

# Monte Carlo Methods

## Lecture I

“Nothing”

Gluon action density:  $2.4 \times 2.4 \times 3.6$  fm  
(1 fm = 1 femtometer = 1 Fermi =  $10^{-15}$  m)

Lattice simulation from  
D. B. Leinweber, hep-lat/0004025

Peter Skands

CERN - Dept of Theoretical Physics

# Topics

## Lecture 1:

Numerical Integration  
Monte Carlo methods  
Importance Sampling  
The Veto Algorithm

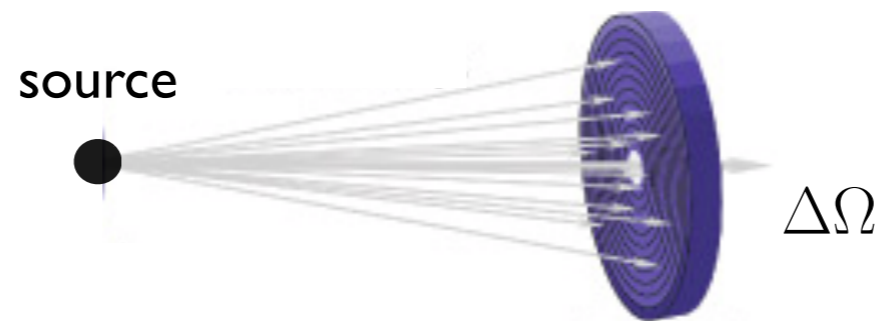
**+ This afternoon**  
**Practical Exercises:**  
PYTHIA 8 kickstart  
([check the instructions](#))

## Lecture 2:

Application of these methods to simulations of particle physics: **Monte Carlo Event Generators**

# Why Integrals?

## Scattering Experiments



LHC detector  
Cosmic-Ray detector  
Neutrino detector  
X-ray telescope  
...

→ Integrate interaction cross sections over specific regions

Predicted number of counts  
= integral over solid angle

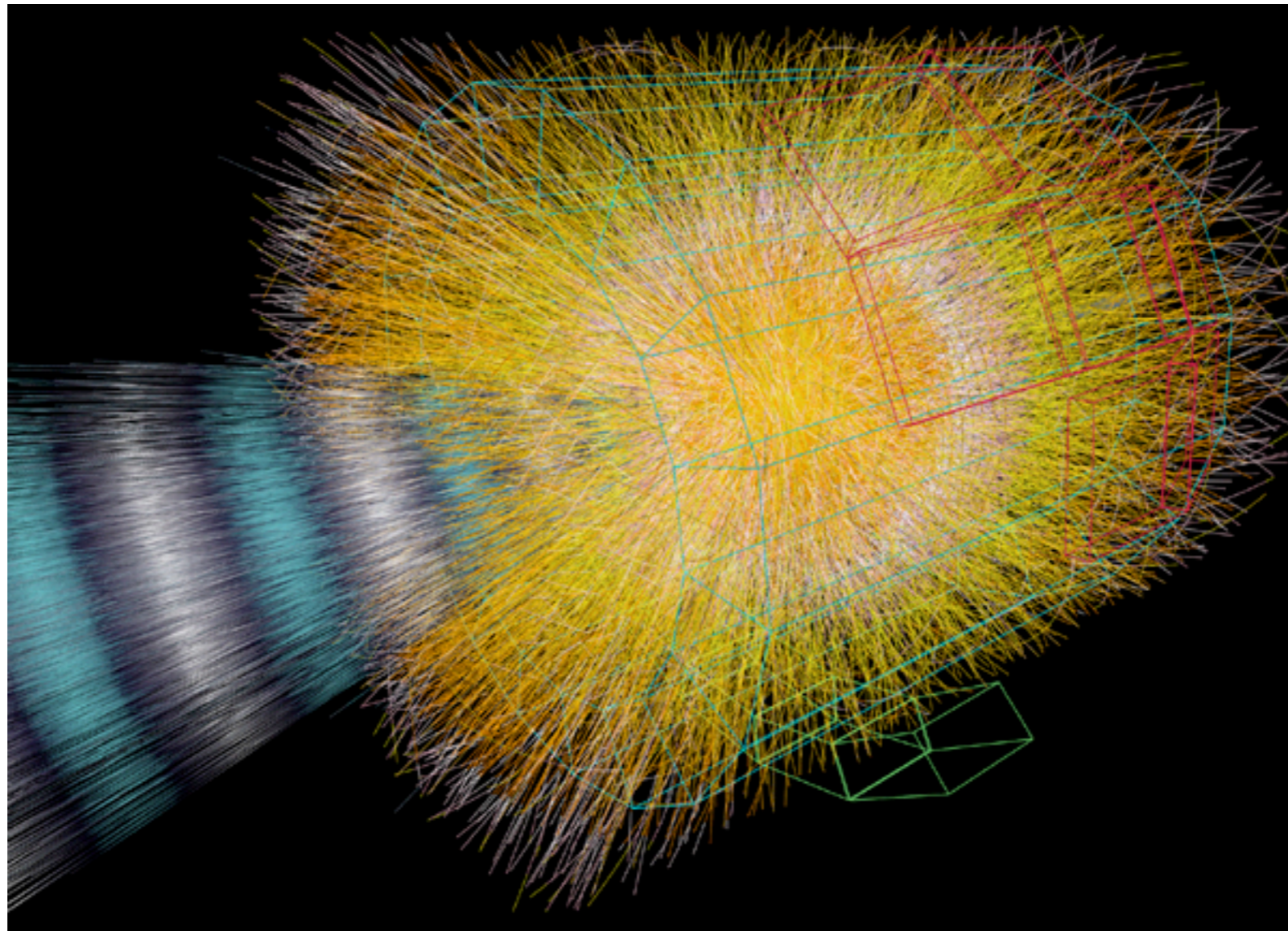
$$N_{\text{count}}(\Delta\Omega) \propto \int_{\Delta\Omega} d\Omega \frac{d\sigma}{d\Omega}$$

Differential solid angle element  $d\Omega = \sin\theta d\theta d\phi$

Differential scattering cross section  $d\sigma$  ( $\sim$  differential scattering probability / interaction probability / ...)

# Particle Physics Example

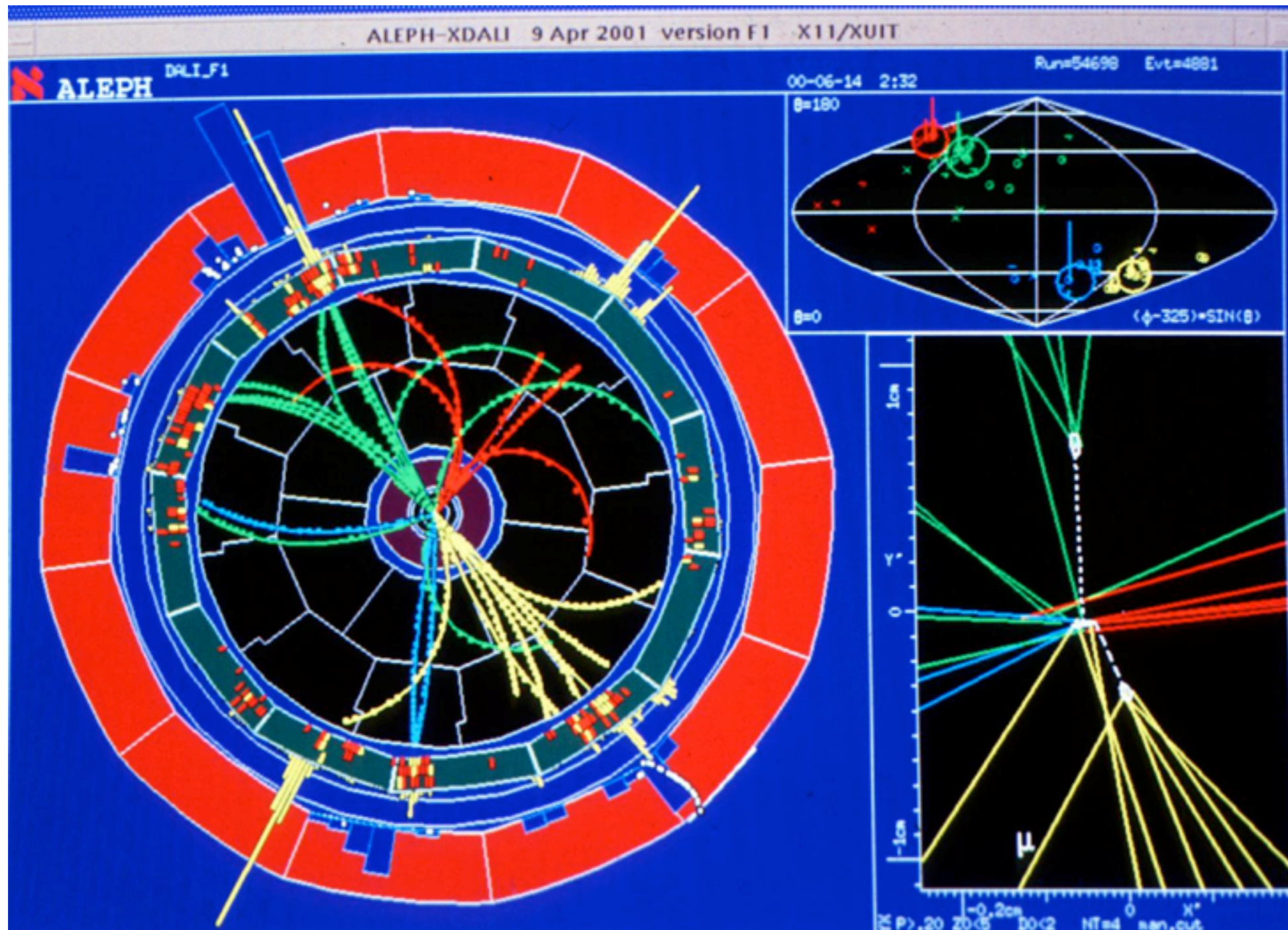
ALICE : One of the 4 experiments at the Large Hadron Collider at CERN



↪ More complicated integrals ...

