^3 Problem Lattice Decomposed onto a 64-node Machine Arranged as a 4^3 Machine Lattice. Points labeled X ("red") or • ("black") can be updated at the same time.

The 16 Time and Eight Internal Gluon Degrees of Freedom



Gluon degrees of freedom (stored as matrix elements of 3x3 SU(3) matrix over complex numbers. This has eight independent degrees of freedom after ensuring matrix is a unitary and determinant one)

The 16 time and eight internal gluon degrees of freedom stored at each point shown in previous figure.