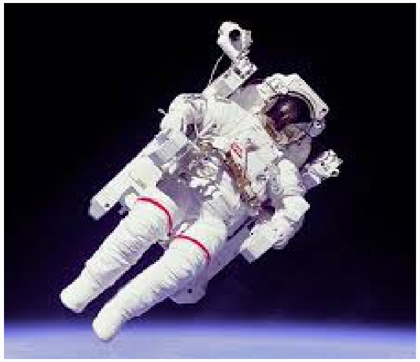
Introduction to the Monte Carlo method and Geant4

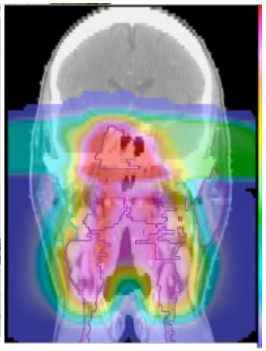
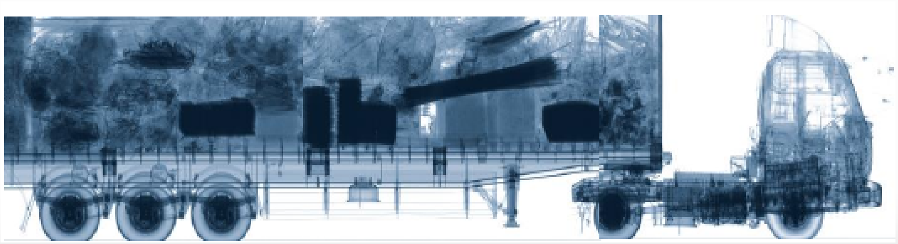
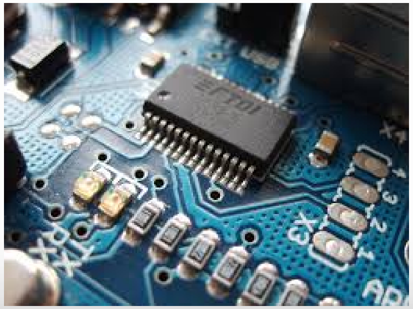
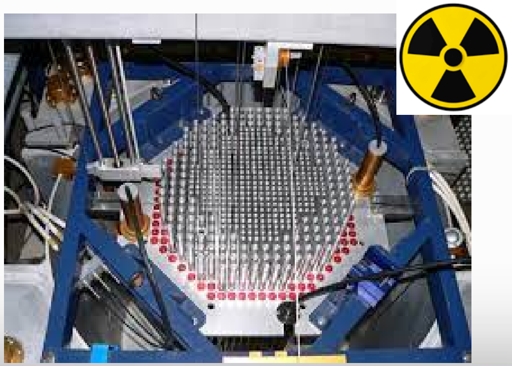
Condensed from presentations by Marc Verderi, Makoto Asai and Dennis Wright

