# Monte Carlo Simulations in Subatomic Physics

Wouter Deconinck

## Monte Carlo Method: Evaluating Complicated Integrals

Typical quantities of interest in subatomic physics:

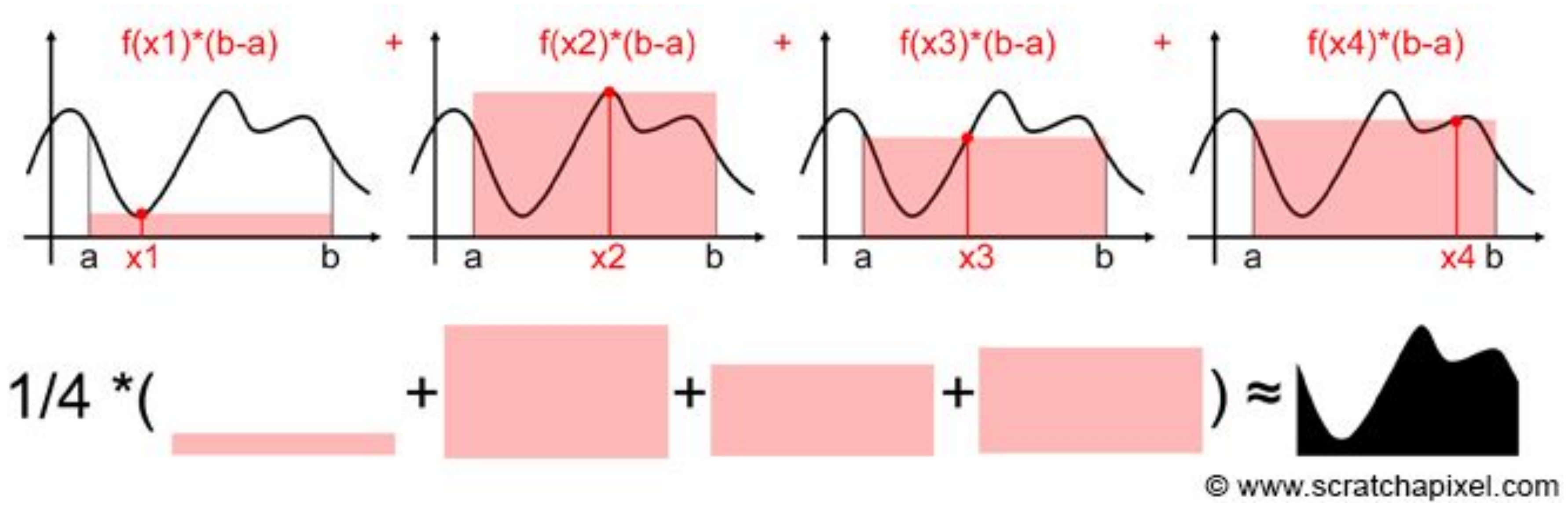
- average rate of particles reaching a detector element
- total helicity-correlated asymmetry for a detector element
- average energy deposition in a collimator or structural element

**Integrals over the full (*d*-dimensional) phase space of initial states (pid, $p_0$) of some integrand ($\mathcal{L}\sigma$, $A\sigma$, $E_{dep}\sigma$) $\times$ an acceptance function $\varepsilon$(pid, $p_0$).**

Monte Carlo integration approximates the integral as a sum over *N* points that are uniformly distributed over the full phase space (on compact domains), with an uncertainty that decreases as RMS/$\sqrt{N}$ (if there is convergence), even if *d* large.

We must 1) **determine $\varepsilon$(pid,$p_0$)**, and 2) **speed convergence** by reducing RMS.

