Introduction to Monte Carlo (MC) methods
Why Scientists like to gamble
Overview

• Integration by random numbers
  – Why?
  – How?

• Uncertainty, Sharply peaked distributions
  – Importance sampling

• Markov Processes and the Metropolis algorithm

• Examples
  – statistical physics
  – finance
  – weather forecasting
Integration – Area under a curve

Tile area with strips of height $f(x)$ and width $\delta x$

Analytical:
$\delta x \rightarrow dx \rightarrow 0$

Numerical: integral replaced with a sum.

Uncertainty depends on size of $\delta x$ (N points) and order of scheme, (Trapezoidal, Simpson, etc)
Multi-dimensional integration

1d integration requires $N$ points

2d integration requires $N^2$

Problem of dimension $m$ requires $N^m$

*Curse of dimensionality*
Calculating $\pi$ by MC

Area of circle = $\pi r^2$
Area of unit square, $s = 1$
Area of shaded arc, $c = \pi/4$
c/s = $\pi/4$

Estimate ratio of shaded to non-shaded area to determine $\pi$
The algorithm

• \( y = \text{rand}() / \text{RAND\_MAX} \) // float \{0.0:1.0\}
• \( x = \text{rand}() / \text{RAND\_MAX} \)
• \( P = x^2 + y^2 \) // \( x^2 + y^2 = 1 \) eqn of circle
• If \( P \leq 1 \)
  – isInCircle
• Else
  – IsOutCircle
• \( \pi = 4 \times \text{isInCircle} / (\text{isOutCircle} + \text{isInCircle}) \)
π from 10 darts

π = 2.8
π from 100 darts

π = 3.0
\[ \pi = 3.12 \]
Estimating the uncertainty

- **Stochastic method**
  - Statistical uncertainty

- **Estimate this**
  - Run each measurement 100 times with different random number sequences
  - Determine the variance of the distribution

\[ \sigma^2 = \left( \bar{x} - x \right)^2 / k \]

- **Standard deviation is** \( \sigma \)
- **How does the uncertainty scale with** \( N \), **number of samples**
Uncertainty versus $N$

- Log-log plot
  \[ y = ax^b \]
  \[ \log y = \log a + b \log x \]
- Exponent $b$, is gradient
- $b \approx -0.5$
- Law of large numbers and central limit theorem

\[ \Delta \sim 1/\sqrt{N} \]

True for all MC methods
More realistic problem

• Imagine traffic model
  – can compute average velocity for a given density
  – this in itself requires random numbers ...

• What if we wanted to know average velocity of cars over a week
  – each day has a different density of cars (weekday, weekend, ...)
  – assume this has been measured (by a man with a clipboard)

<table>
<thead>
<tr>
<th>Density</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>4</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>0.7</td>
<td>2</td>
</tr>
</tbody>
</table>
Expectation values

• Procedure:
  – run a simulation for each density to give average car velocity
  – compute average over week by weighting by probability of that density
    
  – i.e. velocity = 1/7 * (4 * velocity(density = 0.3) + 1 * velocity(density = 0.5) + 2 * velocity(density = 0.7))

• In general, for many states $x_i$ (e.g. density) and some function $f(x_i)$ (e.g. velocity) need to compute expectation value $<f>$

\[
\sum_{1}^{N} p(x_i) * f(x_i)
\]
Continuous distribution

probability of occurrence

density of traffic
Aside: A highly dimensional system
A high dimensional system

- 1 coin has 1 degree of freedom
  - Two possible states Heads and Tails
- 2 coins have 2 degrees of freedoms
  - Four possible micro-states, two of which are the same
  - Three possible states 1*HH, 2*HT, 1*TT
- n coins have n degrees of freedom
  - \(2^n\) microstates: n+1 states
  - Number of micro-states in each state is given by the binomial expansion coefficient

\[
\Omega = 2^n = \sum_{r=0}^{n} r \binom{n}{r} H^r T^{n-r}
\]

\[
r \binom{n}{r} = \frac{n!}{r!(n-r)!}
\]
Highly peaked distribution
Highly peaked distribution

![Probability distribution graph](image)

- Fraction of max number of heads
- Fraction of heads
100 Coins

- 96.48% of all possible outcomes lie between 40 – 60 heads
Importance Sampling (i)

- The **distribution** is often sharply peaked
  - especially high-dimensional functions
  - often with fine structure detail
- Random sampling
  - $p(x_i) \sim 0$ for many $x_i$
  - $N$ large to resolve fine structure
- Importance sampling
  - generate **weighted distribution**
  - proportional to probability
Importance Sampling (ii)

- With random (or uniform) sampling

\[ \langle f \rangle = \sum_{1}^{N} p(x_i) \cdot f(x_i) \]

  - but for highly peaked distributions, \( p(x_i) \sim 0 \) for most cases
  - most of our measurements of \( f(x_i) \) are effectively wasted
  - large statistical uncertainty in result

- If we generate \( x_i \) with *probability proportional* to \( p(x_i) \)

\[ \langle f \rangle = \frac{1}{N} \sum_{1}^{N} f(x_i) \]

  - all measurements contribute equally

- But how do we do this?
Hill-walking example

• Want to spend your time in areas proportional to height $h(x)$

  – walk randomly to explore all positions $x_i$
  – if you always head up-hill or down-hill
    – get stuck at nearest peak or valley
  – if you head up-hill or down-hill with equal probability
    – you don’t prefer peaks over valleys