GEANT4 applications

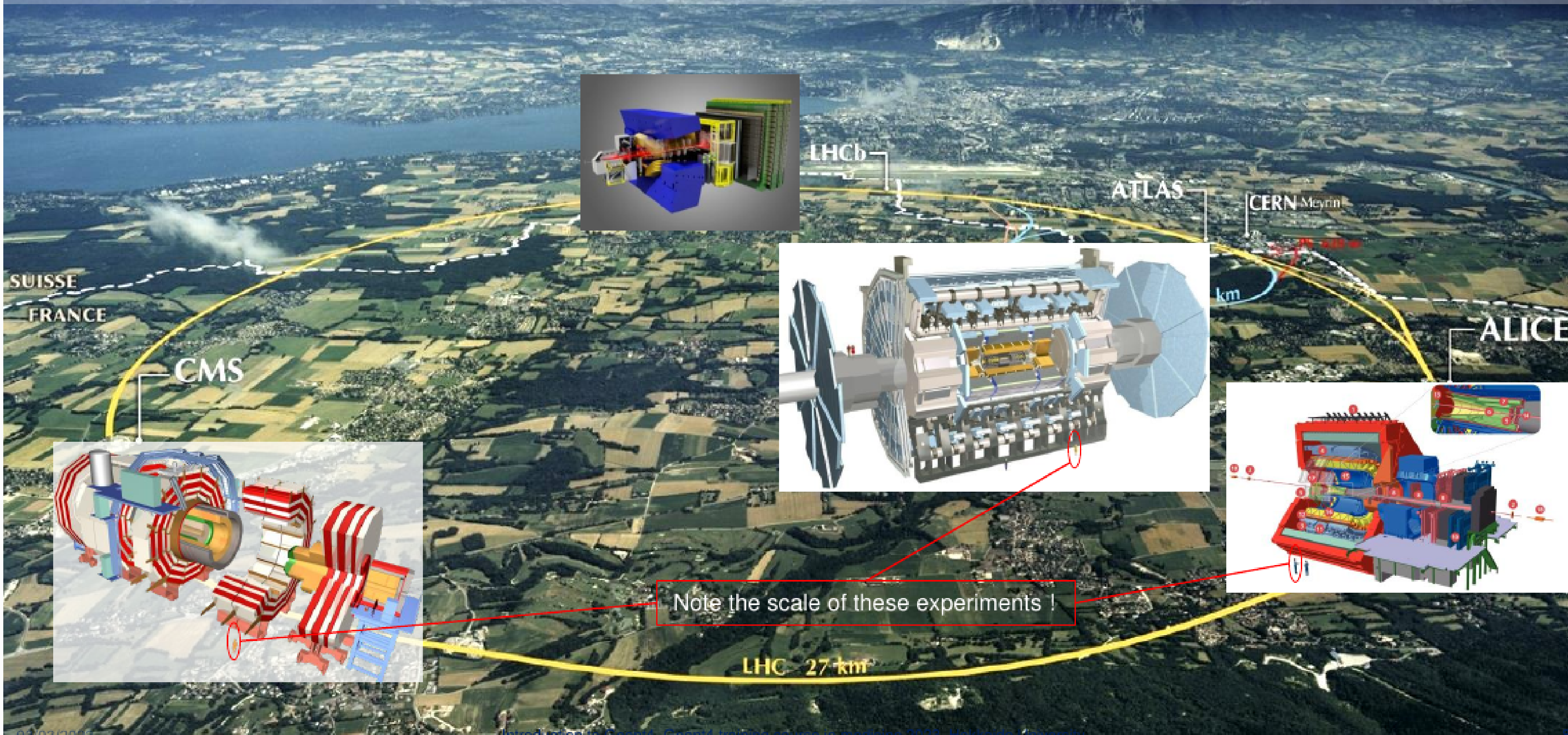
A SIMULATION TOOLKIT



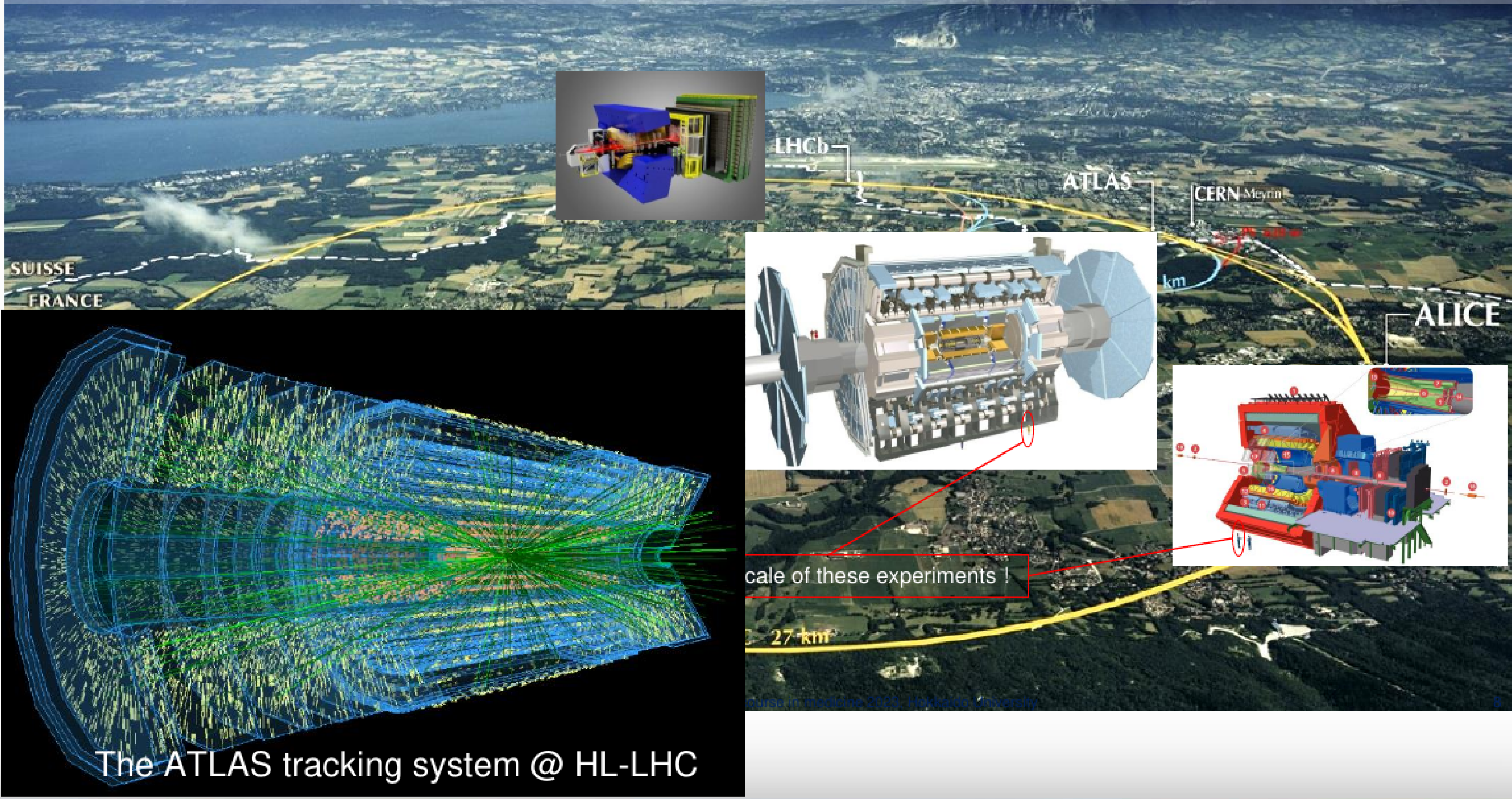



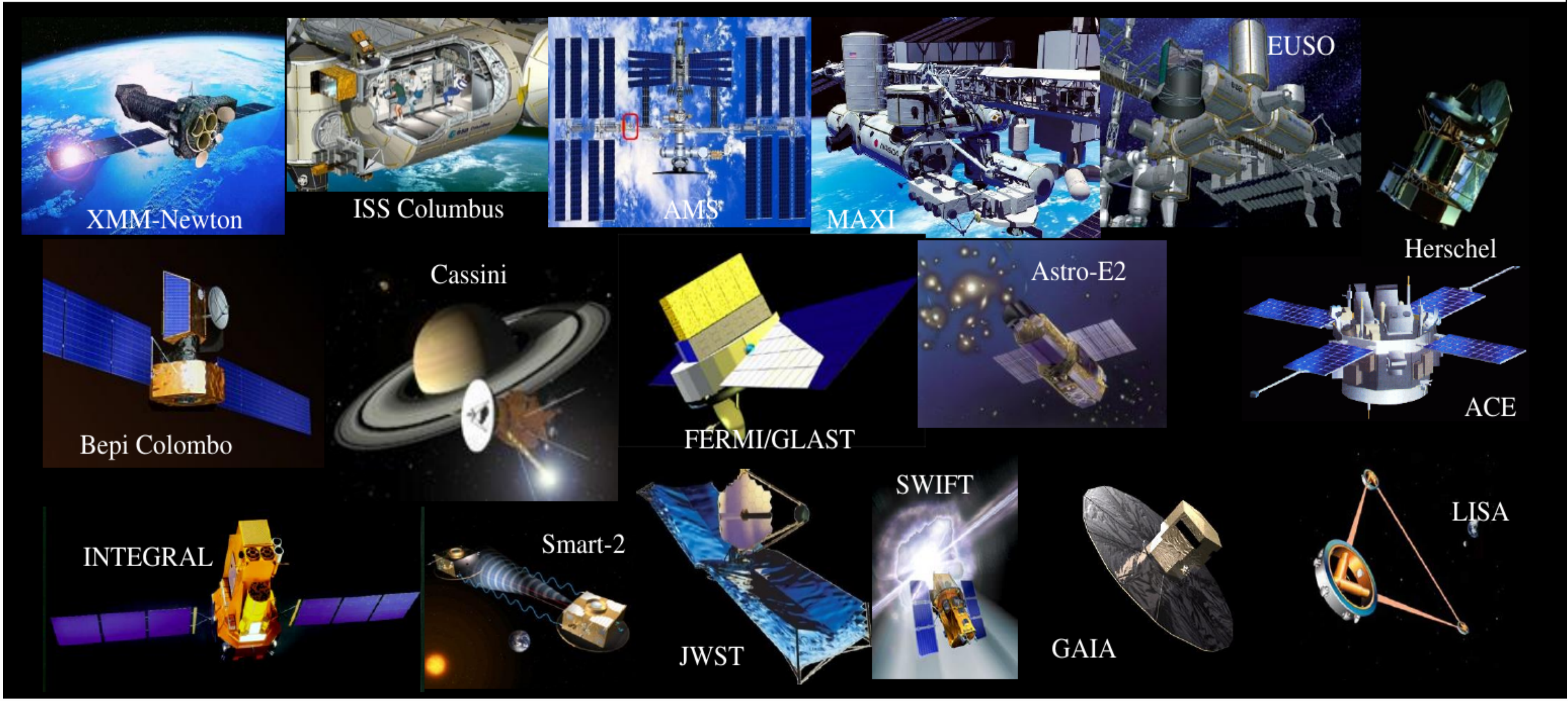
High Energy Physics : LHC @ CERN



High Energy Physics : LHC @ CERN

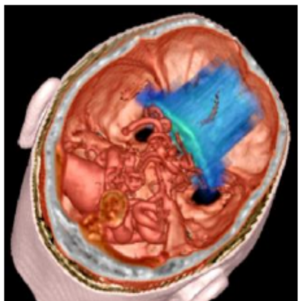


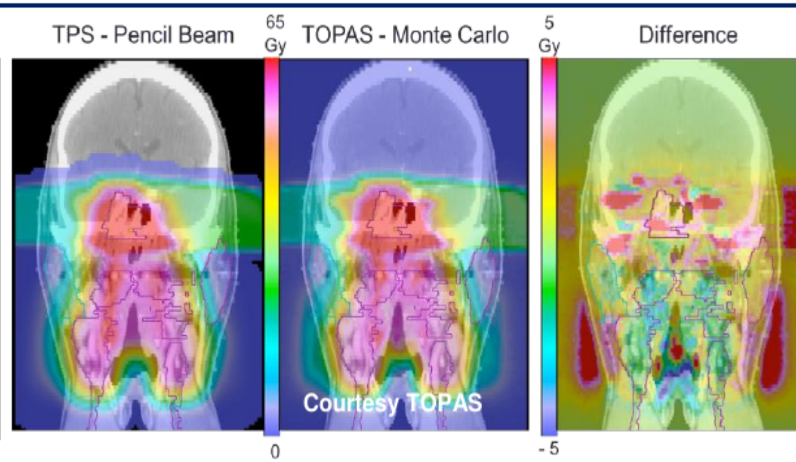
Geant4 in Space



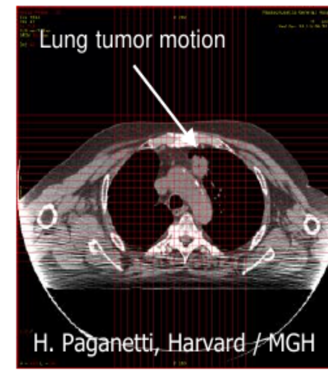
Beam Therapy, Brachytherapy

- Beam therapy

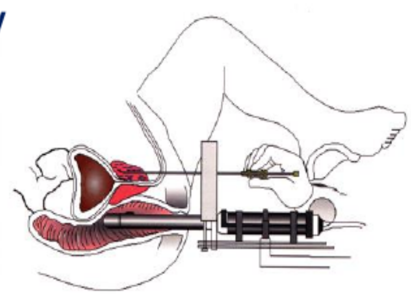

Dose by proton beam
Courtesy of gMocren

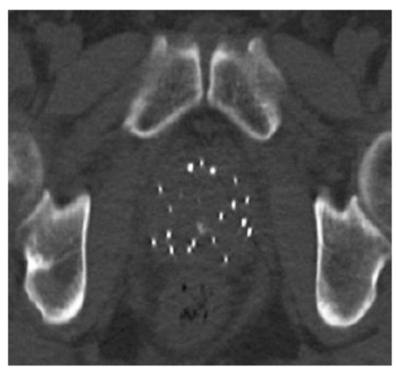


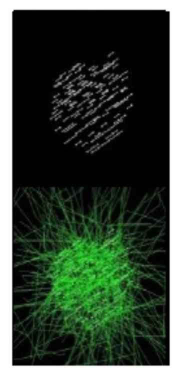
10 beams (6 directions + 4 boost)

