Lecture 20:
AI Monte Carlo

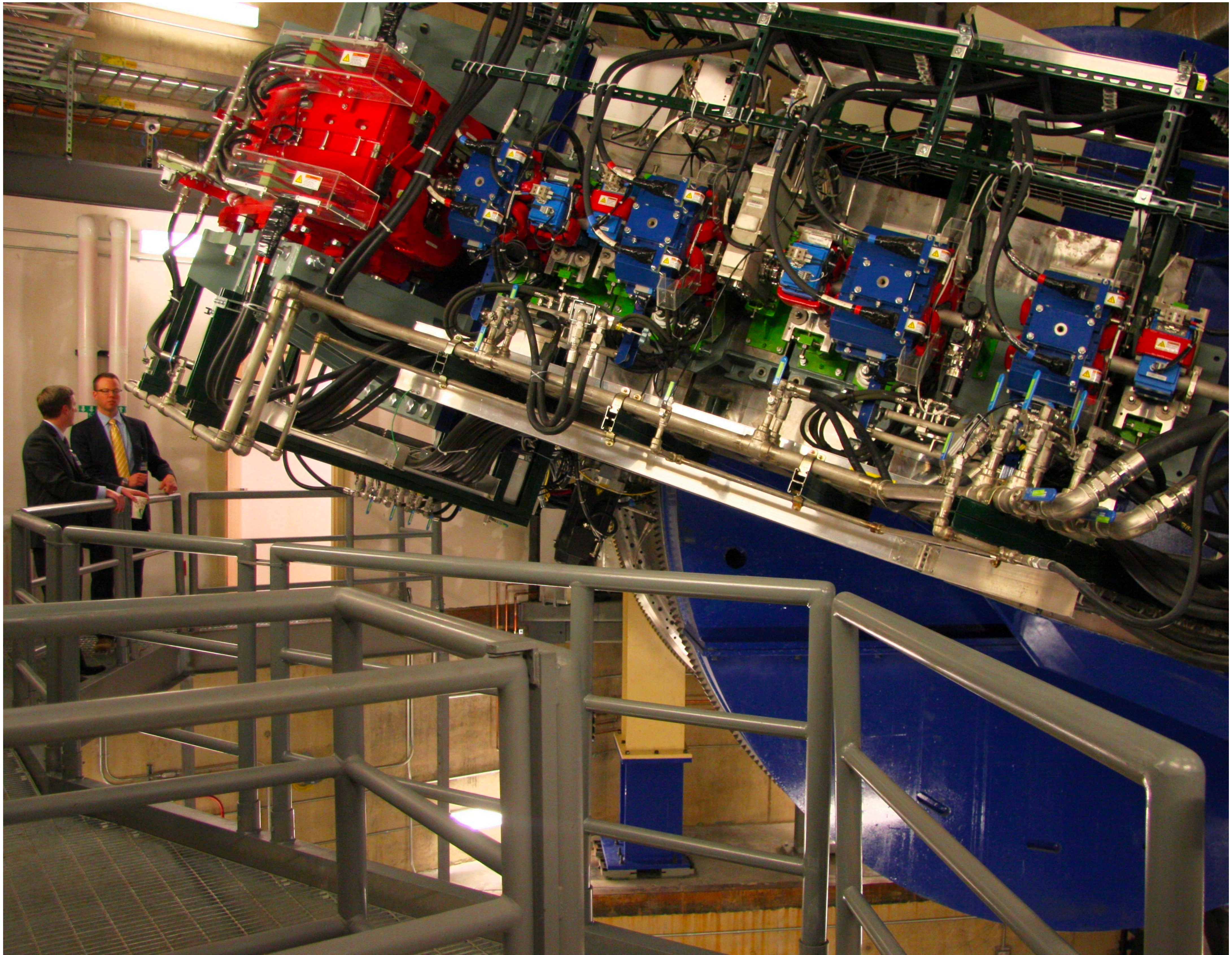
Proton Therapy

Proton Therapy Center at MGH

Typical Device

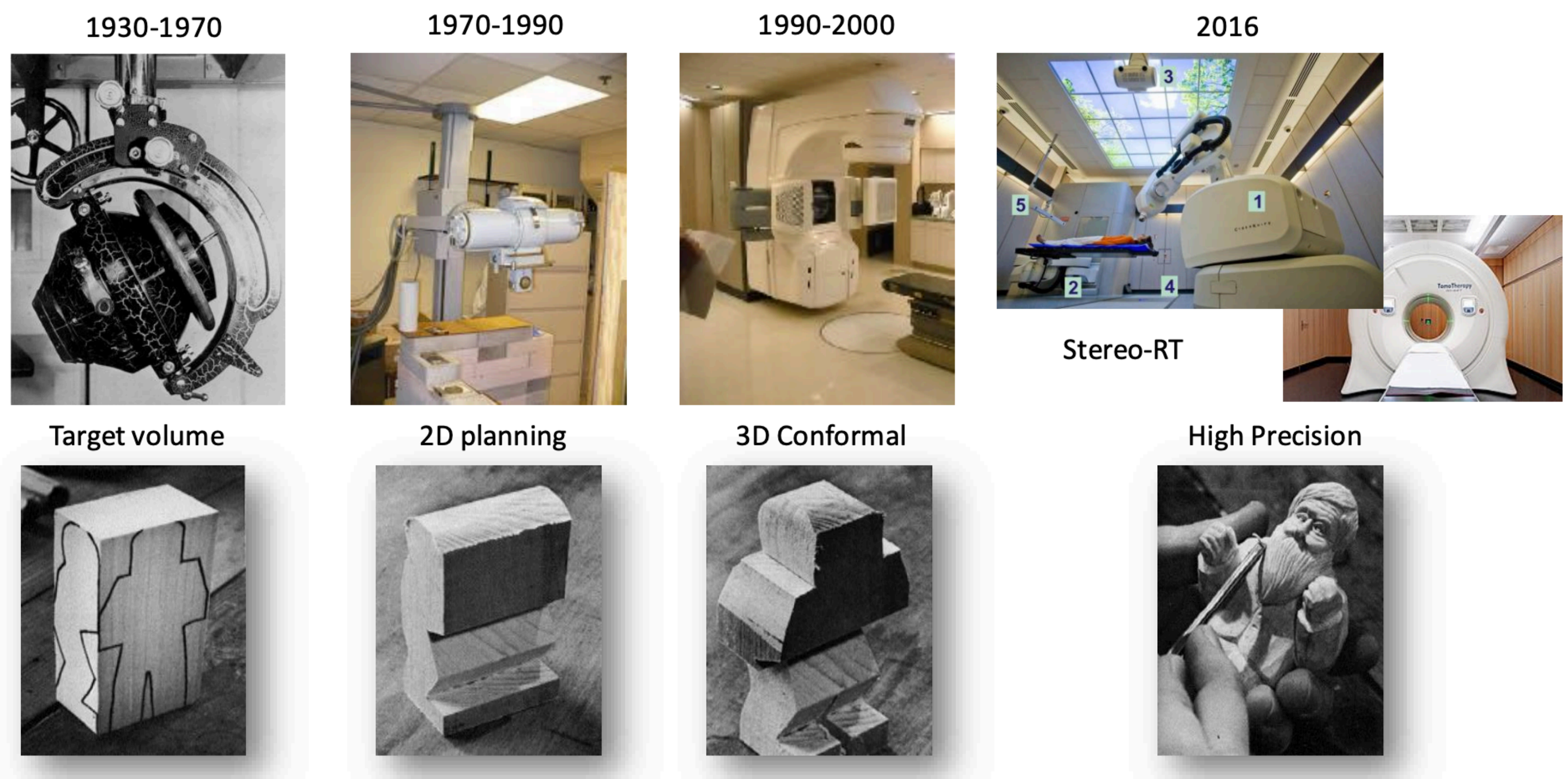