## Monte Carlo Method: Evaluating Complicated Integrals



In the case of our simulations, *x* = *d*-dimensional space and either one of:

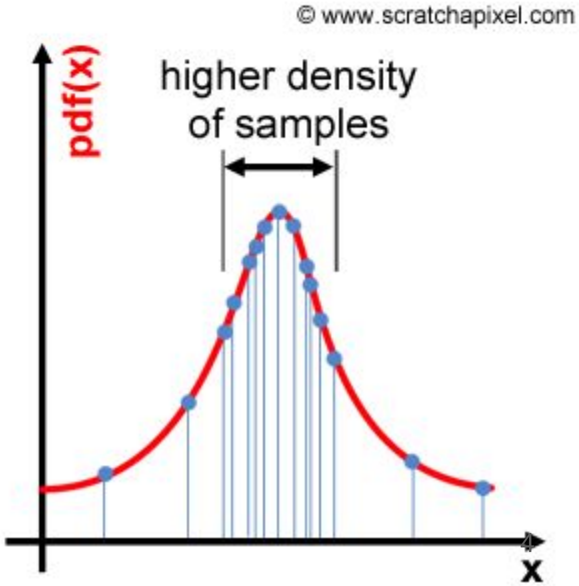
- rastered beam electrons with position-slope correlation upstream of target
- vertex *z*; (*E*,*x*,*y*,*θ*,*φ*) after multiple scattering/raster; (*E'*,*θ'*,*φ'*) after reaction

# Monte Carlo Method: Evaluating Complicated Integrals

Each sample gets a weight, $w = (b-a)/N = V/N$, V = phase space volume

**Importance sampling** and **variance reduction** (RMS):

- Not all parts of phase space contribute equally to the total integral: sample with non-uniform probability.
  - Weight will be 1/pdf(x)/N, and change from sample to sample
- Ideally we want to sample only the part of our phase space that is “important” to the integral:
  - High integrand function: $\mathcal{L}\sigma$, $A\sigma$, $E_{dep}\sigma$
    Very forward scattering, even if low acceptance
    Very high energy deposition, even if low acceptance
  - High acceptance function: $\varepsilon(\mathrm{pid}, p_0)$
    Zero or one bounce from target, even if low cross section



## Monte Carlo Method: Evaluating Complicated Integrals

Each sample gets a weight, $w = (b\text{-}a)/N = V/N$, V = phase space volume

Variable `rate` in output ROOT file can be used as event weight to obtain detected rates where bins can be interpreted as event rates for the simulated beam current.

**“Beam generator”**: rastered beam electrons upstream of target

- weight $w$ is constant for each event (each electron is weighted equally)

**“Physics generator”**: physics processes with calculation of σ and A

- weight $w$ is integrated luminosity $\mathcal{L} \times$ differential cross section $d\sigma/d\Omega \times$ phase space factor $\Delta\Omega$

# Monte Carlo Method: Evaluating Complicated Integrals

By design, $\varepsilon(\text{pid}, p_0)$ is large for Møller scattering which has a low overall cross section (compared to forward radiative processes in the target).

- **"Beam generator"** for upstream electrons: minimum bias, full phase space (limited by physics include in geant4 transport), but low efficiency for Møller
  Each sample gets a weight, *w*, in the `rate` variable:
  - normalized with 1/N for single simulation and $\int\mathcal{L}dt$ per second
- **"Physics generators"** for specific processes: biased, but high efficiency.
  Each sample gets a weight, *w*, in the `rate` variable:
  - based on the event generator sampling strategy,
  - multiplied with cross section $\sigma$ for the event generator process,
  - normalized with 1/N for single simulation and $\int\mathcal{L}dt$ per second

## Monte Carlo Method: Evaluating Complicated Integrals

Intermediate summary:

- Monte Carlo simulations are just numerical integrals over some domain
- Geant4 merely simulates the acceptance function part of the integrand
- The “beam generator” is the only “complete” simulation (to the extent of physics modeled in geant4, e.g. not including pion production)
- The “physics generators” are useful and fast approximations for part of the integrals of interest, including for some processes not included in geant4
    - Geant4 doesn’t know and doesn’t care about our physics generators. All it cares about is what initial state of particles the generator produces. From geant4’s point of view we have three generators: *ee*, *e*, *π*.
        - For initial state *e*, we could simply store the cross section for each of the processes that results in 1 electron in the initial state, and pick the desired one when analyzing.
        - For initial states *ee*, *π*, there is only one process generates them.