# Why Numerical?

Let's look at something simpler ...



14 Jun 2000:  
4-jet event in  
ALEPH at LEP  
(a Higgs  
candidate)

Now compute  
the  
backgrounds ...

# Why Numerical?

## Part of $Z \rightarrow 4$ jets ...

### 5.3 Four-parton tree-level antenna functions

The tree-level four-parton quark-antiquark antenna contains three final states: quark-gluon-gluon-antiquark at leading and subleading colour,  $A_4^0$  and  $\tilde{A}_4^0$  and quark-antiquark-quark-antiquark for non-identical quark flavours  $B_4^0$  as well as the identical-flavour-only contribution  $C_4^0$ . The quark-antiquark-quark-antiquark final state with identical quark flavours is thus described by the sum of antennae for non-identical flavour and identical-flavour-only. The antennae for the  $q\bar{q}g\bar{g}$  final state are:

$$A_4^0(1_q, 3_g, 4_g, 2_{\bar{q}}) = a_4^0(1, 3, 4, 2) + a_4^0(2, 4, 3, 1), \quad (5.27)$$

$$\tilde{A}_4^0(1_q, 3_g, 4_g, 2_{\bar{q}}) = \tilde{a}_4^0(1, 3, 4, 2) + \tilde{a}_4^0(2, 4, 3, 1) + \tilde{a}_4^0(1, 4, 3, 2) + \tilde{a}_4^0(2, 3, 4, 1), \quad (5.28)$$

$$a_4^0(1, 3, 4, 2) = \frac{1}{s_{1234}} \left\{ \frac{1}{2s_{13}s_{24}s_{34}} [2s_{12}s_{14} + 2s_{12}s_{23} + 2s_{12}^2 + s_{14}^2 + s_{23}^2] \right. \\ + \frac{1}{2s_{13}s_{24}s_{134}s_{234}} [3s_{12}s_{34}^2 - 4s_{12}^2s_{34} + 2s_{12}^3 - s_{34}^3] \\ + \frac{1}{s_{13}s_{24}s_{134}} [3s_{12}s_{23} - 3s_{12}s_{34} + 4s_{12}^2 - s_{23}s_{34} + s_{23}^2 + s_{34}^2] \\ + \frac{3}{2s_{13}s_{24}} [2s_{12} + s_{14} + s_{23}] + \frac{1}{s_{13}s_{34}} [4s_{12} + 3s_{23} + 2s_{24}] \\ + \frac{1}{s_{13}s_{134}^2} [s_{12}s_{34} + s_{23}s_{34} + s_{24}s_{34}] \\ + \frac{1}{s_{13}s_{134}s_{234}} [3s_{12}s_{24} + 6s_{12}s_{34} - 4s_{12}^2 - 3s_{24}s_{34} - s_{24}^2 - 3s_{34}^2] \\ + \frac{1}{s_{13}s_{134}} [-6s_{12} - 3s_{23} - s_{24} + 2s_{34}] \\ + \frac{1}{s_{24}s_{34}s_{134}} [2s_{12}s_{14} + 2s_{12}s_{23} + 2s_{12}^2 + 2s_{14}s_{23} + s_{14}^2 + s_{23}^2] \\ + \frac{1}{s_{24}s_{134}} [-4s_{12} - s_{14} - s_{23} + s_{34}] + \frac{1}{s_{34}^2} [s_{12} + 2s_{13} - 2s_{14} - s_{34}] \\ + \frac{1}{s_{34}^2s_{134}^2} [2s_{12}s_{14}^2 + 2s_{14}^2s_{23} + 2s_{14}^2s_{24}] - \frac{2s_{12}s_{14}s_{24}}{s_{34}^2s_{134}s_{234}} \\ + \frac{1}{s_{34}^2s_{134}} [-2s_{12}s_{14} - 4s_{14}s_{24} + 2s_{14}^2] \\ + \frac{1}{s_{34}s_{134}s_{234}} [-2s_{12}s_{14} - 4s_{12}^2 + 2s_{14}s_{24} - s_{14}^2 - s_{24}^2] \\ + \frac{1}{s_{34}s_{134}} [-8s_{12} - 2s_{23} - 2s_{24}] + \frac{1}{s_{134}^2} [s_{12} + s_{23} + s_{24}] \\ \left. + \frac{3}{2s_{134}s_{234}} [2s_{12} + s_{14} - s_{24} - s_{34}] + \frac{1}{2s_{134}} + \mathcal{O}(\epsilon) \right\}, \quad (5.29)$$

$$\tilde{a}_4^0(1, 3, 4, 2) = \frac{1}{s_{1234}} \left\{ \frac{1}{s_{13}s_{24}s_{134}s_{234}} \left[ \frac{3}{2}s_{12}s_{34}^2 - 2s_{12}^2s_{34} + s_{12}^3 - \frac{1}{2}s_{34}^3 \right] \right. \\ + \frac{1}{s_{13}s_{24}s_{134}} [3s_{12}s_{23} - 3s_{12}s_{34} + 4s_{12}^2 - s_{23}s_{34} + s_{23}^2 + s_{34}^2] \\ \left. + \frac{s_{12}^3}{s_{12}} + \frac{1}{s_{134}} \left[ \frac{1}{s_{134}} + \epsilon^2 \right] \right\}$$

This is one of the simplest processes ... computed at lowest order in the theory.

$$\frac{1}{s_{13}s_{134}s_{234}} [s_{12}s_{24} + s_{12}s_{34} + 2s_{12}^2] \\ + \frac{1}{s_{13}s_{134}(s_{13} + s_{23})} [s_{12}s_{24} + s_{12}s_{34} + 2s_{12}^2] \\ - 2s_{12}^2]$$

Now compute and add the quantum corrections ...

$$+ \frac{-s_{12}}{s_{13}(s_{13} + s_{23})(s_{14} + s_{24})(s_{13} + s_{14})} \\ + \frac{1}{s_{13}(s_{13} + s_{23})(s_{13} + s_{14})} [s_{12}s_{24} + 2s_{12}^2] \\ + \frac{1}{s_{134}s_{234}} [s_{12}s_{23} + 2s_{12}^2] \\ (5.30)$$

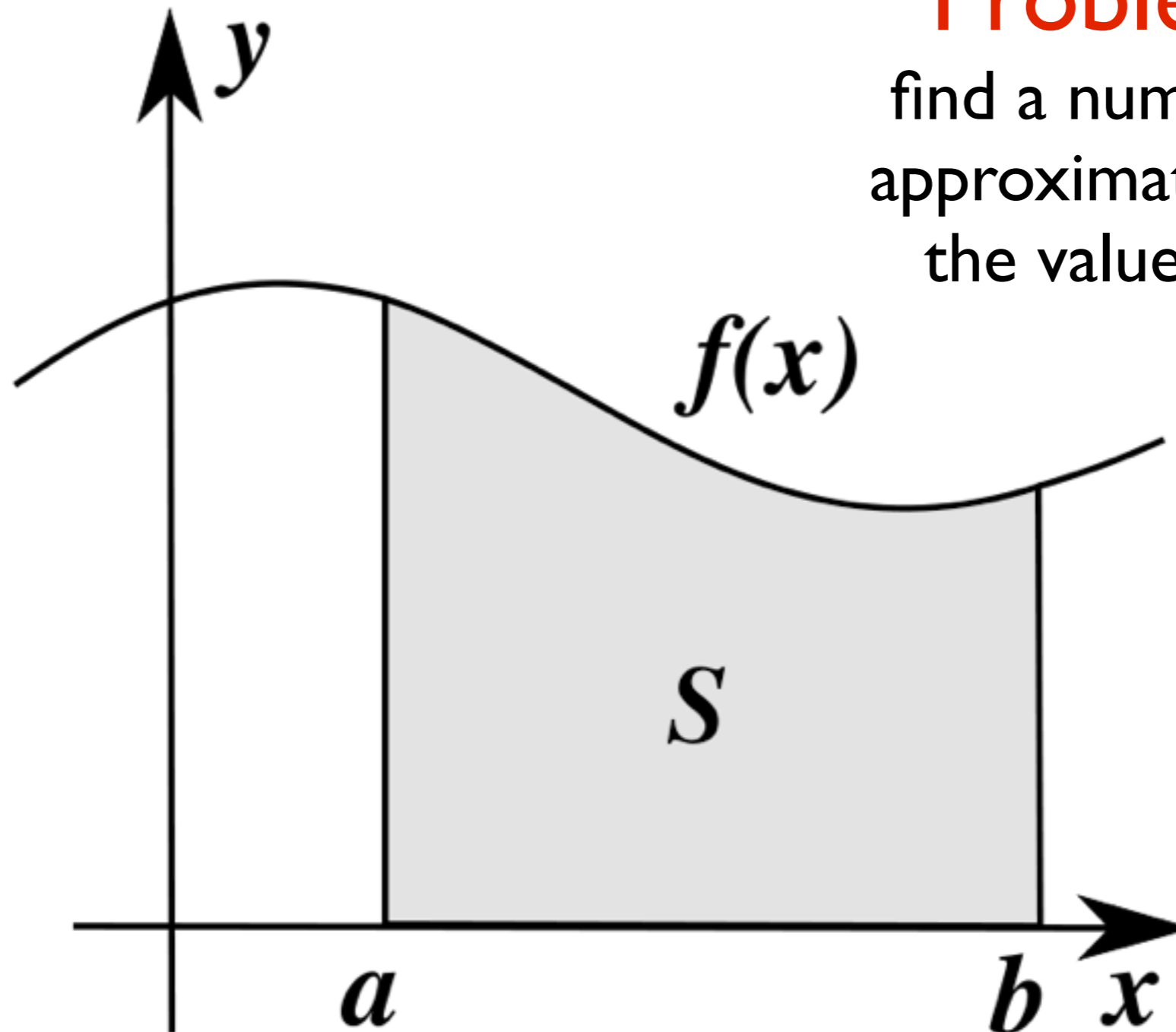
Then maybe worry about simulating the detector too ...