Particle Therapy Centre

Mayo Clinic



Radiation Therapy

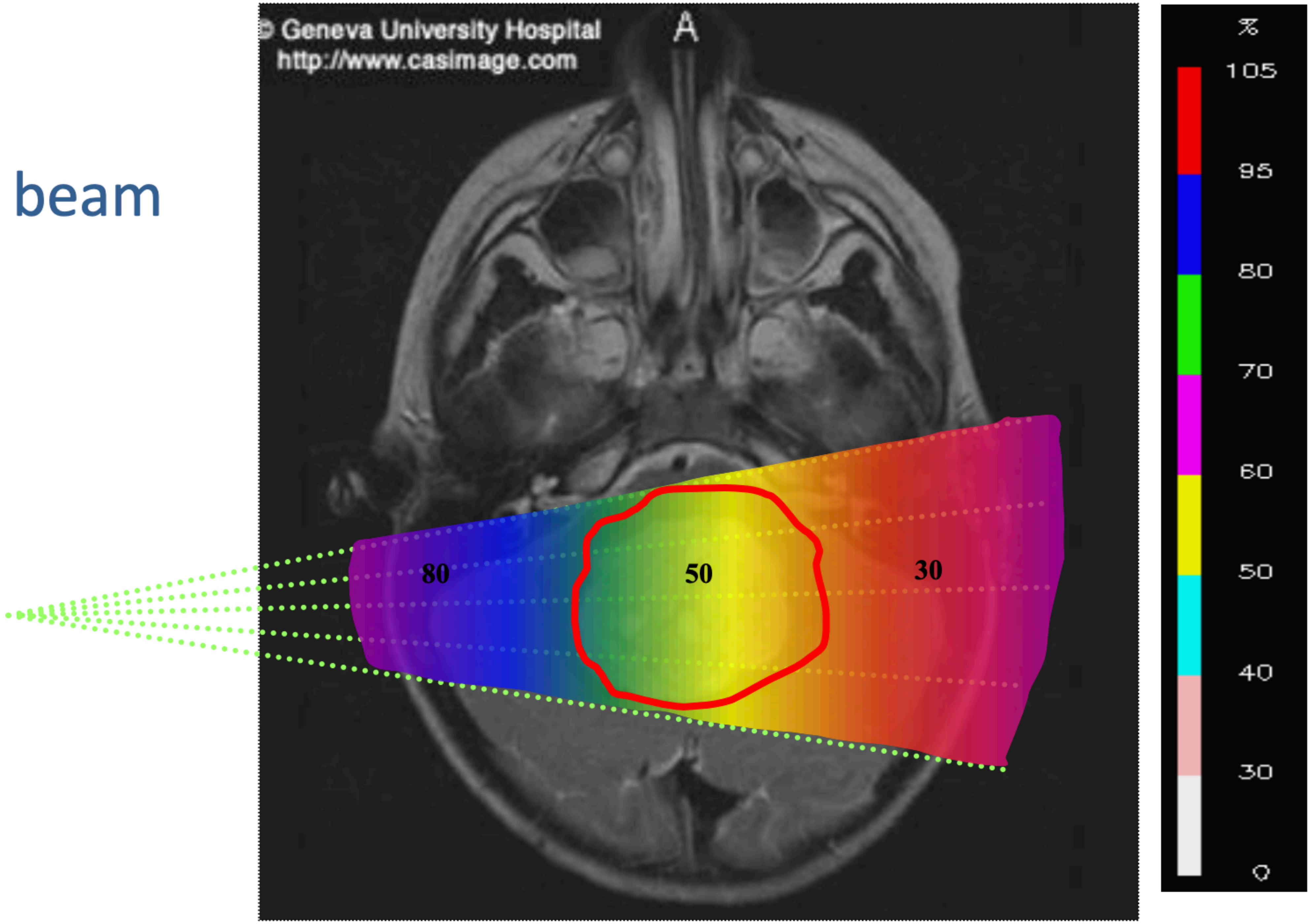
Fractionation and Enhanced precision



- To fight Cancer
  - Radiation therapy has had a long history of usage
  - Radiation is sent to a tumor to kill it
  - Critical when you can’t cut the tumor out

