Monte Carlo programs

Alexander Khanov

PHYS6260: Experimental Methods is HEP
Oklahoma State University

October 18, 2023

Monte Carlo methods: the idea

- In classical physics, if we need to measure something we just do it
- In quantum physics, it's not so: a single measurement doesn't make sense, all measurements are statistical
  - we perform many measurements and obtain probability distribution of outputs
- Each experiment consists of a long series of "mini-experiments" (called "events" in HEP jargon), each of them may look differently
  - e.g. we measure the mass of a Z boson produced in $p\bar{p}$ collisions, sometimes it's produced alone, sometimes it's accompanied by other particles; it may decay to electrons or muons, etc.
- This paradigm propagates back to theory: instead of deriving formulas for predicted particle properties, we can mimic the experiment and simulate a series of mini-experiments using random number generators

History

- MC method was invented by Stanislaw Ulam in the late 40's during his work on Manhattan Project
- It was named MC by Nicholas Metropolis after Monte Carlo Casino where Ulam's uncle loved to gamble
- As soon as it was invented, John von Neumann implemented it for ENIAC
  - Electronic Numerical Integrator and Computer: the world's first general purpose electronic computer (1946)

Applications of MC methods

- Cross section calculations
- Event generation
- Detector simulation

Calculation of cross sections

- "Cross section" in HEP = probability that particles collide and react in a certain way
  - related to the (classical) cross section for historical reasons – early experiments (Rutherfords) tried to measure the size of elementary particles
  - it's better than reaction rate because it's independent of beam size / intensity, so results from various detectors can be easily compared
- Cross section can be calculated as

$$\sigma = \int |M|^2 \, d\Phi$$

  - This is an integral over $3n - 4$ dimensional surface (called "phase space") in $4n$ dimensional space
  - $d\Phi$ = phase space element, $|M|^2$ = matrix element

Phase space and matrix element

- Phase space element = solid angle + 4-momentum conservation

$$d\Phi = (2\pi)^4 \delta^{(4)} \left(P - \sum_{i=1}^{n} p_i \right) \prod_{i=1}^{n} \frac{d^3 p_i}{(2\pi)^3 2E_i}$$

- If $|M|^2 = 1$ (or const) then things are simple:
  - $d\Phi$ is simple if n is small
  - $\int d\Phi$ is easy to calculate if particles are massless
- Unfortunately the most interesting cases are when $|M|^2 \neq \text{const}$

Cross section calculation

- Use phase space generator to produce 4-momenta of products uniformly distributed in phase space
- For each random point in phase space, calculate "weight" $w_k = |M|^2$
- Cross section:

$$\sigma = \frac{1}{N} \sum_{k=1}^{N} w_k$$

Event generation

- In the above example, each "event" (a combination of 4-momenta of products for which the matrix element is calculated) consists of a fixed set of product particles
- In practice various combinations of product particles are created in the same process
  - it's usually convenient to consider a family of processes (think Feynman diagrams) instead of a single process – one needs to take into account the interference
  - once created, most particles undergo further interactions: hadronization, decays
- "Event generators" = programs which combine details of production (e.g. parton density functions), matrix element calculations and post-production effects
- Examples of event generators: PYTHIA, HERWIG, POWHEG, ISAJET

Event generation interface

- Parts of event generators are interchangeable – e.g. matrix element generation by POWHEG can be followed by hadronization by PYTHIA
- The standard interface between the MC programs: the Les Houches Accord
  `https://arxiv.org/pdf/hep-ph/0109068.pdf`

> Modularization of High Energy Particle Physics event generation is becoming increasingly useful as the complexity of Monte Carlo programs grows. To accommodate this trend, several authors of popular Monte Carlo and matrix element programs attending the *Physics at TeV Colliders Workshop* in Les Houches, 2001 have agreed on a generic format for the transfer of parton level event configurations from matrix element event generators (MEG) to showering and hadronization event generators (SHG).

CompHEP [1]		
Grace [2]		Herwig [6]
MadGraph [3]	$\Rightarrow$	Isajet [7]
VecBos [4]		Pythia [8]
WbbGen [5]		...
...		

Detector response

- Event generators give us 4-vectors of product particles, but that's not what we see in our detectors
- The particles created in reactions go through the detector material and undergo interactions
- We need to simulate passage of particle through matter and detector response
- The result of the simulation if "how the detector would see it"

Particle tracking: the algorithm

- For each particle in the event, take its 4-momentum and calculate where the particle will be after short step $d\mathbf{x}$ (watch for magnetic field!)
- Calculate probabilities of everything that could happen to the particle
- Throw a random number and decide if we want it to happen
  - the particle may "stop" (disappear) – remove it from the list of active particles
  - the particle may decay – remove it from the list and add new particles – the decay products
  - the particle may produce additional particle(s), e.g. delta-rays – add them to the list of particles and fix the particle 4-momentum
- If nothing bad is going to happen to the particle, then
  - use the Bethe formula to evaluate mean dE/dx, generate random (Landau distributed) energy loss and reduce the particle energy
  - generate the random angle of multiple scattering and correct the particle direction
- Repeat!

Fast vs full simulation

- The simulation program has a cut-off energy below which the particles are eliminated from the list
  - if the cut-off is too low then the number of particles due to secondary interactions explodes anyway
- For more realistic calculations, need to set the cut-off reasonably low – the calculations become very slow
  - if details are not important, it's OK to keep the cut-off high, and just record the disappearance of particles without producing secondaries ("fast simulation")
- Ultimately, one can only consider the particles coming from the event generator and use parameterizations to estimate the detector response