+ Additional Subleading Terms ...

# Numerical Integration

## Problem:

find a numerical approximation to the value of  $S$

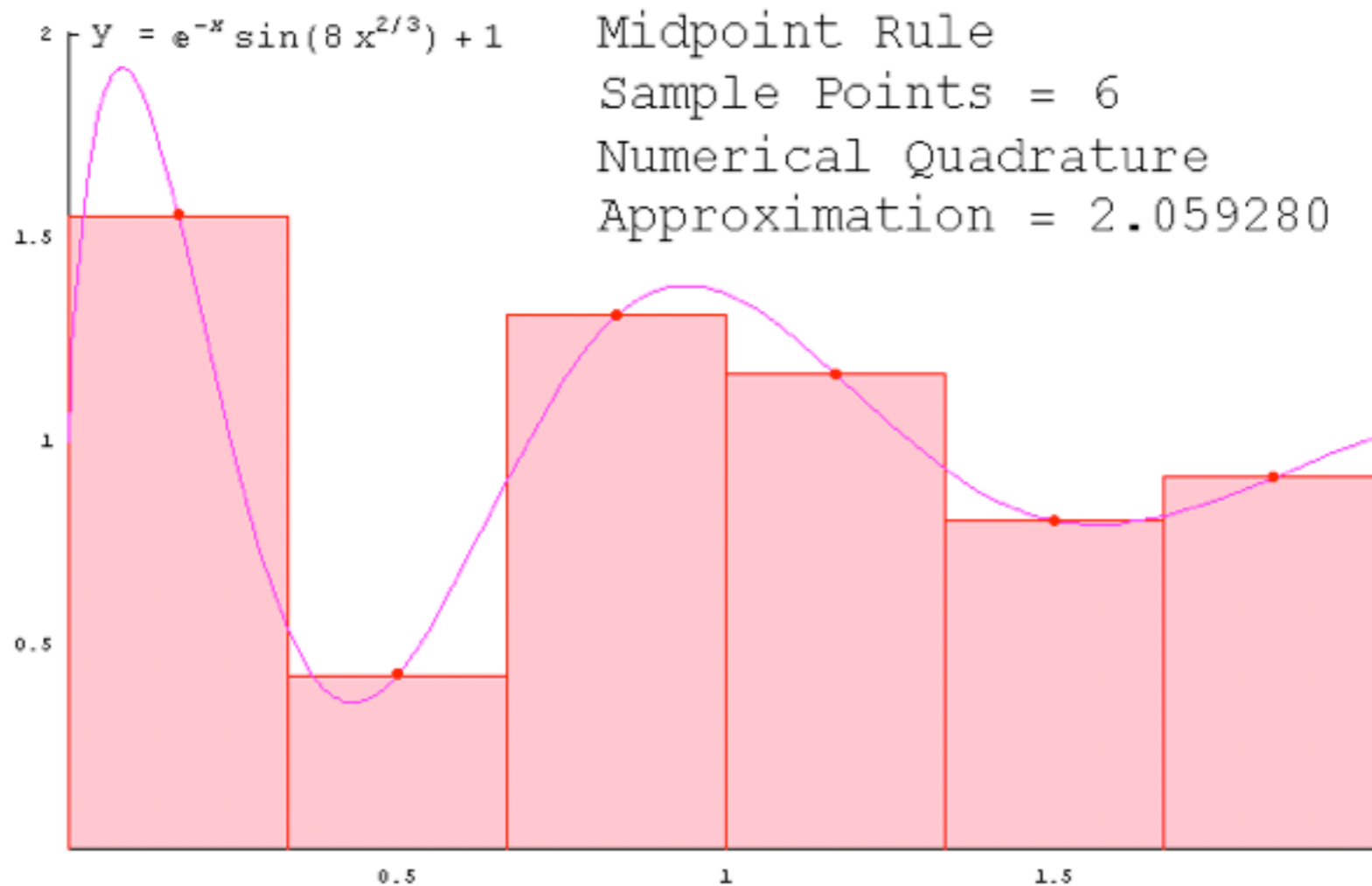


# Riemann Sums

$$\int_a^b f(x)dx = \lim_{n \rightarrow \infty} \sum_{i=1}^n f(t_i)(x_{i+1} - x_i)$$



B. Riemann,  
(1826-1866)





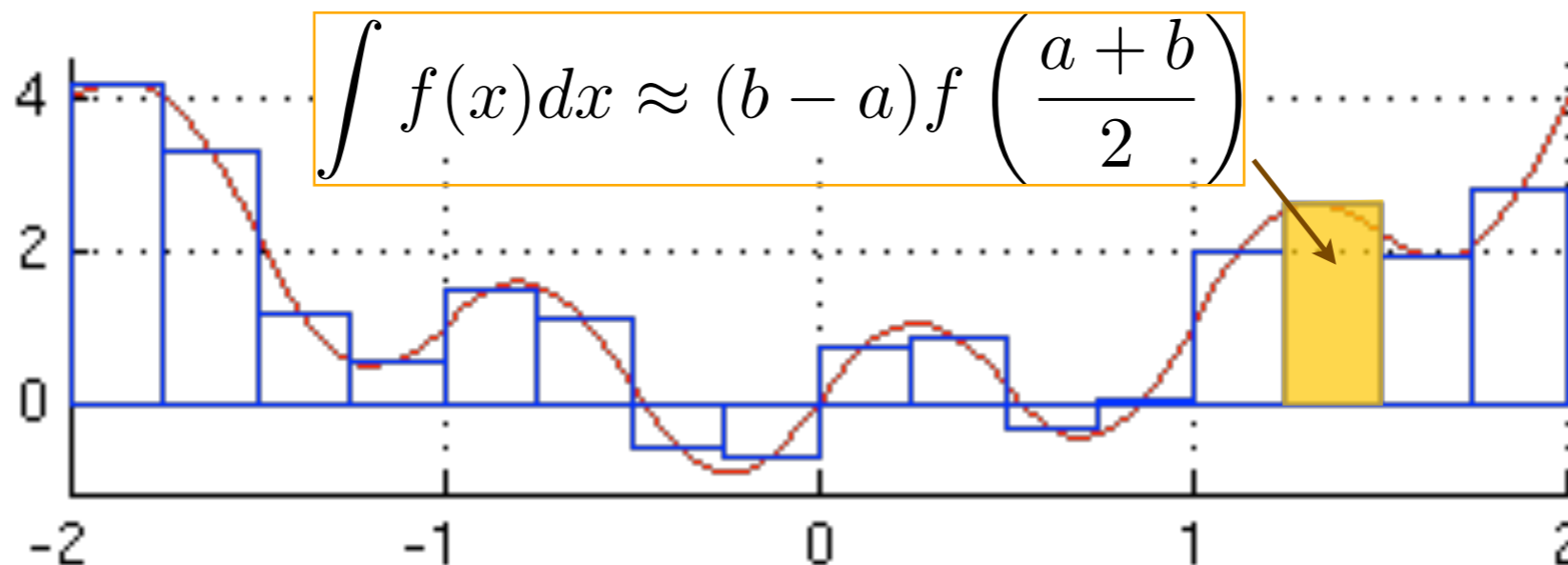
# Numerical Integration in 1D

## Midpoint (rectangular) Rule:

Fixed-Grid n-point  
Quadrature Rules

Divide into N “bins” of size  $\Delta$   
Approximate  $f(x) \approx$  **constant** in each bin  
Sum over all **rectangles** inside your region