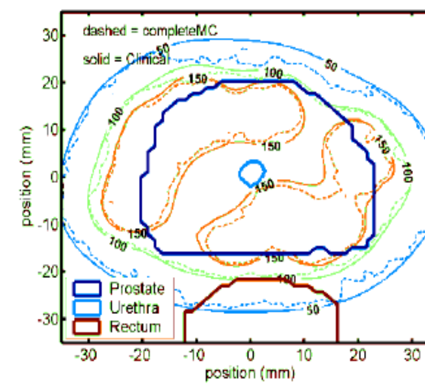


- Brachytherapy


Jean-François Carrier, CHUM

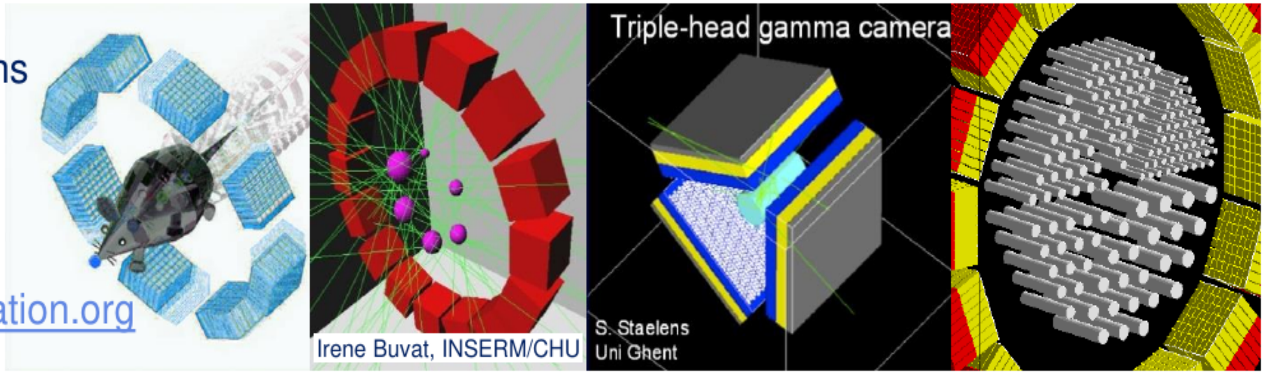




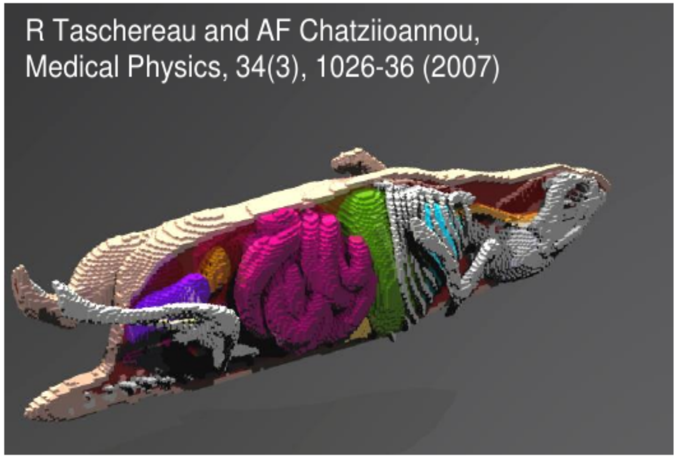


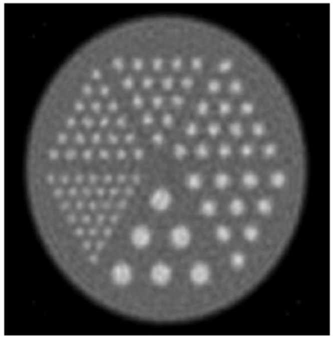
Imaging

- GATE
  - Toolkit for Imaging applications
  - based on the Geant4 toolkit
  - easier to use for Imaging applications
  - http://www.opengatecollaboration.org



- Ex of High resolution phantoms
  - (400 µm)^3 voxelized mouse phantom
  - Simulated map of 18-fluorine absorbed dose

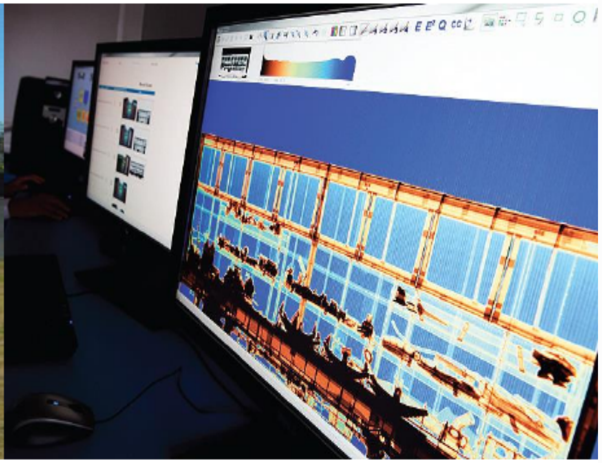


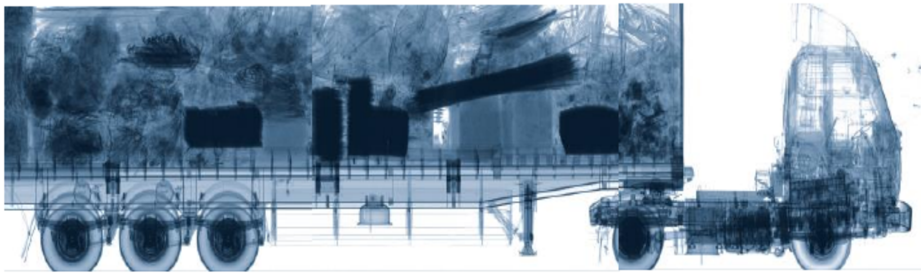
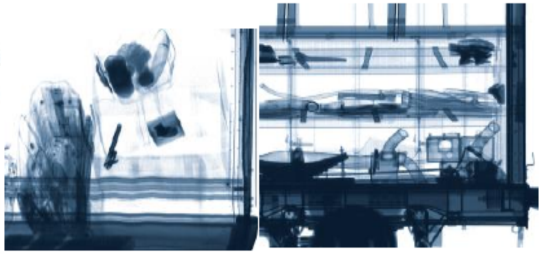


One reconstruction example, extracted from https://doi.org/10.1186/s40658-020-00309-8

Geant4 in Homeland Security : simulating X-ray cargo radiography





website: https://geant4.web.cern.ch/

GEANT4 A SIMULATION TOOLKIT

geant4.web.cern.ch

About | Download | Documentation | User Forum | Bug Reports | Events | Contact Us

Geant4

Toolkit for the simulation of the passage of particles through matter. Its areas of application include high energy, nuclear and accelerator physics, as well as studies in medical and space science.

Getting started

Get started

Everything you need to get started with Geant4.

I'm ready to start!

Download

Geant4 source code and installers are available for download, with source code under an open source license.

Latest: 11.3.2

Docs

Documentation for Geant4, along with tutorials and guides, are available online.

Read documentation

News

» More

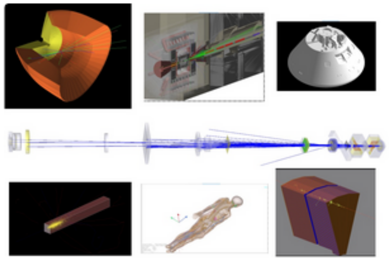
26 Jun 2025
Release 11.4.beta

25 Apr 2025
Release 11.3.2

21 Mar 2025
Release 11.3.1

11 Mar 2025
2025 Planned Features

06 Dec 2024
Release 11.3



About us

Collaboration

Contribute

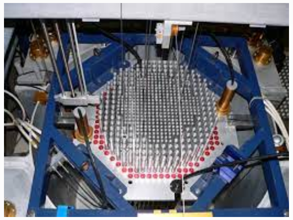
Let's have a look!