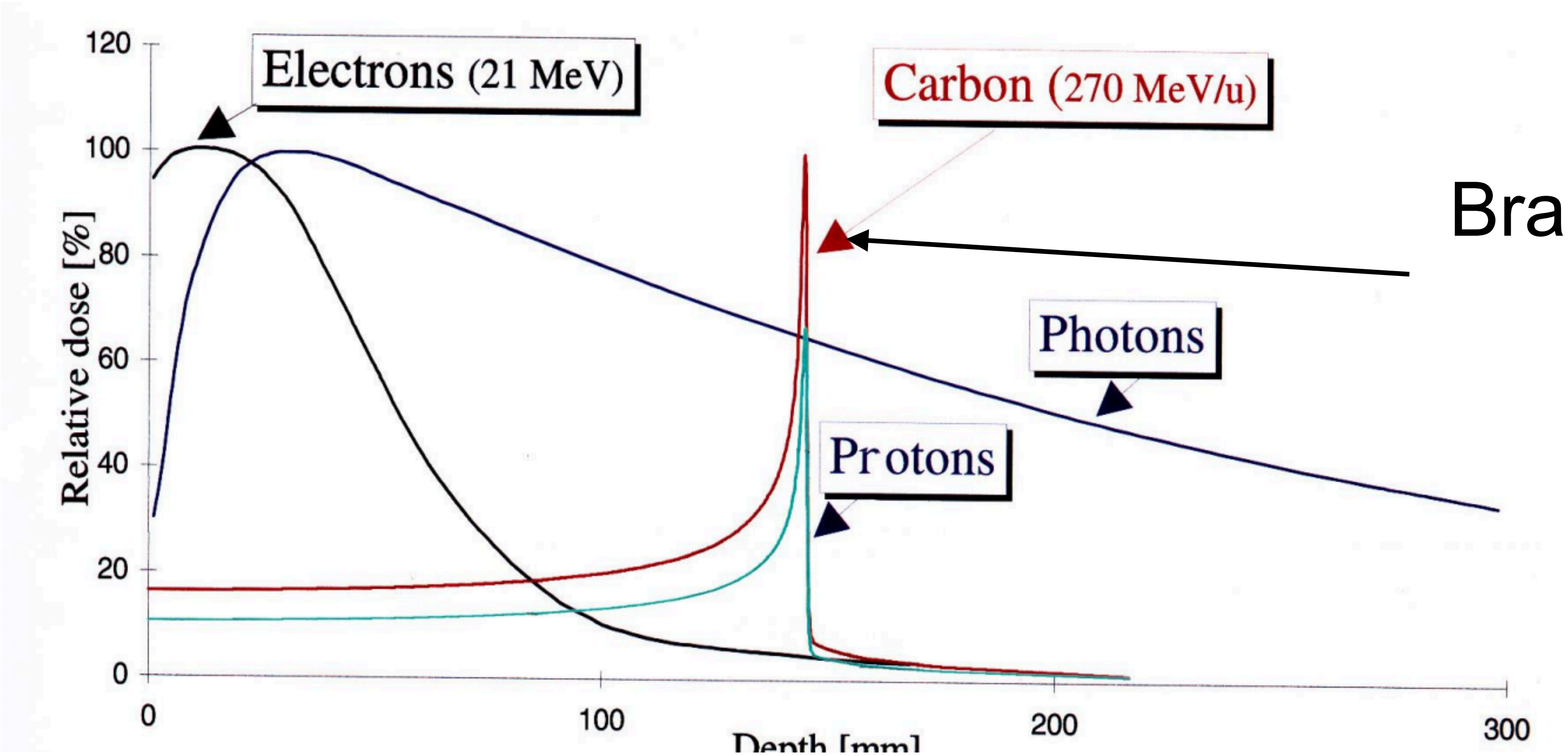
Classical Radiotherapy with X-rays

single beam



Hadron Therapy

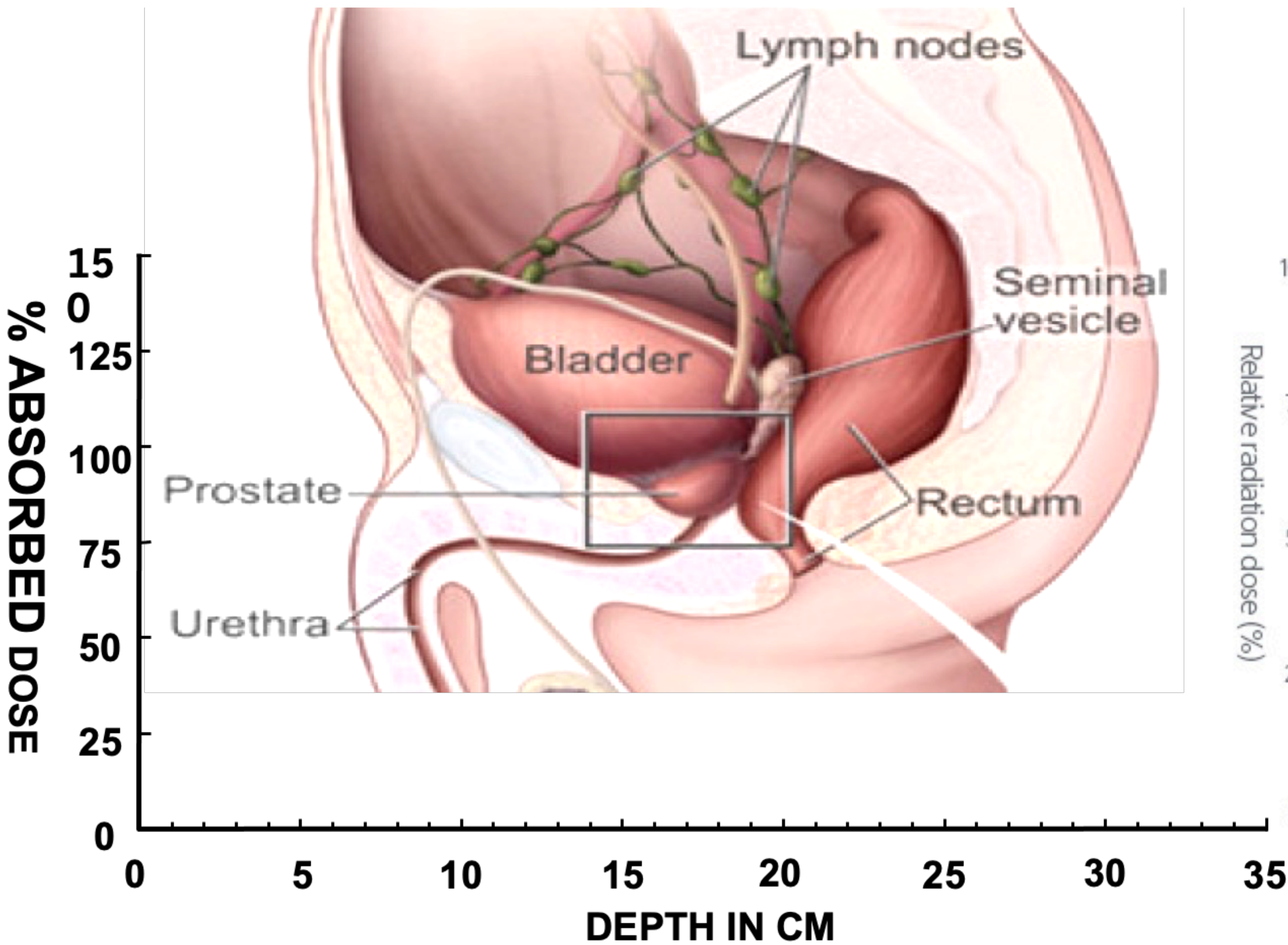
- Therapy
  - Hadrons allow you to control deposit
  - Can vary the depth of the hadrons through Bragg scatter