1 function evaluation per bin



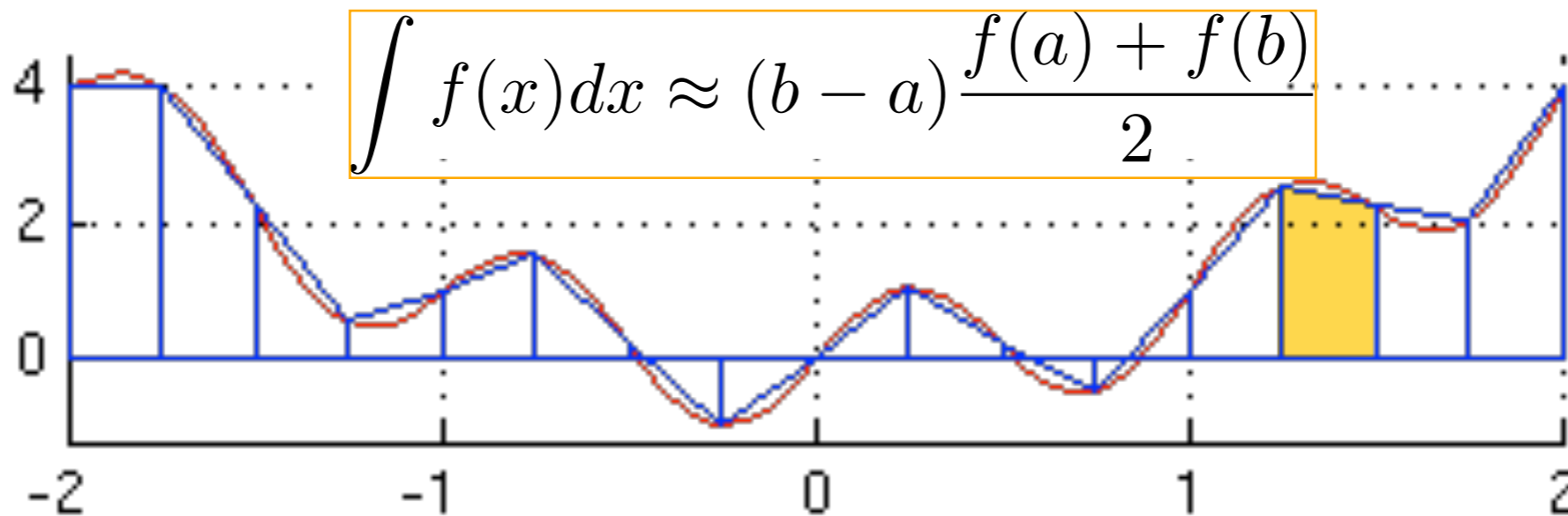
# Numerical Integration in 1D

## Trapezoidal Rule:

Fixed-Grid n-point  
Quadrature Rules

Approximate  $f(x) \approx$  **linear** in each bin  
Sum over all **trapeziums** inside your region

2 function evaluations per bin



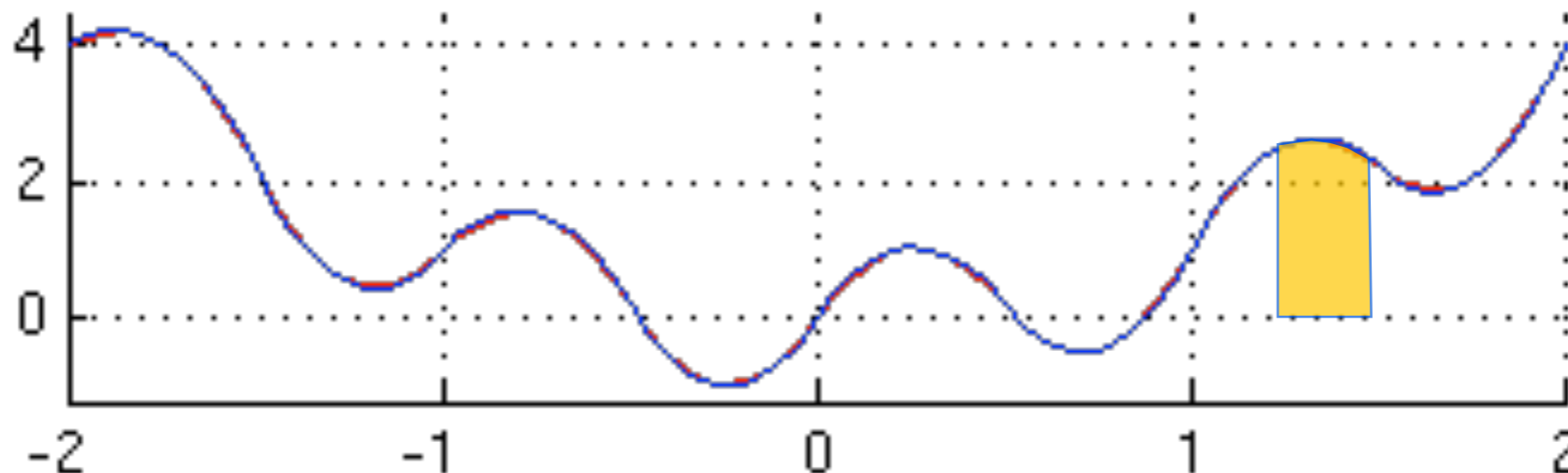
# Numerical Integration in 1D

## Simpson's Rule:

Fixed-Grid n-point  
Quadrature Rules

Approximate  $f(x) \approx$  **quadratic** in each bin  
Sum over all “**Simpsons**” inside your region

3 function evaluations per bin



... and so on for higher n-point rules ...

# Convergence Rate

## The most important question:

How long do I have to wait?

*How many evaluations do I need to calculate for a given precision?*

Uncertainty (after n evaluations)	$n_{\text{eval}} / \text{bin}$	Approx Conv. Rate (in 1D)
Trapezoidal Rule (2-point)	2	$1/N^2$
Simpson's Rule (3-point)	3	$1/N^4$
... m-point (Gauss quadrature)	m	$1/N^{2m-1}$

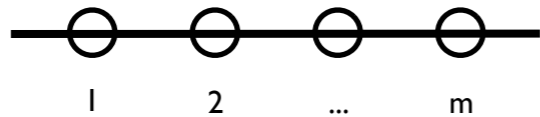
See, e.g., Numerical Recipes

See, e.g., F. James, "Monte Carlo Theory and Practice"

# Higher Dimensions

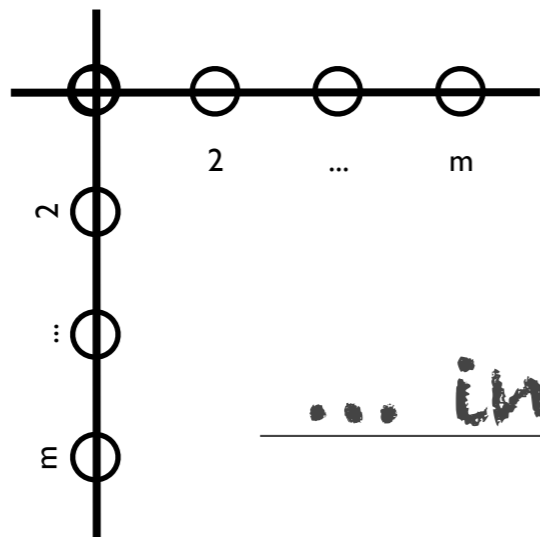
Fixed-Grid (Product) Rules scale exponentially with  $D$

## $m$ -point rule in 1 dimension



→  $m$  function evaluations per bin

## ... in 2 dimensions



→  $m^2$  evaluations per bin

... in  $D$  dimensions →  $m^D$  per bin

E.g., to evaluate a 12-point rule in 10 dimensions, need 1000 billion evaluations per bin

# Convergence Rate

+ **Convergence is slower** in higher **Dimensions!**

→ More points for less precision



Uncertainty (after n evaluations)	$n_{\text{eval}} / \text{bin}$	Approx Conv. Rate (in D dim)
Trapezoidal Rule (2-point)	$2^D$	$1/n^{2/D}$
Simpson's Rule (3-point)	$3^D$	$1/n^{4/D}$
... m-point (Gauss rule)	$m^D$	$1/n^{(2m-1)/D}$

See, e.g., Numerical Recipes

See, e.g., F. James, "Monte Carlo Theory and Practice"

A Monte Carlo technique: is any technique making use of random numbers to solve a problem

### Convergence:

**Calculus:**  $\{A\}$  converges to  $B$  if an  $n$  exists for which  $|A_{i>n} - B| < \epsilon$ , for any  $\epsilon > 0$

**Monte Carlo:**  $\{A\}$  converges to  $B$  if  $n$  exists for which the probability for  $|A_{i>n} - B| < \epsilon$ , for any  $\epsilon > 0$ , is  $> P$ , for any  $P[0 < P < 1]$

“This risk, that convergence is only given with a certain probability, is inherent in Monte Carlo calculations and is the reason why this technique was named after the world’s most famous gambling casino. Indeed, the name is doubly appropriate because the style of gambling in the Monte Carlo casino, not to be confused with the noisy and tasteless gambling houses of Las Vegas and Reno, is serious and sophisticated.”