Monte Carlo

Geant4 approach: Monte Carlo method

Monte Carlo (MC) is a perfect example of **computer simulations** (not only computer) of the **real-world phenomena**

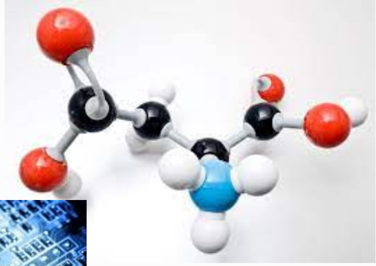


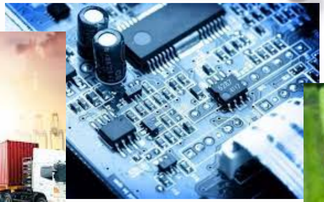


Monte Carlo applications:

- **Physics**: particle physics, astrophysics, nuclear physics, radiation damage,...
- **Medicine**: radiation therapy, nuclear medicine, computer tomography,…
- **Chemistry**: molecular modeling, semiconductor devices,...
- **Finance**: financial market simulations, pricing, forecast sales, currency,…
- **Optimization problems**: manufacturing, transportation, health care, agriculture,...
- **Data production** for **neural nets**
- And **much more**!







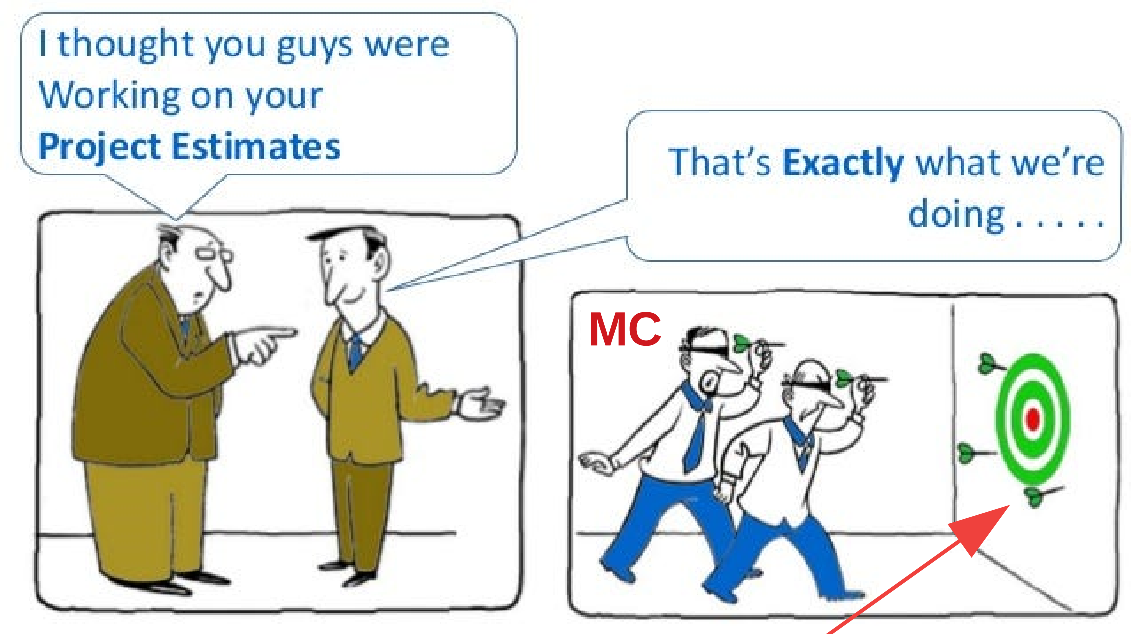

MC vs **Neural Nets**:
slower but more accurate and controllable

The simplest Monte Carlo example: probabilities of roulette

What is the probability of red?

- Observe the result **many times** (it is not necessary to stake:)
- Count the total of red wins: $\mathbf{N_{red}}$
- Count the total of games: $\mathbf{N_{total}}$
- The measured probability of red will be: $\mathbf{P_{red}} = \mathbf{N_{red}}/\mathbf{N_{total}}$
- If $\mathbf{N_{total}} \to \infty \Rightarrow \mathbf{P_{red}} \to \mathbf{P_{red\ true}} = 18/(18+18+1) = 0.486$

Monte Carlo is a simple and a general method



The **Monte Carlo (MC)** method is a method to obtain **deterministic results** from **random** values

In other words, **try many times** and **count** the **total** of the outcomes you like

- Generate **N random points** $\vec{x}_i$ in the problem space
- Calculate the **score** $f_i = f(\vec{x}_i)$ for the N points
- Calculate the **result** of your **average score**: $\boxed{\bar{f} = \frac{1}{N}\sum_{i=1}^{N} f_i}$
- According to the **Central Limit Theorem**, $\bar{f}$ will approach the **true** average value $\boxed{\langle f \rangle = \lim_{N \to \infty} \bar{f}}$

MC example: Laplace's method of calculating π (1886)

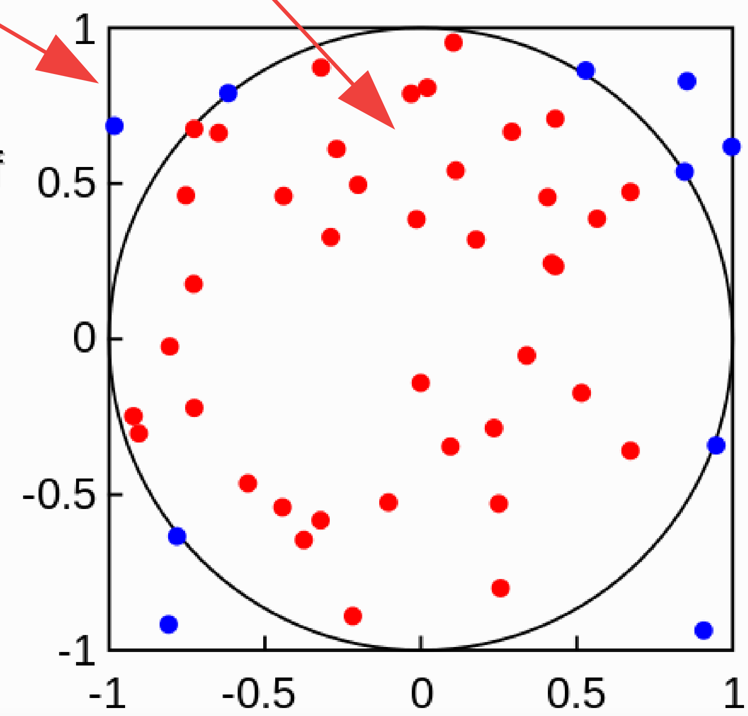
- **Side** of the square = 1
- Area of the **square** = **A** = 4
- Area of the **circle** is integral we are calculating: $\boxed{I=\pi}$

$$f_i=f(\vec{x}_i)=\begin{cases}1, if\ \vec{x}_i\in I\\ 0, if\ \vec{x}_i\notin I\end{cases}$$

- Everything we need is to **count** the number of points $\vec{x}_i$ inside the circle: $N_c=N_{\vec{x}_i\in I}=\sum_{i=1}^{N} f_i$

- This will give the value of our integral:

$$I_{MC}=\frac{A}{N}\sum_{i=1}^{N} f_i=\boxed{\frac{4}{N}N_c \underset{N\to\infty}{\rightarrow} \pi}$$



Monte Carlo numerical integration: extremely useful for multidimensional integrals!