• Strategy
  – take both up-hill and down-hill steps but with a preference for up-hill
• Generate samples of \( \{x_i\} \) with probability \( p(x) \)
• \( x_i \) no longer chosen independently
• Generate new value from old – evolution
  \[
x_{i+1} = x_i + \delta x
\]

• Accept/reject change based on \( p(x_i) \) and \( p(x_{i+1}) \)
  – if \( p(x_{i+1}) > p(x_i) \) then accept the change
  – if \( p(x_{i+1}) < p(x_i) \) then accept with probability \( \frac{p(x_{i+1})}{p(x_i)} \)

• Asymptotic probability of \( x_i \) appearing is proportional to \( p(x) \)
• Need random numbers
  – to generate random moves \( \delta x \) and to do accept/reject step
Markov Chains

• The generated sample forms a Markov chain

• The update process must be ergodic
  – Able to reach all \( x \)
  – If the updates are non-ergodic then some states will be absent
  – Probability distribution will not be sampled correctly
  – computed expectation values will be incorrect!

• Takes some time to equilibrate
  – need to forget where you started from

• Accept / reject step is called the Metropolis algorithm
Markov Chains and Convergence

\[ \langle f \rangle = \frac{1}{10} \sum_{i=4}^{13} f(x_i) \]
Statistical Physics

• Many applications use MC
• Statistical physics is an example
• Systems have extremely high dimensionality
  – e.g. positions and orientations of millions of atoms
• Use MC to generate “snapshots” or configurations of the system
• Average over these to obtain answer
  – Each individual state has no real meaning on its own
  – Quantities determined as averages across all the states
MC in Finance

• Used to price *options*

• An option is a *contract*, holder has the *right*
  – buy an asset – *call*
  – sell an asset – *put*
  – at some time in the future (T)
  – For a predetermined price (*strike* price) \( X \)

• Terminal pay off for the holder is then

\[
\max(\pm(S_T - X), 0)
\]

  – where \( S_T \) is the price of the underlying asset at time \( T \)
  – \( \pm \) call/put

• How much should the option cost?
**MC in Finance II**

- Price model called Black-Scholes equation
  - Partial differential equation
  - Based on geometric brownian motion (GMB) of underlying asset

- Assumes a “perfect” market
  - Markets are not perfect, especially during crashes!
  - Many extensions
  - Area of active research

- Use MC to generate many different GMB paths
  - Statistically analyse ensemble
Image taken by NASA’s Terra Satellite
7th January 2010

Britain in the grip of a very cold spell of weather
NWP in the UK

• Weather forecasts used by the media in the UK (e.g. BBC news) are generated by the UK Met office
  – Code is called the Unified Model
  – Same code runs climate model and weather forecast
  – Can cover the whole globe

• Newest supercomputer
  – Cray XC40
  – almost half a million processor-cores
  – weighs 140 tonnes

Initial conditions and the Butterfly effect

- The equations are extremely sensitive to initial conditions
  - Small changes in the initial conditions result in large changes in outcome

- Discovered by Edward Lorenz *circa* 1960
  - 12 variable computer model
  - Minute variations in input parameters
  - Resulted in grossly different weather patterns

- The Butterfly effect
  - The flap of a butterfly’s wings can effect the path of a tornado
  - My prediction is wrong because of effects too small to see
Chaos, randomness and probability

• A Chaotic system evolves to very different states from close initial states
  – no discernible pattern

• We can use this to estimate how reliable our forecast is:
  • Perturb the initial conditions
    – Based on uncertainty of measurement
    – Run a new forecast
  • Repeat many times (random numbers to do perturbation)
    – Generate an “ensemble” of forecasts
    – Can then estimate the probability of the forecast being correct

• If we ran 100 simulations and 70 said it would rain
  – probability of rain is 70%
  – called ensemble weather forecasting
Optimisation Problems

• Optima of function rather than averages
• Often need to minimise or maximise functions of many variables
  – minimum distance for travelling salesman problem
  – minimum error for a set of linear equations
• Procedure
  – take an initial guess
  – successively update to progress towards solution
• What changes should be proposed?
  – could reduce/increase the function with each update (steepest descent/ascent) ...
  – ... but this will only find the local minimum/maximum
Stochastic Optimisation

• Add a random component to updates
• Sometimes make "bad" moves
  – possible to escape from local minima
  – but want more up-hill steps than down-hill ones
• Hill-walking example
  – find the highest peak in the Alps by maximising $h(x)$
Simulated Annealing

- Monte Carlo technique applied to optimisation
- Analogy with Metropolis and Statistical Mechanics
- Initial “high-temperature” phase
  - accept both up-hill and down-hill steps to explore the space
- Intermediate phase
  - start to prefer up-hill steps to look for highest mountain
- Final “zero-temperature” phase
  - only accept up-hill steps to locate the peak of the mountain
- A lot of freedom in how you vary the temperature ...
Summary

• Random numbers used in many simulations

• Mainly to efficiently sample a large space of possibilities

• One state generated from another: Markov Chain
  – Metropolis algorithm gives a guided random walk

• Real simulations can require trillions of random numbers!
  – parallelisation introduces additional complexities ...