*F. James, “Monte Carlo theory and practice”, Rept. Prog. Phys. 43 (1980) 1145*

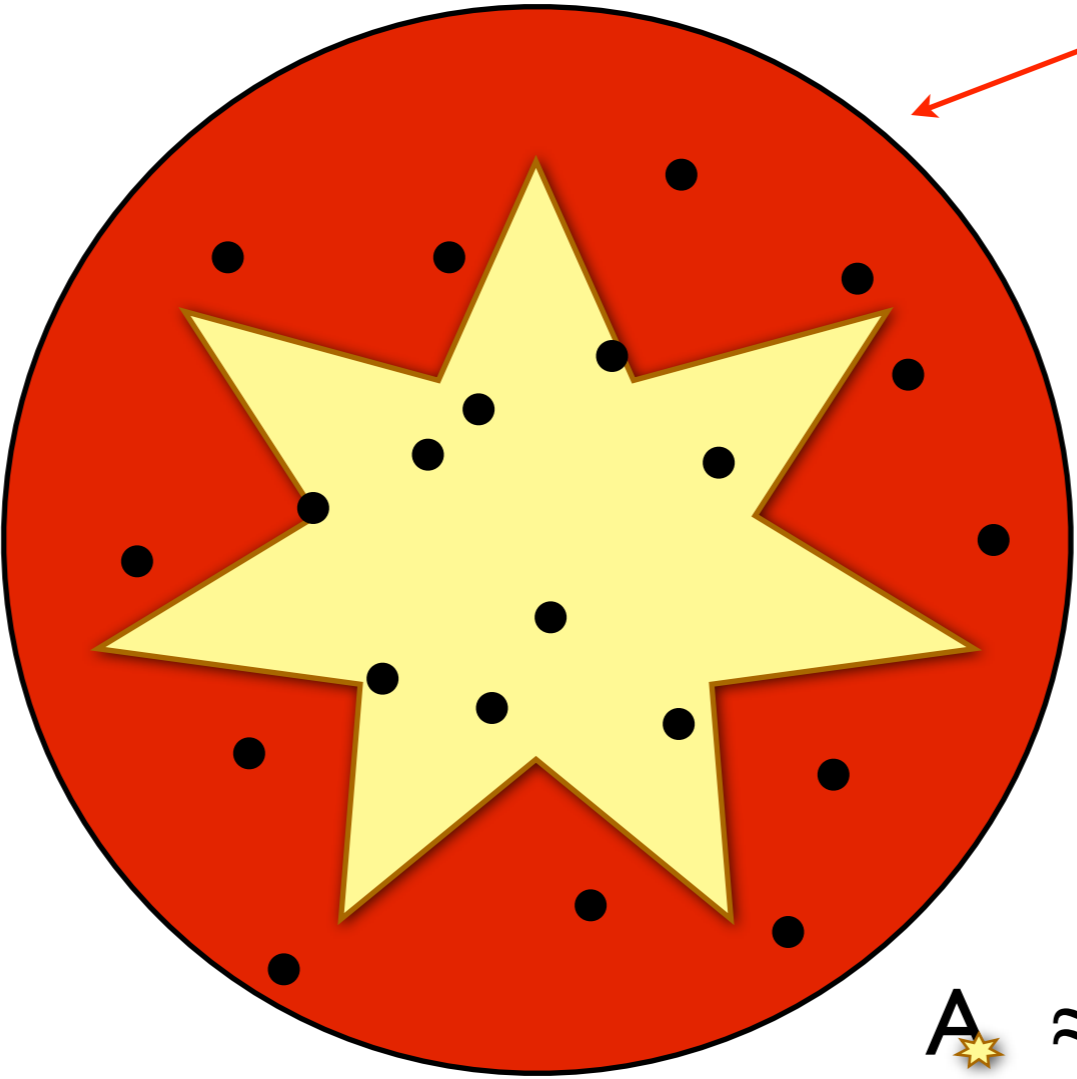
# Random Numbers and Monte Carlo

Example I: simple function (=constant); complicated boundary


**Example: you want to know the area of this shape:**

Now get a few friends, some balls, and throw random shots inside the circle  
(PS: be careful to make your shots truly random)

Count how many shots hit the shape inside and how many miss



Assume you know the area of this shape:  
 $\pi R^2$   
(an overestimate)



Earliest Example of MC calculation: Buffon's Needle (1777) to calculate  $\pi$

G. Leclerc, Comte de Buffon (1707-1788)

$$A_{\star} \approx N_{\text{hit}}/N_{\text{miss}} \times \pi R^2$$



# Random Numbers

**I will not tell you how to write a Random-number generator.** (For that, see the references at the end.)

**Instead,** I assume that you can write a computer code and link to a random-number generator, from a library

E.g., ROOT includes one that you can use if you like.

PYTHIA also includes one

From the PYTHIA 8 HTML documentation, under “Random Numbers”:

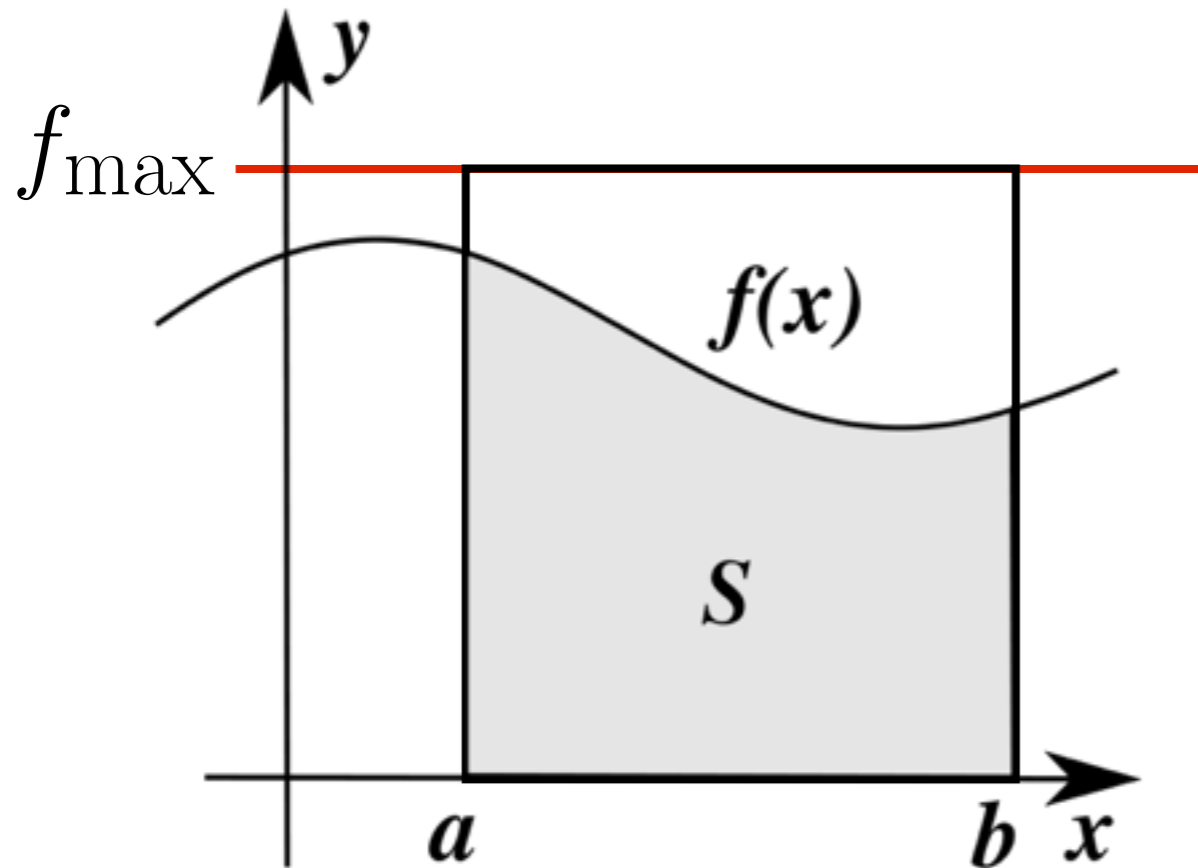
Random numbers  $R$  uniformly distributed in  $0 < R < 1$  are obtained with

```
Pythia8::Rndm::flat();
```

+ Other methods for exp,  $x^*exp$ , 1D Gauss, 2D Gauss.