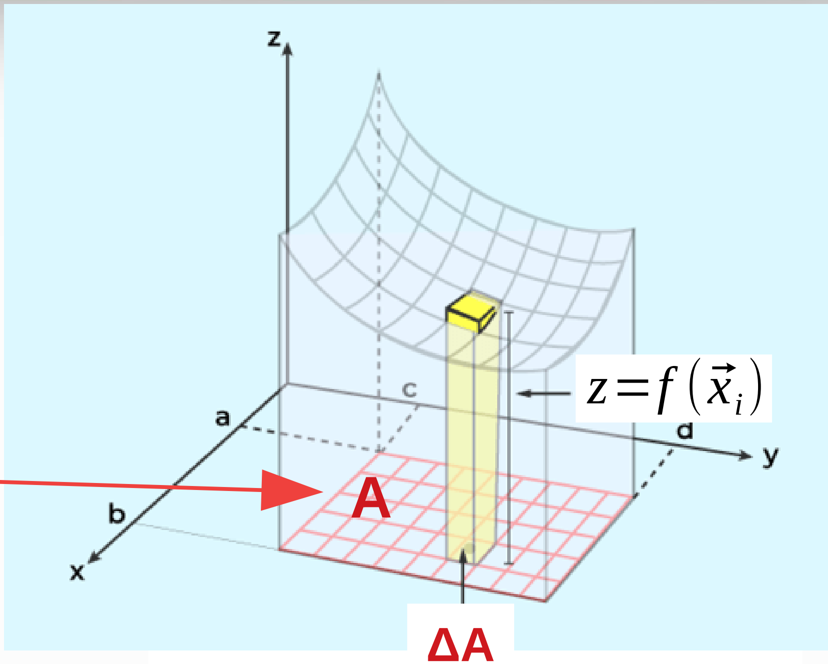
$$A = \int_A d\vec{x}_i ; \qquad d\vec{x}_i = dx_{1i} dx_{2i} dx_{3i} \ldots = dA$$

$$I = \int_A f(\vec{x}_i) d\vec{x}_i - ?$$

Idea is exactly the same!

- Generate **N random points** in $\vec{x}_i \in A$
- Calculate the **score** $f_i = f(\vec{x}_i)$ for the N points
- Calculate the **result** of your **integral**:

$$I = \int_A f(\vec{x}_i) d\vec{x}_i \approx I_{MC} = \sum_{i=1}^{N} f_i \Delta A = \frac{A}{N} \sum_{i=1}^{N} f_i = A \bar{f}$$

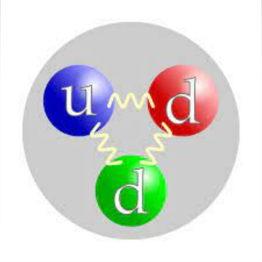


$$\Delta A = \frac{A}{N}$$

- Following the **Central Limit Theorem**, I_{MC} will approach the **true** integral value:

$$I = \int_A f(\vec{x}_i) d\vec{x}_i = \lim_{N \to \infty} I_{MC} = A \lim_{N \to \infty} \bar{f}$$

History of Monte Carlo



- Fermi (1930): random method to calculate the properties of the newly discovered neutron
- Manhattan project (40’s): simulations during the initial development of thermonuclear weapons. Von Neumann and Ulam coined the term “**Monte Carlo**”

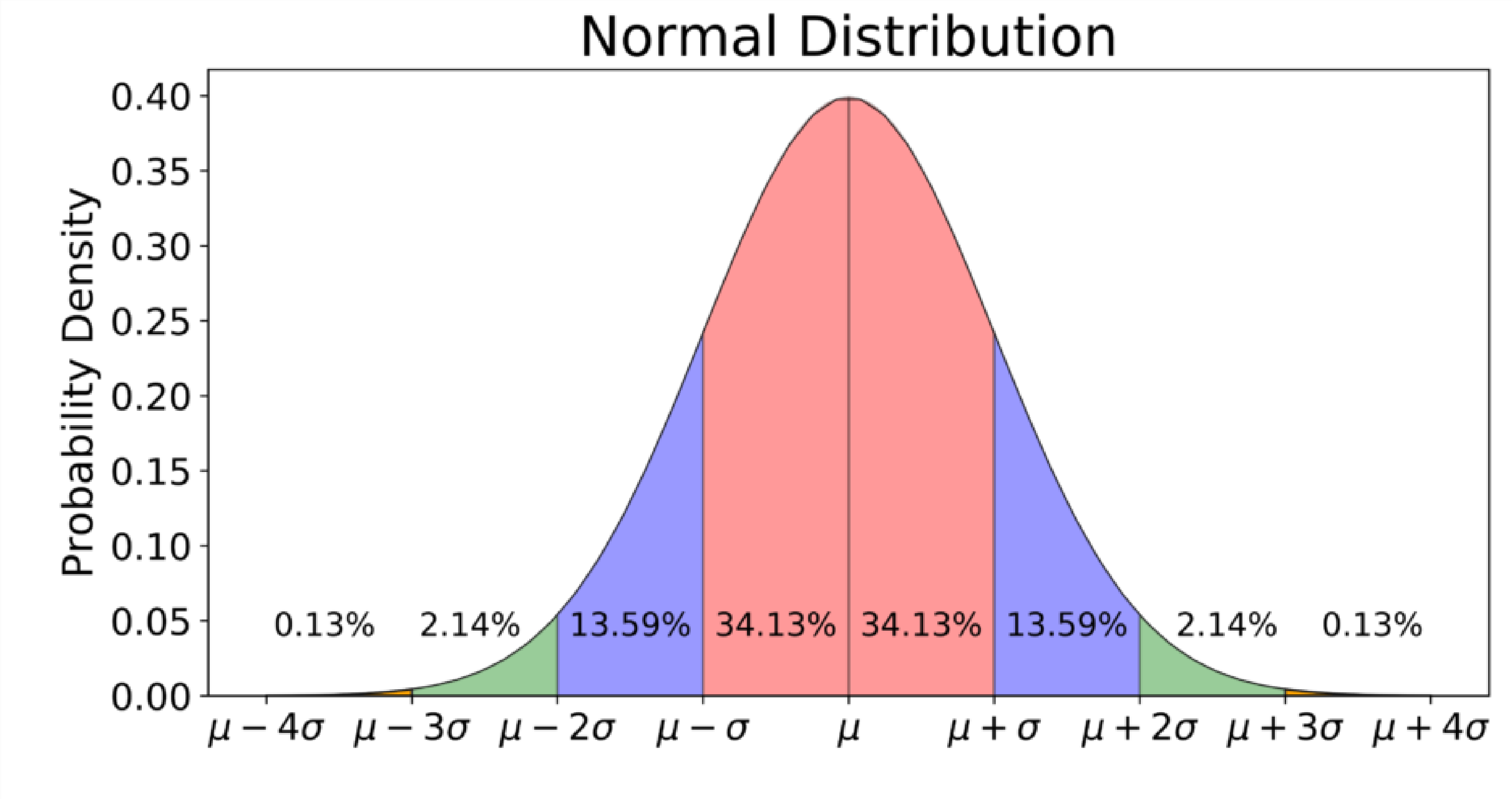
- Metropolis (1948) first actual Monte Carlo calculations using a computer (ENIAC)

- Berger (1963): first complete coupled electron-photon transport code that became known as ETRAN
- Exponential growth since the 1980’s with the availability of digital computers

However, sometimes the statistics is a problem

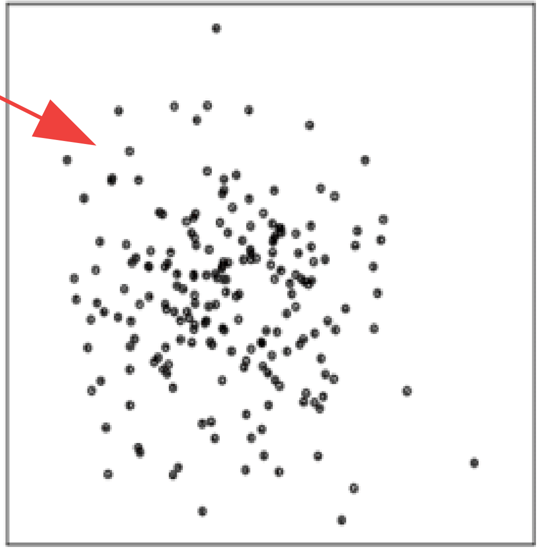
How does the **MC error** depend on the **MC statistics *N***?

First, we need to know about distributions: PDF and CDF



Probability Density Function (PDF)

- If we generate a set of **random variables** $\vec{x}_i \in A$, the **probability** of them is **not necessarily equal**. In some zones of *A* we can find more random variables and some of them less.



- However, we can define a function related to the probability of the generated points, so called ***probability density function*** **(PDF)**.