## Geant4: Determining Acceptance Function $\varepsilon(\mathrm{pid}, p_0)$

**This is the role of geant4: to determine what happens with the initial state.**

One initial state can result in multiple outcomes; which one is randomly decided:

- for **physical** reasons: particle decays happen with exponential probability.
- for **practical** reasons: materials are not modeled as individual atoms but as a set of interaction lengths at which processes happen. A positron might have:
  - $\ell_1$ = 20 mm: ionization of an atom in the material, creating an ion and an electron
  - $\ell_2$ = 50 mm: Bremsstrahlung, creating a gamma
  - $\ell_3$ = 80 mm: positron-electron annihilation, creating two gammas

  This would make ionization most likely, at this point. This depends on energy.

Each initial state, after having been processed by geant4, can only be considered as part of a statistical sum. Each outcome is one statistically possible outcome.

## Geant4: Determining Acceptance Function $\varepsilon(\mathrm{pid},p_0)$

$\varepsilon(\mathrm{pid},p_0)$ ≡ number of “hits” in a sensitive detector volume, an integer ≥ 0.

Here, “hits” could be:

- any charged particle in a specific quartz tile,
- electrons going through a virtual detector plane,
- photo-electrons generated a PMT photocathode,
- electromagnetic interactions in a magnet coil.

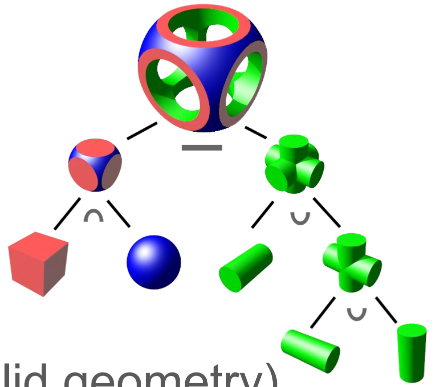
Two approaches are available:

- Store all “hits” and do the Monte Carlo sum with weight *w* in offline analysis
- Integrate “hits” for each event during simulation, e.g. $\langle E_{dep}\rangle = \sum E_{dep}\cdot\varepsilon(\mathrm{pid},p_0)$, and follow up with $\sum w\langle E_{dep}\rangle$ in offline analysis (this uses the `sum` branch)

## Physics Lists and Limits

- List of all allowable processes for each particle at given energy.
- Several predetermined options are available in geant4. Recommendations:
    - QGSP_BERT (default)
    - QGSP_BERT_HP (for better shielding simulations)
- Production cuts: minimum required interaction length for new particles
- User Limits: minimum energy, maximum length, maximum time
    - We may set maximum propagation length to zero for some volumes, and call it "kryptonite". All tracks will stop immediately in these volumes.

# Geant4: Definition of Geometry



There are three core geometry classes in geant4:

- **Solids** = a primitive geometric shape (CSG = constructive solid geometry), such as a box, tube,..., polycone,..., up to boolean combinations of other solids with relative position and rotation
- **Logical volumes** = a combination of a solid and a material (e.g. quartz box), local B-fields, sensitive detector, visibility attributes, GDML auxiliary attributes
- **Physical volumes** = placement of a 'daughter' logical volume (with relative position and rotation) inside a 'mother' logical volume; single or replication

One solid can be reused for multiple logical volumes.
One logical volume can be reused for multiple physical volumes.

This structure appears in any geometry definition, whether in C++ or in GDML.

# Geant4: Definition of Geometry

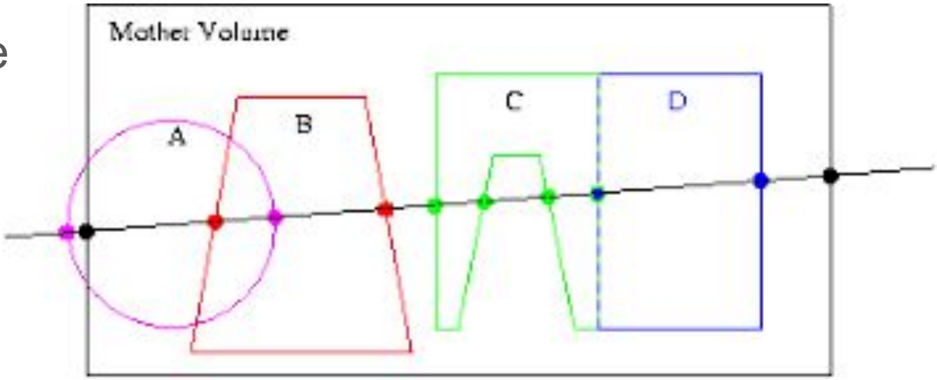
When inside a volume, at every step, geant4 evaluates the following:

- Check if still inside the volume itself (i.e. distance to outside)
  - depends on how complicated the volume is
- Check if entering any new daughters (i.e. distance to inside)
  - depends on how complicated, how many daughter volumes, bounding box calculation
- When exiting the volume, enter mother volume and any of its daughters

Geometry do’s and don’ts:

- Reuse solids and logical volumes if possible
- Encapsulate complicated volumes in simple primitives
- Balance the geometry tree: avoid large numbers of daughters
  - Exception: large parameterized or replicated volume numbers are encouraged

## Geant4: Definition of Geometry



Overlapping volumes: A, B, mother volume

- Tracks in one direction look different from tracks in the opposite direction.

Check for overlaps (done automatically)

- Pick random straight lines from center point and checks intersections
- Can miss overlaps, distances will not be consistent

Shared faces are allowed

- There is NO need to add arbitrary small distances to avoid shared faces, such as between C and D. Geant4 handles this at a few times numerical precision.

## Randomness: Don't Leave It To Chance

- Only when events are statistically independent (IID) does the Monte Carlo method work, but: computers are perfectly deterministic.
- (Pseudo-)Random number generator has an internal state (many numbers), which can be set with a seed (one number).
  - Many more states than seeds. Not every state has a corresponding seed.
- Every event’s state is saved in output ROOT tree so you can reproduce that exact event (on the exact same compiled remoll version).
- Remoll sets a (pseudo-)random seed with (pseudo-)randomness from the system, unless you specify the seed explicitly or load a full state explicitly.

## Geant4: Definition of a Hit

Geant4 hit ≡ any endpoint of a step in a logical volume that is a sensitive detector.

This is different from what we often think of as a “hit”:

- One track can leave multiple “geant4 hits” in a sensitive detector, even if we would digitize them into a single signal and think of them as a single “hit”.
    - E.g. using photo-electrons in a PMT to determine rate is going to overestimate actual rate by the average number of photo-electrons per event.
- Multiple tracks can each leave “geant4 hits” in a sensitive detector, and we would still digitize them into a single signal and a single “hit”.
    - E.g. a showermax calorimeter would likely have many hits from many tracks, but ultimately you would just add them up in some way and interpret the single pulse as a hit.

This difference between a “geant4 hit” and a “hit” is not handled by remoll, beyond the `sum` branch which can be used for summing hits.

## Geant4 System of Units: 1 = mm, ns, radian, MeV

Geant4 uses a self-consistent system of units with a corresponding set of physical constants. Physical quantities in geant4 and remoll are implicitly in these units. Other units are of course also available, e.g. `GeV == 1000`.

A subset of common units is stored in the output ROOT file.

- Input quantities with their units: `0.5*cm, 11*GeV`
- Output quantities by dividing units: `hit.x/mm, hit.p/GeV`

This will make units very apparent in automatic plot labels.

Geant4 has a full set of physical constants in this unit system, e.g. `hbarc` = $\hbar c$

## Optical Photons

- Many experiments create optical photons, and don't just register charged particles in volumes. Geant4 can simulate scintillation, Cherenkov light.
- One charged particle can generate many 100s or 1000s of optical photons, so this is not enabled by default.
    - Must be explicitly enabled at a global level
    - Must be enabled at a volume level (to allow propagation)
- Surface properties become important: diffuse reflection due to polish, refraction when changing media,...
    - Standalone simulations of individual detectors are possible inside remoll, and make it much easier to integrate the component in the full model later on.

# Remoll Data Structure

The output ROOT file from remoll includes the geant4 hits in one list per event.

- `units:` branch with common units
- `seed:` branch with random state information
- `rate:` variable for rate-weighted histograms
- `ev:` event-level quantities: asymmetry, beam energy, cross section
- `bm:` beam-level quantities: raster position, angles
- `part:` generated particles in initial state (two per event for Møller generator)
- `hit:` hit-level quantities: time, position, energy, pid, etc
- `sum:` summed hit-level quantities: $E_{dep}$, $\langle x \rangle$,$\langle y \rangle$,$\langle z \rangle$