# Random Numbers and Monte Carlo

Example 2: complicated function; simple boundary



The integral is then  $\approx$

Start from **overestimate**,

$$f_{\max}$$

Generate uniformly distributed random points between a and b

$$\frac{f(x_i)}{f_{\max}} = P_{\text{hit}}$$

$$(b - a) f_{\max} \frac{1}{n} \sum_{i=1}^n \frac{f(x_i)}{f_{\max}}$$

area of rectangle

fraction that 'hit'

# Justification

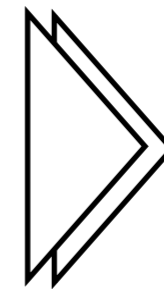
## 1. Law of large numbers

For a function,  $f$ , of random variables,  $x_i$ ,

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n f(x_i) = \frac{1}{b-a} \int_a^b f(x) dx$$

Monte Carlo Estimate

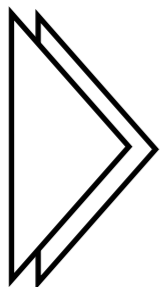
The Integral



**For infinite n:**  
Monte Carlo is a  
consistent  
estimator

## 2. Central limit theorem

The sum of  $n$  independent random variables (of finite expectations and variances) is asymptotically Gaussian  
(no matter how the individual random variables are distributed)



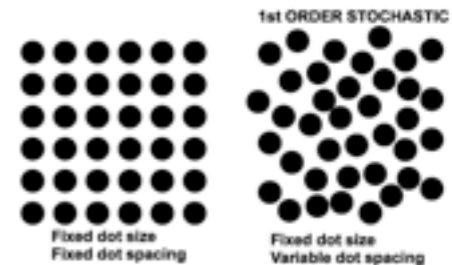
**For finite n:**  
The Monte Carlo estimate is Gauss distributed around the true value

# Convergence

MC = Monte Carlo

MC convergence is **Stochastic!**

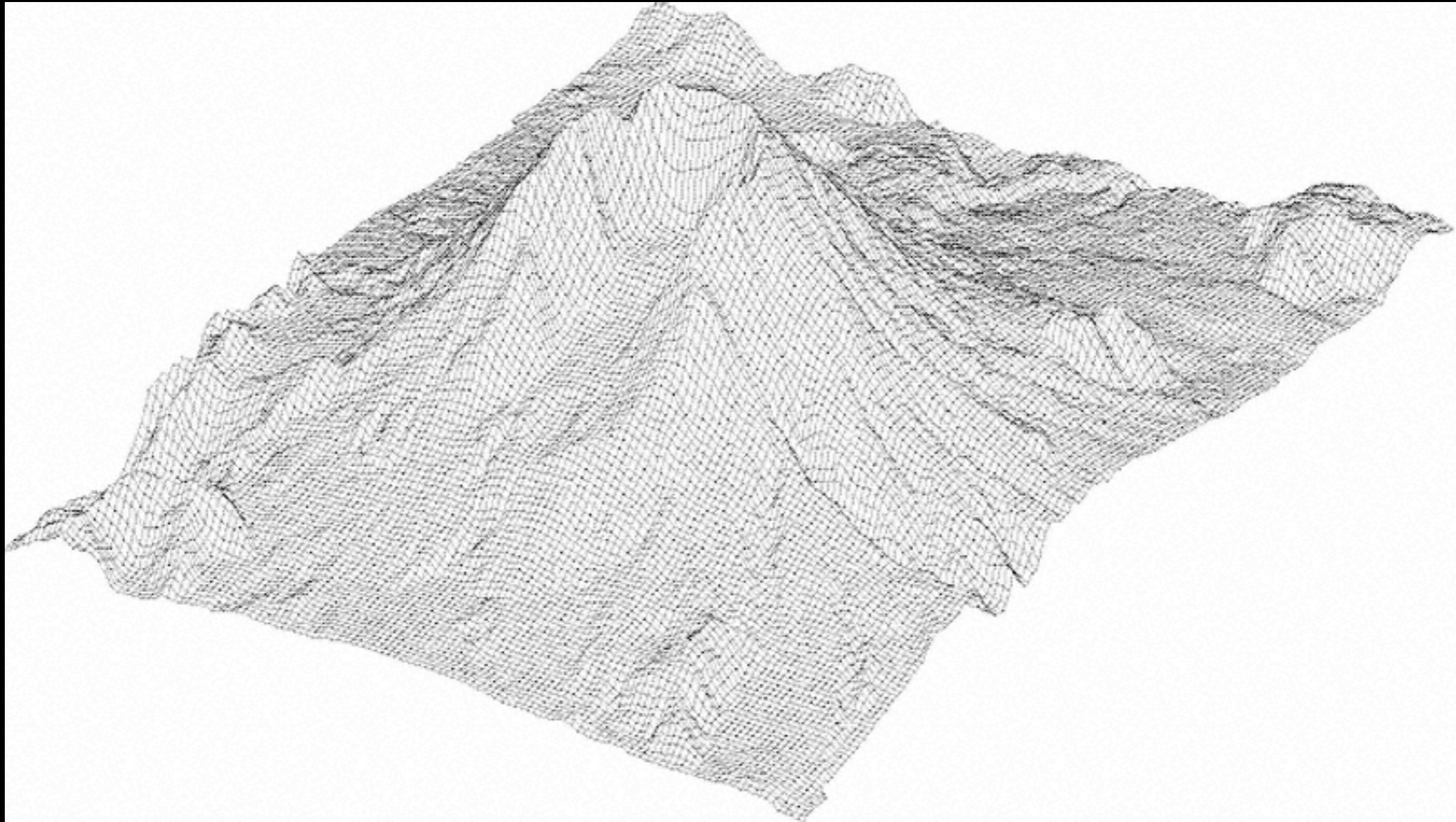
$$\frac{1}{\sqrt{n}} \text{ in any dimension}$$



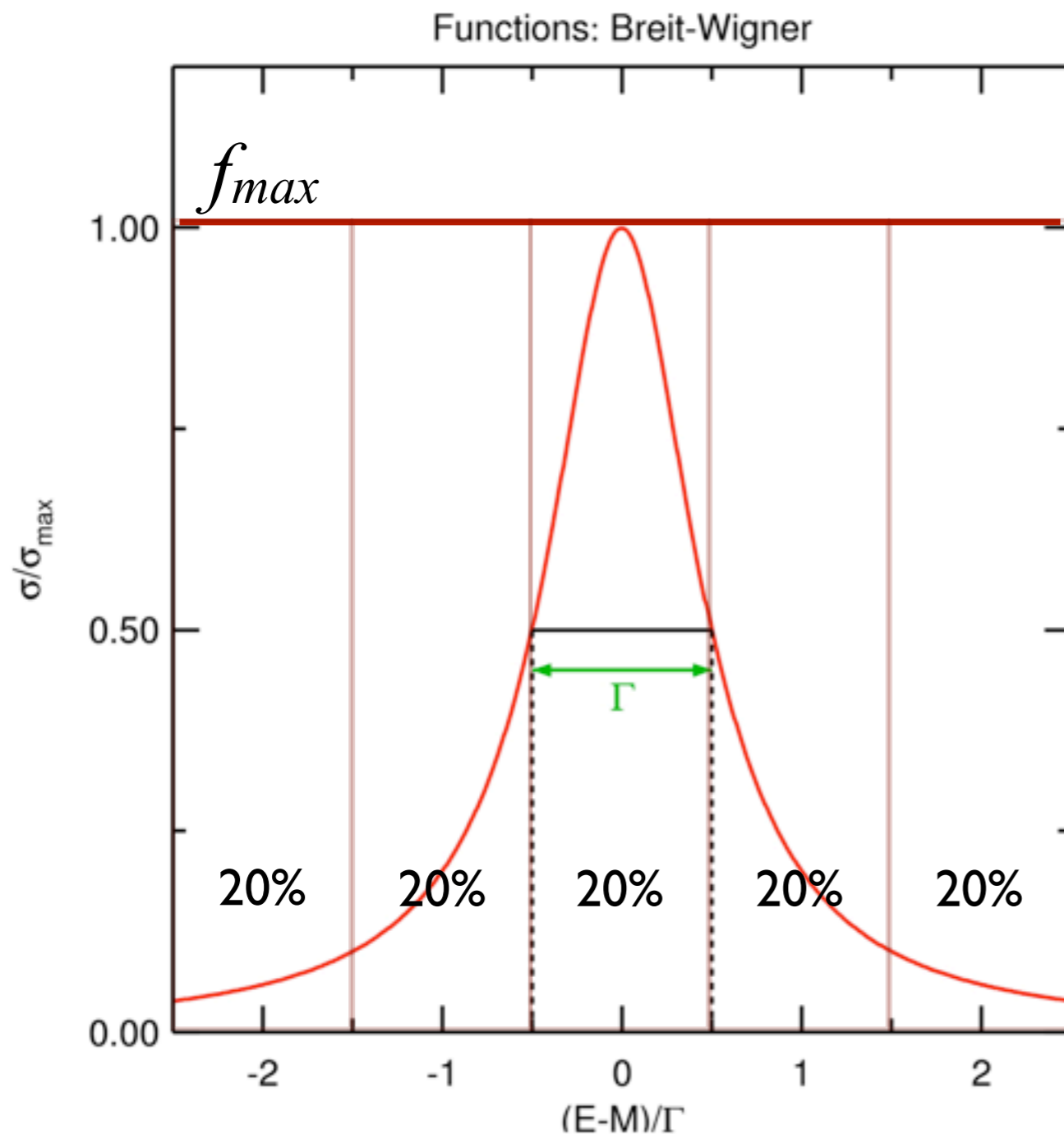
Uncertainty (after n evaluations)	$n_{\text{eval}} / \text{bin}$	Approx Conv. Rate (in 1D)	Approx Conv. Rate (in D dim)
Trapezoidal Rule (2-point)	$2^D$	$1/n^2$	$1/n^{2/D}$
Simpson's Rule (3-point)	$3^D$	$1/n^4$	$1/n^{4/D}$
... m-point (Gauss rule)	$m^D$	$1/n^{2m-1}$	$1/n^{(2m-1)/D}$
Monte Carlo	1	$1/n^{1/2}$	$1/n^{1/2}$

+ can re-use previously generated points ( $\approx$  nesting)