- **Probability Density Function (PDF)** $p(\vec{x}_i)$ of vector $\vec{x}_i$ is a function that has three properties:

For simplicity let's switch to the **1D case**:

1) belongs to some region A: $\boxed{\vec{x}_i \in A}$ $\boxed{a \leq x \leq b}$

2) is non-negative in this region: $\boxed{p(\vec{x}_i) \underset{\vec{x}_i \in A}{\geq} 0}$ $\boxed{p(x) \underset{a \leq x \leq b}{\geq} 0}$

3) is normalized: $\boxed{\int_A p(\vec{x}_i)\, d\vec{x}_i = 1}$ $\boxed{\int_a^b p(x)\, dx = 1}$

Cumulative Distribution Function (PDF)

PDF IS NOT A PROBABILITY
It is a probability density

Probability is the integral of PDF:

$$Prob\{x_1 \le x \le x_2\} = \int_{x_1}^{x_2} p(x)\,dx$$

- **Cumulative Density Function (CDF)** is a direct measure of probability:

$$\boxed{F(x) = Prob\{a \le x \le x'\} = \int_a^x p(x')\,dx'}$$

$$\boxed{Prob\{x_1 \le x \le x_2\} = F(x_2) - F(x_1)}$$

- **CDF** has the following **properties**:

1) F(a) = 0, F(b) = 1;

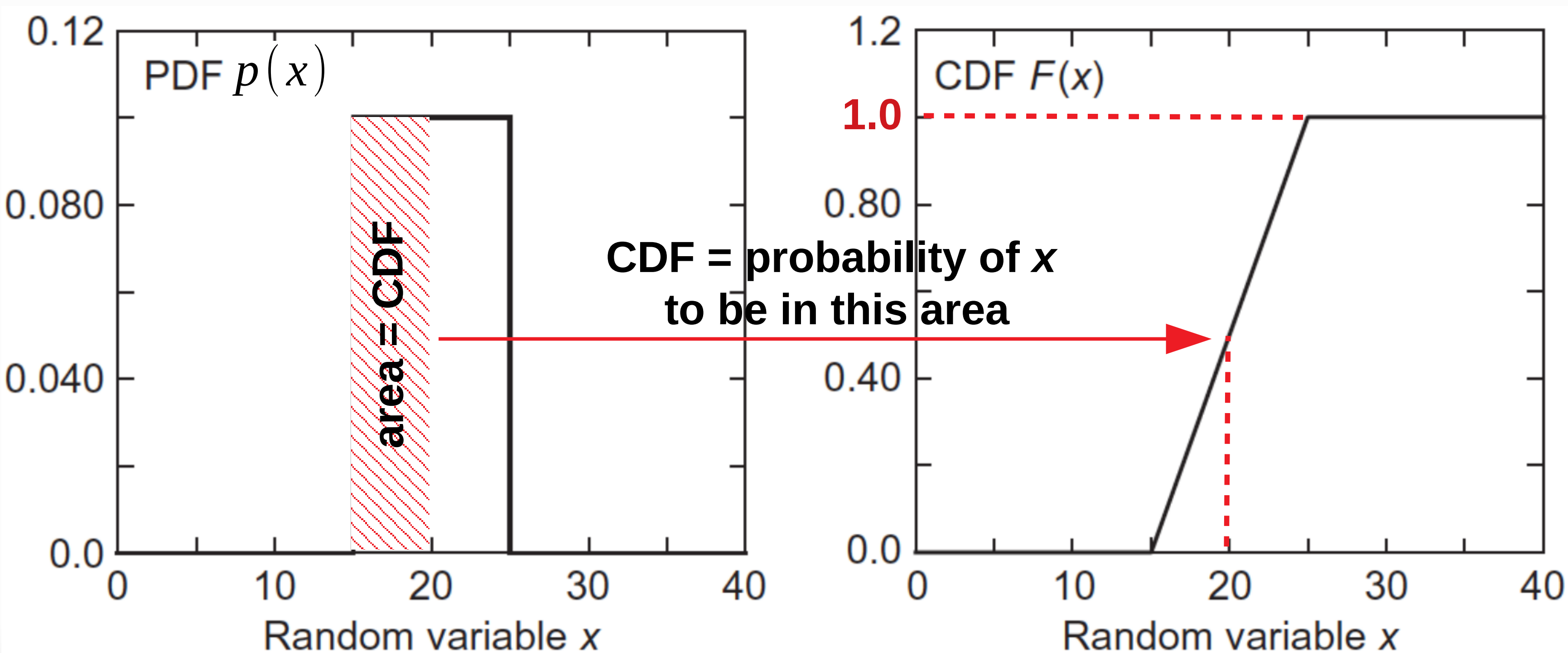
2) F(x) is monotonically increasing, since p(x)≥0.

Some example distributions – Uniform PDF

- The uniform (rectangular) PDF on the interval [a, b] and its CDF are given by

$$\boxed{p(x) = \frac{1}{b-a}}$$

$$\boxed{F(x) = \int_a^x \frac{1}{b-a}\,dx' = \frac{x-a}{b-a}}$$



Where we use the uniform distribution

- **Side** of the square = 1
- Area of the **square** = **A** = 4
- Area of the **circle** is integral we are calculating: $\boxed{I=\pi}$

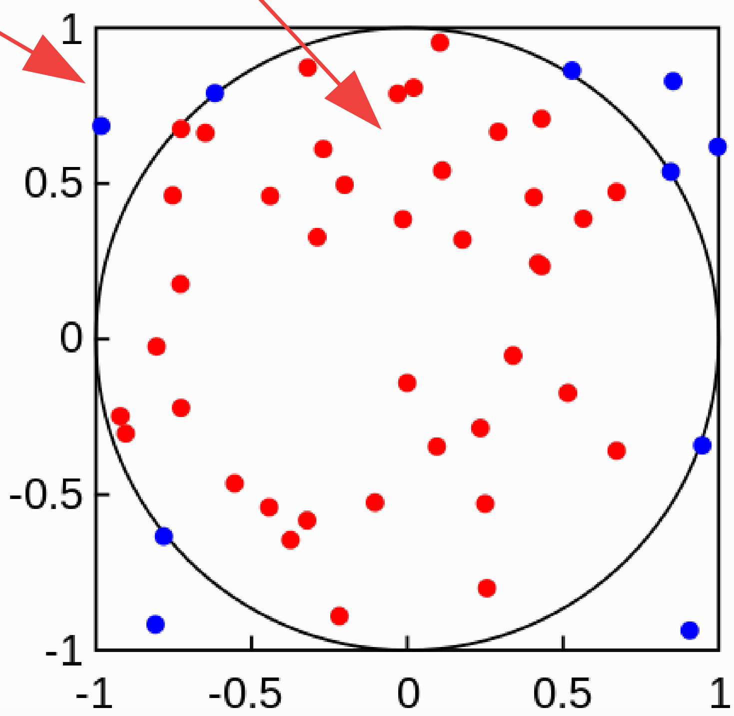
$$f_i = f(\vec{x}_i) = \begin{cases} 1, if\ \vec{x}_i \in I \\ 0, if\ \vec{x}_i \notin I \end{cases}$$

- Everything we need is to **count** the number of points $\vec{x}_i$ inside the circle:

$$N_c = N_{\vec{x}_i \in I} = \sum_{i=1}^{N} f_i$$

- This will give the value of our integral:

$$I_{MC} = \frac{A}{N} \sum_{i=1}^{N} f_i = \boxed{\frac{4}{N} N_c \xrightarrow[N\to\infty]{} \pi}$$

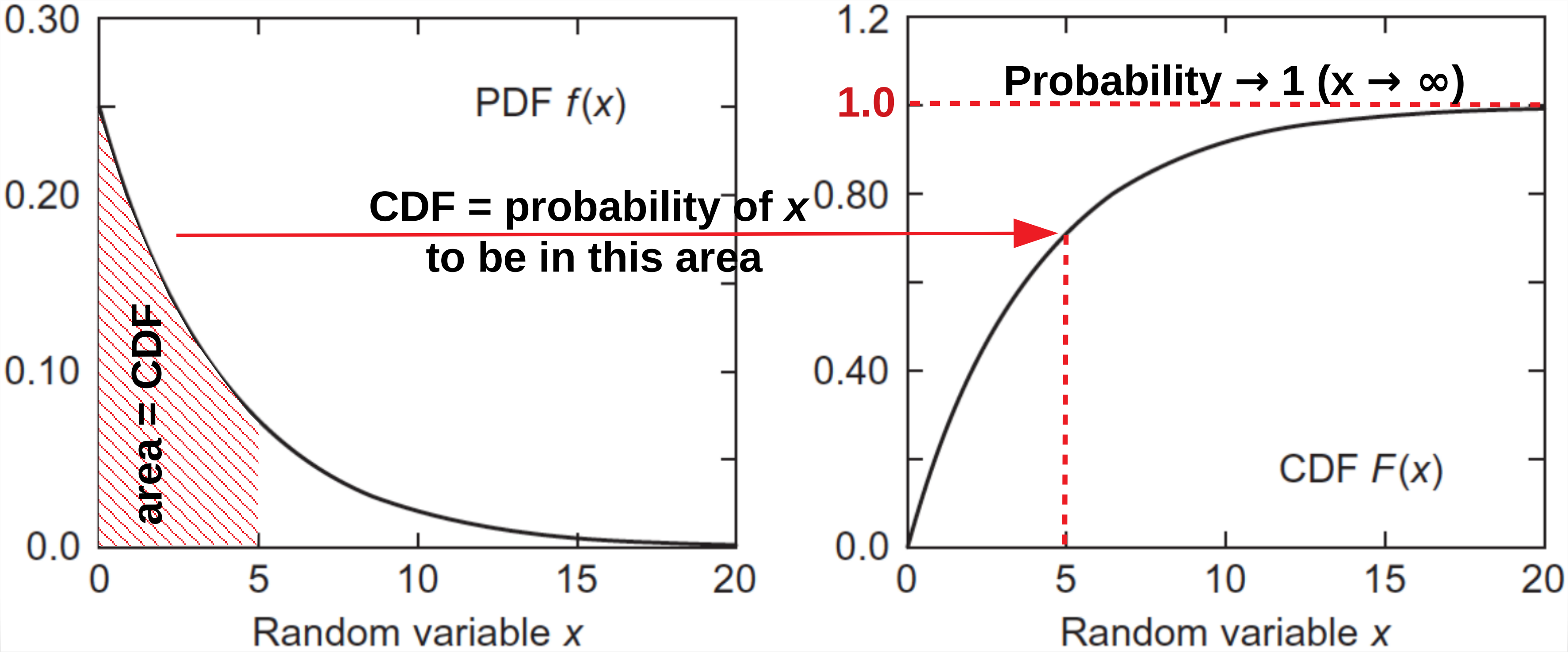


Some example distributions – Exponential PDF

- The exponential PDF on the interval [0, ∞] and its CDF are given by

$$\boxed{p(x) = p(x|a) = \alpha e^{-\alpha x}}$$

$$\boxed{F(x) = \int_0^x \alpha e^{-\alpha x'} dx' = 1 - e^{-\alpha x}}$$



Exponential distribution example: nuclear decay

- The time of nuclear decay is a random value with probability density function

$$\boxed{p(t) = \frac{1}{\tau} e^{-\frac{t}{\tau}}}$$

where **τ** is the **mean lifetime** of the nucleus; the **half-life** time $\mathbf{t_{1/2} = \tau \ln(2)}$

- The **probability** of **decay** at time **t** is calculated using the **CDF**:

$$\boxed{P_{decay}(t) = F(t) = \int_0^t \frac{1}{\tau} e^{-\frac{t'}{\tau}} dt' = 1 - e^{-\frac{t}{\tau}} \in [0,1]}$$

- To use Monte Carlo to generate the decay time t one needs to replace $P_{decay}(t)$ by a random number $\xi \in [0,1]$:

$$\boxed{t = -\tau \ln(1-\xi) = -\tau \ln \xi}$$

- **Nuclear decay applications**: nuclear physics, nuclear reactors, nuclear medicine, SPECT, PET, ...

Mean, variance and standard deviation

- Consider a function **z(x)**, where x is a random variable described by a PDF p(x).
- The function **z(x)** itself is a **random** variable. Thus, the **mean** value of z(x) is defined as:

$$\langle z \rangle \equiv \mu(z) \equiv \int_a^b z(x)\, p(x)\, dx$$

- Then, variance of z(x) is given as this

$$\sigma^2(z) = \langle (z(x) - \langle z \rangle)^2 \rangle = \int_a^b (z(x) - \langle z \rangle)^2 p(x)\, dx = \langle z^2 \rangle - \langle z \rangle^2$$

- The heart of a Monte Carlo analysis is to obtain an estimate of a mean value (a.k.a. **expected value**). If one forms the estimate

$$\boxed{\bar{z} = \frac{1}{N} \sum_{i=1}^{N} z_i = \frac{1}{N} \sum_{i=1}^{N} z(x_i)} \qquad \boxed{\langle z \rangle = \lim_{N \to \infty} \bar{z}}$$

- The variance of $\bar{z}$ is given as $\boxed{\sigma^2(\bar{z}) = \sigma^2\left(\frac{1}{N} \sum_{i=1}^{N} z_i\right) = \frac{1}{N^2} \sum_{i=1}^{N} \sigma^2(z) = \frac{1}{N} \sigma^2(z)}$

Monte Carlo error

- The Monte Carlo error is given by the standard deviation of the expected value:

$$\sigma(\bar{z}) = \frac{\sigma(z)}{\sqrt{N}}\,;\ \sigma(z) = \sqrt{\sum_{i=1}^{N} \left(z_i - \langle z \rangle\right)^2 / N}$$

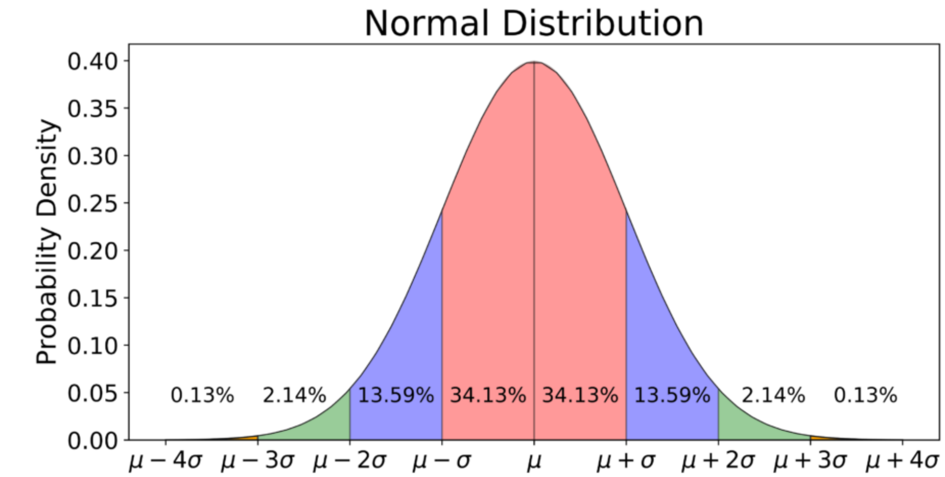
- Since in MC we don't know the true value $\langle z \rangle$, we should use corrected ("unbiased") sample standard deviation:

$$s(z) = \sqrt{\sum_{i=1}^{N} \left(z_i - \bar{z}\right)^2 / (N-1)}$$

- **Confidence coefficient**:

$$Prob\left\{\bar{z} - \lambda \frac{s(z)}{\sqrt{N}} < \langle z \rangle < \bar{z} + \lambda \frac{s(z)}{\sqrt{N}}\right\} \simeq \frac{1}{\sqrt{2\pi}} \int_{\lambda}^{\lambda} e^{-u^2/2}\, du$$

λ	confidence coefficient	confidence level
0.25	0.1974	20%
0.50	0.3829	38%
1.00	0.6827	68%
1.50	0.8664	87%
2.00	0.9545	95%
3.00	0.9973	99%
4.00	0.9999	99.99%



Higgs boson **discovery**:
λ=5 («5σ»)

However, sometimes the statistics is a problem

$$\boxed{\bar{z} = p_{win} = N_{win}/N = 1/14000605 = 7.14\cdot 10^{-8}}$$

z_i = 0 (loss) or 1 (win)

Standard deviation:

$$s(z) = \sqrt{\sum_{i=1}^{N}(z_i - \bar{z})^2/(N-1)} \approx \sigma(z)$$

$$= \sqrt{\sum_{i=1}^{N} z_i^2/N - \bar{z}^2} = [z_i = 0;1] = \sqrt{\sum_{i=1}^{N} z_i/N - \bar{z}^2}$$

$$= \sqrt{\bar{z} - \bar{z}^2} = \sqrt{\bar{z}(1-\bar{z})} = \sqrt{p_{win}(1-p_{win})}$$

$$\boxed{MC_{error}(1\sigma) = \frac{s(z)}{\sqrt{N}} \approx \sqrt{\frac{p_{win}(1-p_{win})}{N}}} \approx \sqrt{\frac{p_{win}}{N}} = \frac{\sqrt{N_{win}}}{N} = 1/14000605 = 7.14\cdot 10^{-8}$$