# Importance Sampling



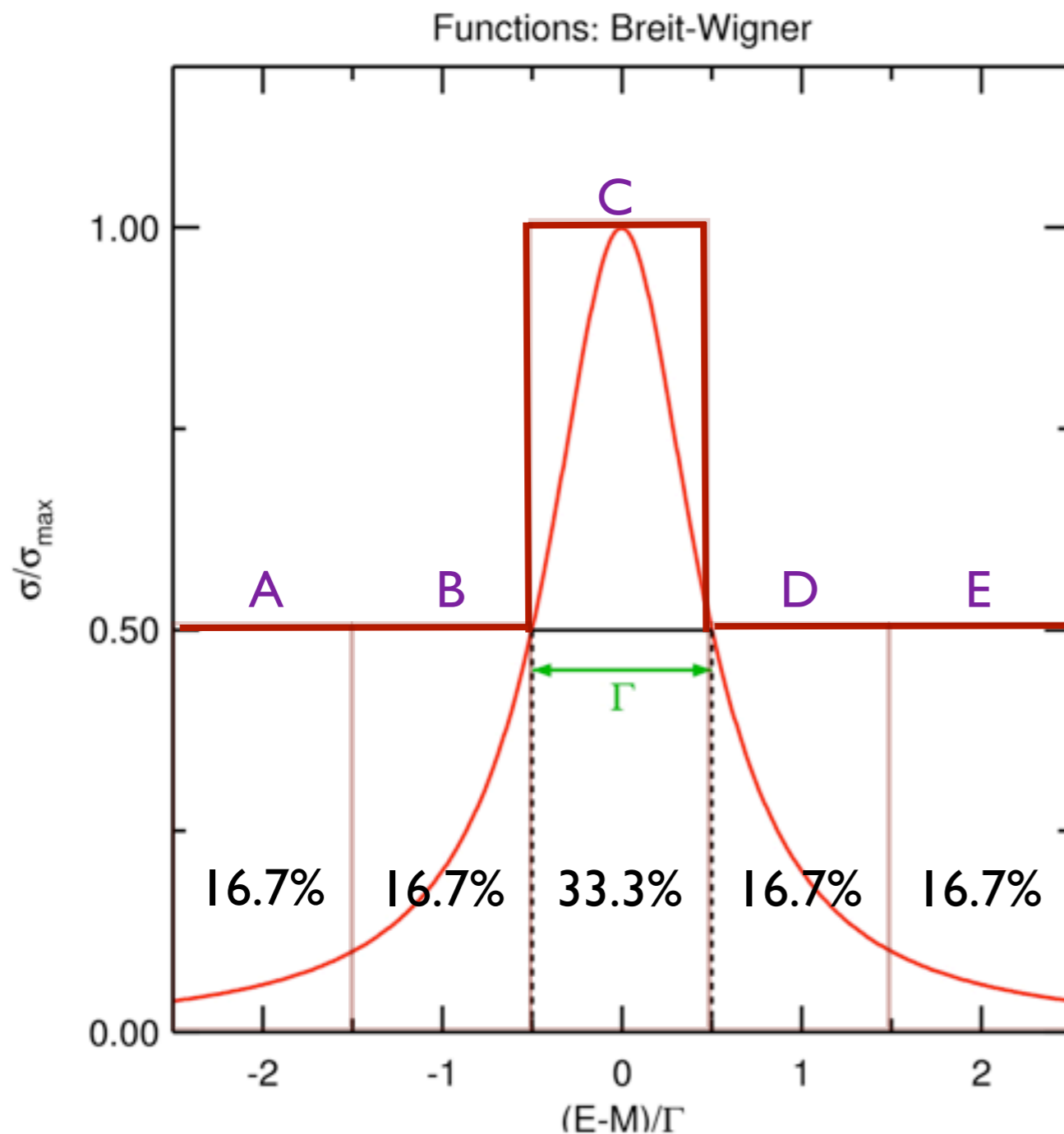
# Peaked Functions



Precision on integral dominated by the points with  $f \approx f_{\max}$  (i.e., peak regions)

→ slow convergence if high, narrow peaks

# Stratified Sampling




→ Make it twice as likely to throw points in the peak

Choose:

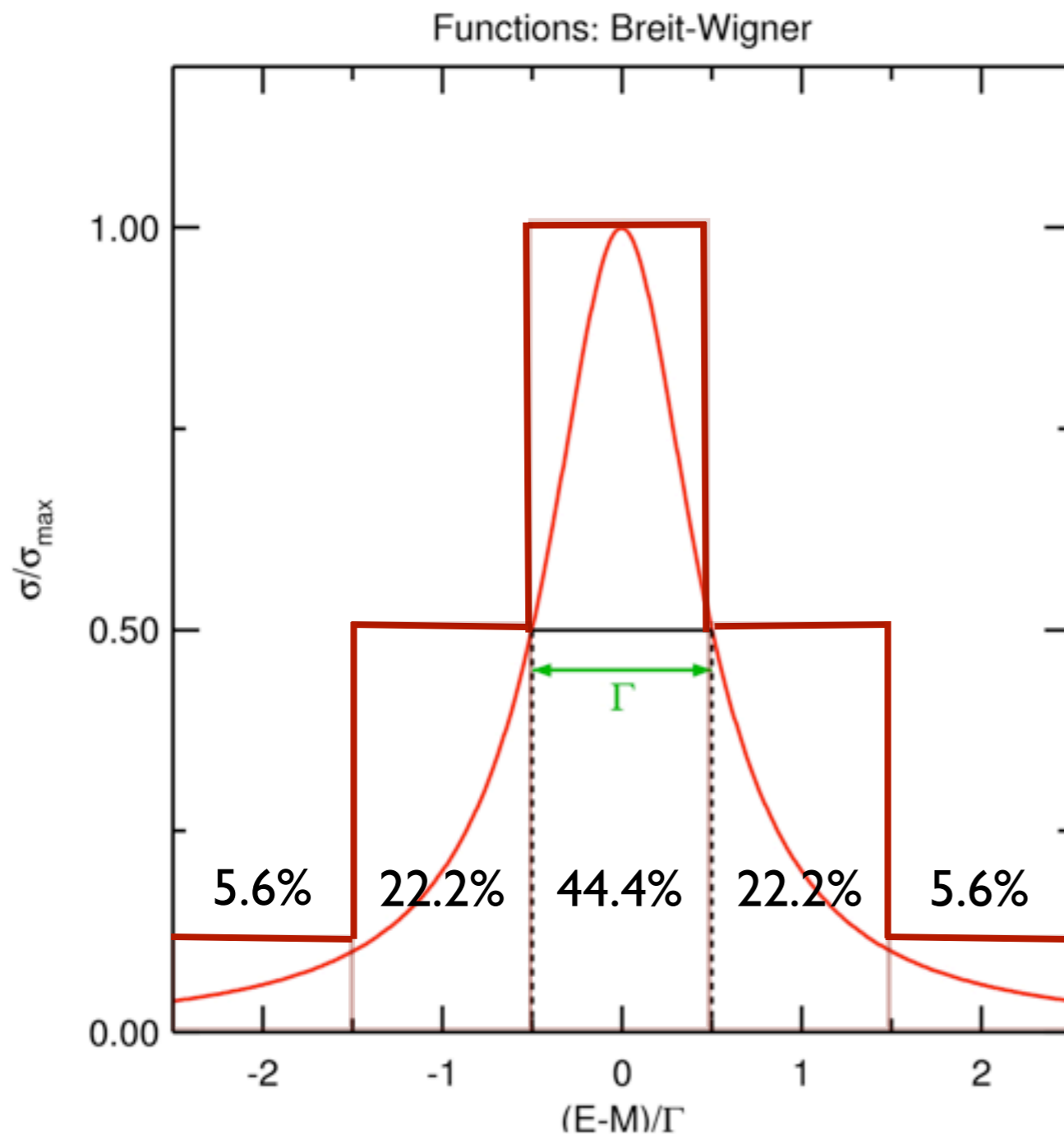
For:

$[0, 1]$	→ Region A
$[1, 2]$	→ Region B
$6 * R_1 \in [2, 4]$	→ Region C
$[4, 5]$	→ Region D
$[5, 6]$	→ Region E