Avengers win with **(1 ± 1)/14000605** probability => they need more statistics (confidence level 68 %)

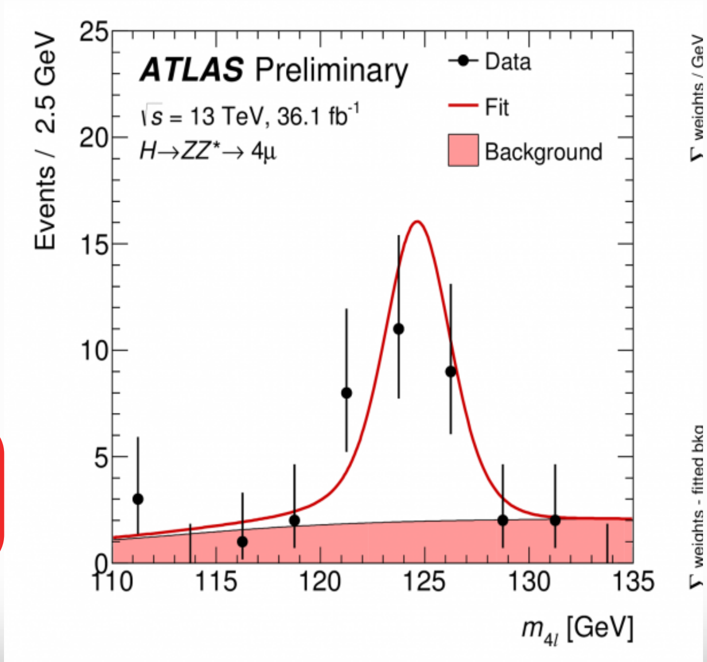
Real world case: particle physics

- **Decay** of an unstable particle itself is a **random process**
- This decay may happen through **different channels** => **Branching ratio:**

$\pi^+ \rightarrow \mu^+ \nu_\mu$	(99.9877 %)
$\pi^+ \rightarrow \mu^+ \nu_\mu \gamma$	(2.00 x 10^{-4} %)
$\pi^+ \rightarrow e^+ \nu_e$	(1.23 x 10^{-4} %)
$\pi^+ \rightarrow e^+ \nu_e \gamma$	(7.39 x 10^{-7} %)
$\pi^+ \rightarrow e^+ \nu_e \pi^0$	(1.036 x 10^{-8} %)
$\pi^+ \rightarrow e^+ \nu_e e^+ e^-$	(3.2 x 10^{-9} %)

Very low probability

Higgs boson events* **errorbars**



- The **statistical error** of decay events in a **decay channel** or of the **errorbars** in any **histogram** can be estimated using the same formula:

$$Error(1\sigma) = \sqrt{\frac{p(1-p)}{N}}$$

for **3σ** multiply it by 3, confidence level **99%**

A trick to reduce the statistics required for rare events

$$s(z)=\sqrt{\sum_{i=1}^{N}(z_i-\bar{z})^2/(N-1)}\approx\sigma(z)$$

$$\boxed{=\sqrt{\sum_{i=1}^{N} z_i^2/N-\bar{z}^2}\leqslant\sqrt{\sum_{i=1}^{N} z_i/N-\bar{z}^2}}$$

Let's have several relative wins instead of 1 definite:

$z_i \in [0,1]$

$z_{i\ wins}$ ={0.51,0.23,0.15,0.08,0.03}
(the average p_{win} is the same)

Avengers win with (**1.0** ± **0.6**)/14000605 probability (confidence level 68 %)

In particle physics use a **particle** probability **weight**:
weight = 1 - p_{decay}; $p_{decay} \in [0,1]$

GEANT4

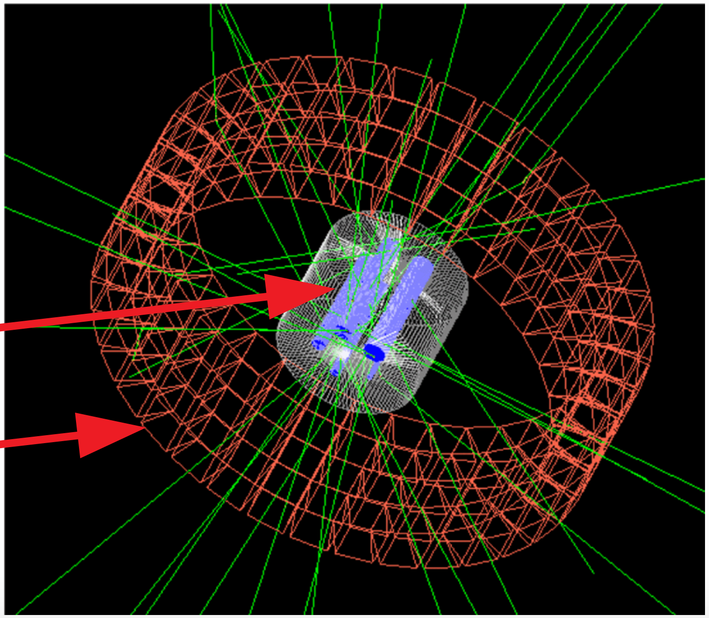
A SIMULATION TOOLKIT

Geant4*: a Monte Carlo simulation toolkit

- **Geant4** generates **primary beam** of particles randomly according the distribution set up.
- All the **Geant4** primary particles are simulated independently.
- Primary particles are **tracked** in the material, can **decay** and **produce secondary particles**, for instance **radiation.** This is simulated using various **Geant4 processes** most of which are **random**, which is also illustration of Monte Carlo.
- The **Geant4 output** is some **distribution** of particles as well as **scoring** of interesting events.

In **Positron Emission Tomography** (PET) we have (picture from **):

- a **source** of **gamma**-rays distributed in some space **randomly emitting** the photons and surrounded by some material
- A **detector** to **score** these **gamma**-rays



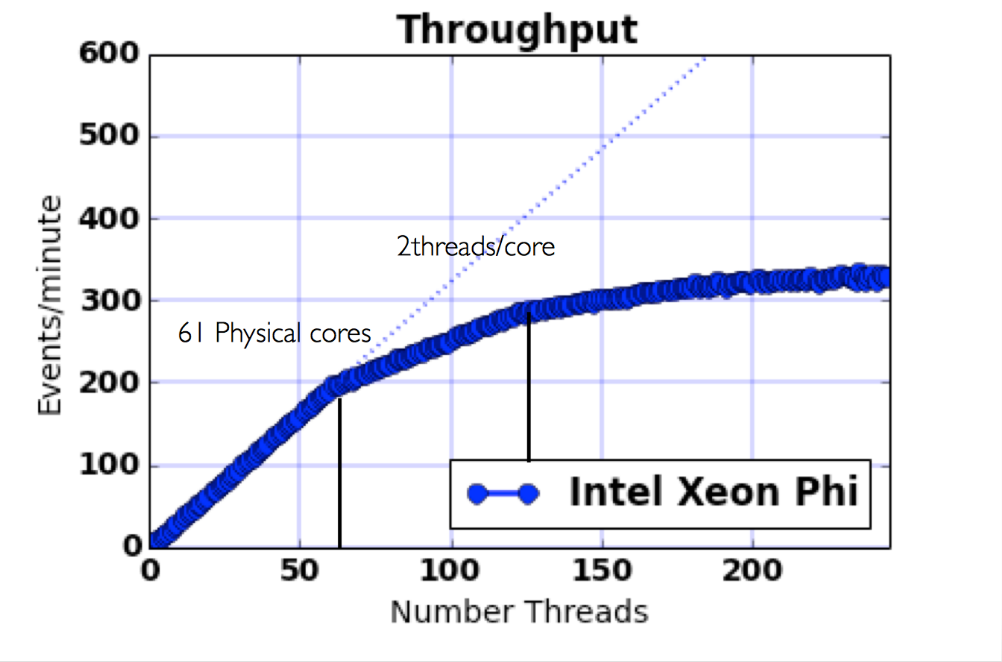
*https://geant4.web.cern.ch/

**D. P. Watts et al. Nature Communications, 12, 2646 (2021)

Monte Carlo parallelization => supercomputing

- All **Monte Carlo** points are **independent** => simple parallelization
- In **Geant4** all primary particles are automatically distributed between different cores of the CPU using **multithreading**
- **Geant4** includes also **MPI parallelization** to parallelize across on **multiple nodes**

Linear scaling on physical cores*



NURION@KISTI (Korea)


REAL
WORLD
MONTE CARLO
SIMULATION