→ faster convergence for same number of function evaluations

# Adaptive Sampling

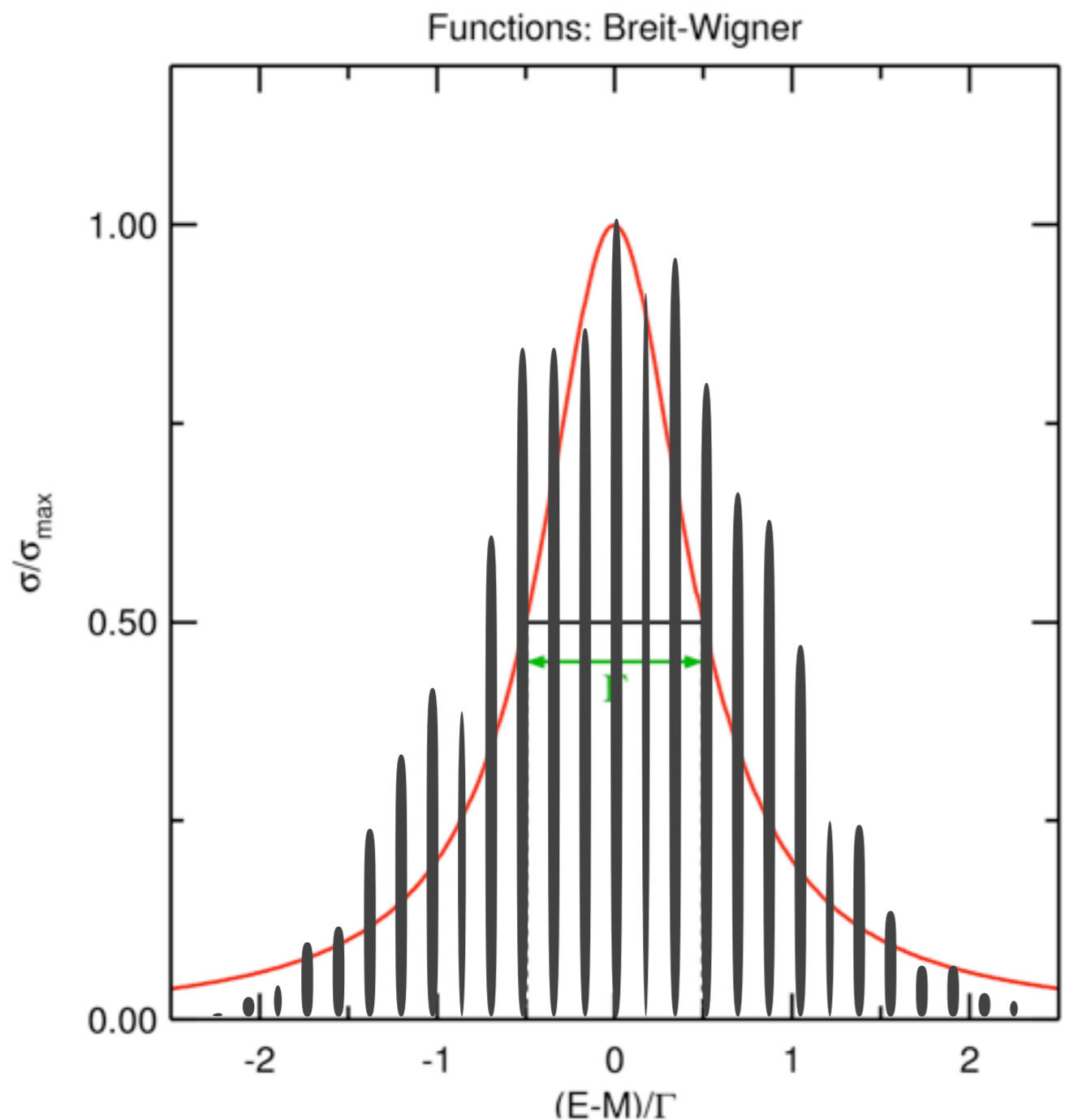


→ Can even design algorithms that do this automatically as they run (not covered here)

→ Adaptive sampling



# Importance Sampling



→ or throw points according to some smooth peaked function for which you have, or can construct, a random number generator (here: Gauss)

E.g., VEGAS algorithm, by G. Lepage

# Why does this work?

**1) You are inputting knowledge: obviously need to know where the peaks are to begin with ...**

**(say you know, e.g., the location and width of a resonance)**

**2) Stratified sampling increases efficiency by combining n-point quadrature with the MC method, with further gains from adaptation**

**3) Importance sampling:**

$$\int_a^b f(x) dx = \int_a^b \frac{f(x)}{g(x)} dG(x)$$

Effectively does flat MC with changed integration variables

Fast convergence if  $f(x)/g(x) \approx 1$

# The Veto Algorithm

Hit



Miss

# How we do Monte Carlo

## Take your system

Set of radioactive nuclei

Set of hard scattering processes

Set of resonances that are going to decay

Set of particles coming into your detector

Set of cosmic photons traveling across the galaxy

Set of molecules

...



# How we do Monte Carlo

## Take your system

## Generate a “trial” (event/decay/interaction/... )

Not easy to generate random numbers distributed according to exactly the right distribution?

May have complicated dynamics, interactions ...

→ use a simpler “trial” distribution

Flat with some stratification

Or importance sample with simple overestimating function (for which you can generate random #s)

# How we do Monte Carlo

**Take your system**

**Generate a “trial”** (event/decay/interaction/... )

**Accept trial with probability  $f(x)/g(x)$**

*$f(x)$  contains all the complicated dynamics*

*$g(x)$  is the simple trial function*

**If accept: replace with new system state**

**If reject: keep previous system state**

no dependence on  $g$  in final result -  
only affects convergence rate

**And keep going: generate next trial ...**



# How we do Monte Carlo

**Take your system**

**Generate a “trial”** (event/decay)

Accept trial with probability  $f(x)/g(x)$

*f(x) contains all the complicated dynamics*

*g(x) is the simple trial function*

If accept: replace with new system state

If reject: keep previous system state

no dependence on g in final result -  
only affects convergence rate

**And keep going: generate next trial ...**

Sounds deceptively simple, but ...

**with it, you can integrate**

arbitrarily complicated functions (in particular chains of nested functions), over arbitrarily complicated regions, in arbitrarily many dimensions ...



**Example:** Number of students who will get hit by a car during the next 3 weeks

## **Complicated Function:**

### ***Time-dependent***

Traffic density during day, week-days vs week-ends

(simulates non-trivial time evolution of system)

### ***No two students are the same***

Need to compute probability for each and sum

(simulates having several distinct types of “evolvers”)

### ***Multiple outcomes:***

Hit → keep walking, or go to hospital?

Multiple hits = Product of single hits, or more complicated?



# Monte Carlo Approach

## Approximate Traffic

### Simple overestimate:

*highest recorded density  
of most careless drivers,  
driving at highest recorded speed*

*etc. (If this becomes too slow (computing time), try more clever  
“stratifications”, adaptations, and/or importance sampling)*



## Approximate Student

**by most accident-prone** Left- and Right-hand traffic  
student (overestimate)

# Hit Generator

Off we go...

Throw random accidents according to:

$$R = \int_{t_0}^{t_e} dt \int dx \sum_{i=1}^{n_{\text{stud}}} \alpha_i(x, t) \rho_i(x, t) \rho_c(x, t)$$

Student-Car Coupling
Density of Student i
Density of Cars

Sum over students
(possibly weighted by speed × drunkenness)

$t_e$  : time of accident

Stratification

$$R = (t_e - t_0) \Delta x \left( \alpha_{L, \max} N_L + \alpha_{R, \max} N_R \right) \rho_{c, \max}$$

Coupling of most accident-prone left-hand-traffic student
Coupling of most accident-prone right-hand-traffic student
Rush-hour density of cars

Too Difficult

Simple Overestimate

# Hit Generator

## Trial Generator: (generate $t_e$ )

$$R = (t_e - t_0) \Delta x \left( \alpha_{L,\max} N_L + \alpha_{R,\max} N_R \right) \rho_{c\max}$$

$t_e$  : time of accident

Coupling of most accident-prone left-hand-traffic student

Coupling of most accident-prone right-hand-traffic student

Rush-hour density of cars

Simple Overestimate

(Also generate trial  $x_e$ , uniformly in Kumasi)

## Accept with probability

$$P_{\text{accept}} = \frac{\alpha_i(x, t) \rho_i(x, t) \rho_c(x, t)}{(\alpha_{L,\max} N_L + \alpha_{R,\max} N_R) \rho_{c\max}}$$

→ True integral = number of accepted hits  
(note: we didn't really treat multiple hits ... → Markov Chain)

# Summary - Lecture 1

**Quantum Scattering Problems** are common to many areas of physics:  
To compute expectation value of observable: integrate over phase space

**Complicated functions** → Numerical Integration

**High Dimensions** → Monte Carlo (stochastic) convergence is fastest  
+ Additional power by stratification and/or importance sampling



**Additional Bonus** → Veto algorithm → direct simulation  
of arbitrarily complicated reaction chains → next lecture

# Recommended Reading

F. James

*Monte Carlo Theory and Practice*

**Rept.Prog.Phys.43 (1980) p.1145**

S. Weinzierl

Topical lectures given at the Research School Subatomic physics, Amsterdam, June 2000

*Introduction to Monte Carlo Methods*

**e-Print: hep-ph/0006269**

S. Teukolsky, B. Flannery, W. Press, T. Vetterling

*Numerical Recipes* (in FORTRAN, C, ...)

**<http://www.nr.com/>**

# LHC@home 2.0

Test4Theory - A Virtual Atom Smasher



<http://lhathome2.cern.ch/>

Over 400 billion simulated collision events

# Test4Theory

**10,000 Volunteers wanted a virtual atom smasher**  
(to help do high-energy theoretical-physics calculations)

**Problem: Lots of different machine architectures**

→ Use Virtualization (CernVM)

Provides standardized computing environment (in our case Scientific Linux) on *any* machine

Exact replica of our normal working environment → no worries

**Sending Jobs and Retrieving output**

Using BOINC platform for volunteer clouds

But can also use other distributed computing resources

See *Volunteer Clouds and citizen cyberscience for LHC physics*, by the LHC@home 2.0 team, C.Aguado Sanchez et al., CHEP 2010, J.Phys.Conf.Ser. 331 (2011) 062022.

# Last 24 Hours: 2853 machines



See [Volunteer Clouds and citizen cyberscience for LHC physics](#), by the LHC@home 2.0 team, C.Aguado Sanchez et al., CHEP 2010, J.Phys.Conf.Ser. 331 (2011) 062022.