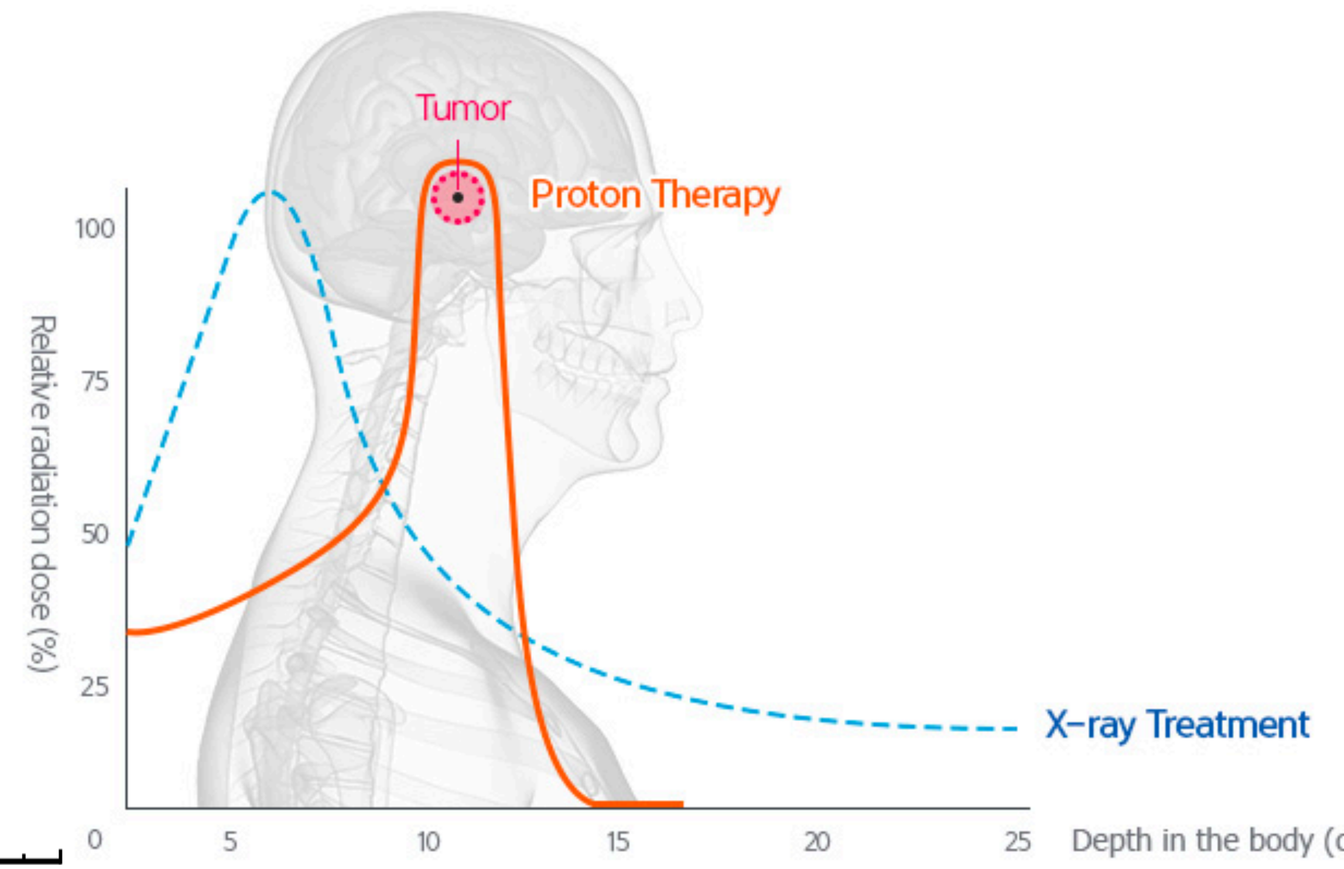


Bragg Scatter Peak

Proton Therapy

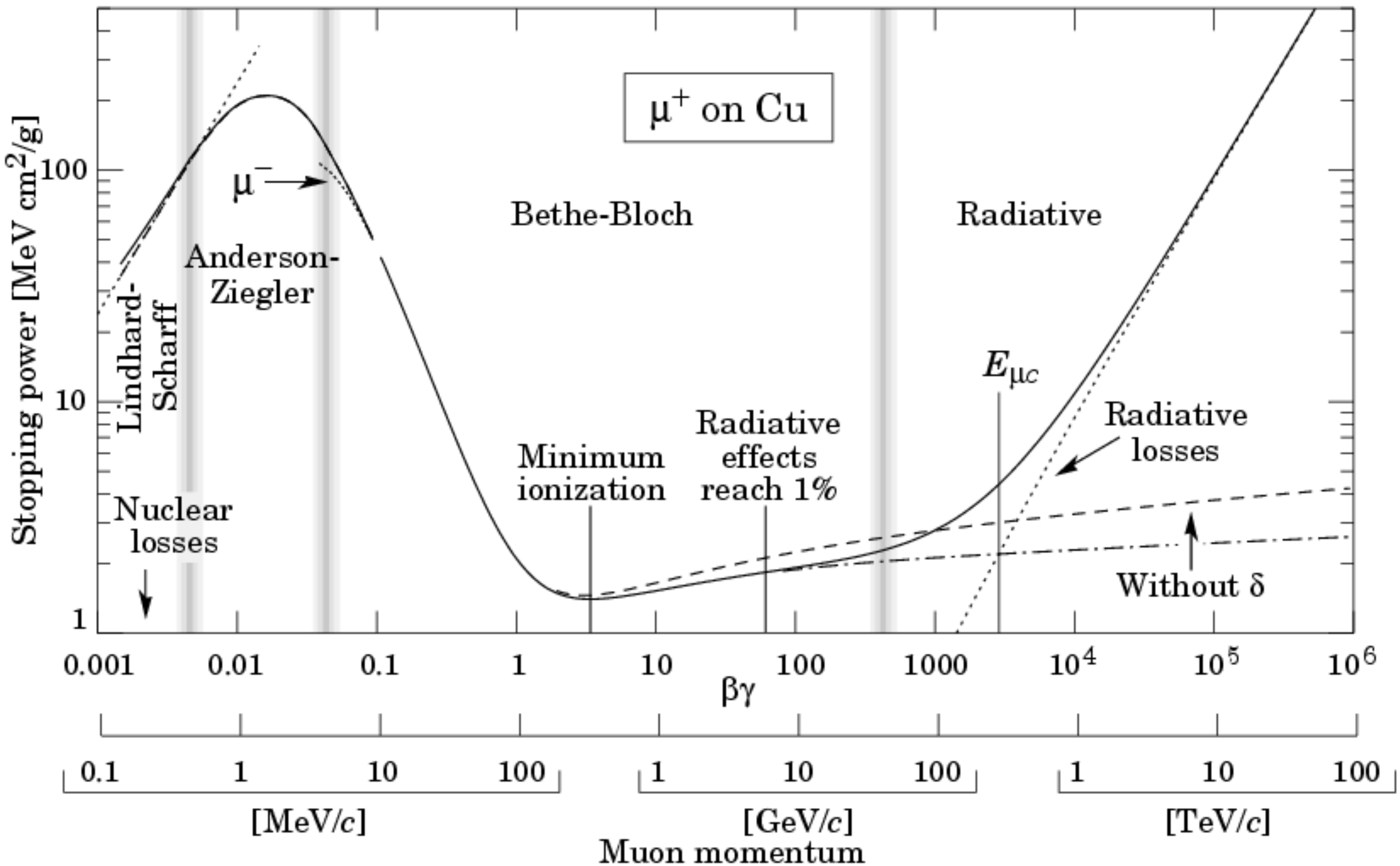
- Quickly has become the best way for
  - Prostrate cancer and Brain Cancer (scary!)
  - > 80% success rate





Bethe-Bloch Equation

- Charged Particles in matter are goverened by this equation



Protons Governed

$$\boxed{-\left\langle \frac{dE}{dx} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \ln \frac{2 m_e c^2 \beta^2 \gamma^2 T_{\max}}{I^2} - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right]} \; [\cdot \rho]$$

density

K $= 4\pi N_A r_e^2 m_e c^2 = 0.307$ MeV g^{-1} cm^2

$T_{max} = 2 m_e c^2 \beta^2 \gamma^2 / (1 + 2\gamma\, m_e/M + (m_e/M)^2)$

[Max. energy transfer in single collision]

z : Charge of incident particle

M : Mass of incident particle

Z : Charge number of medium

A : Atomic mass of medium

I : Mean excitation energy of medium

δ : Density correction [transv. extension of electric field]

$N_A = 6.022 \cdot 10^{23}$

[Avogardo's number]

$r_e = e^2/4\pi\varepsilon_0 m_e c^2 = 2.8$ fm

[Classical electron radius]

$m_e = 511$ keV

[Electron mass]

$\beta = v/c$

[Velocity]

$\gamma = (1-\beta^2)^{-2}$

[Lorentz factor]

Validity:

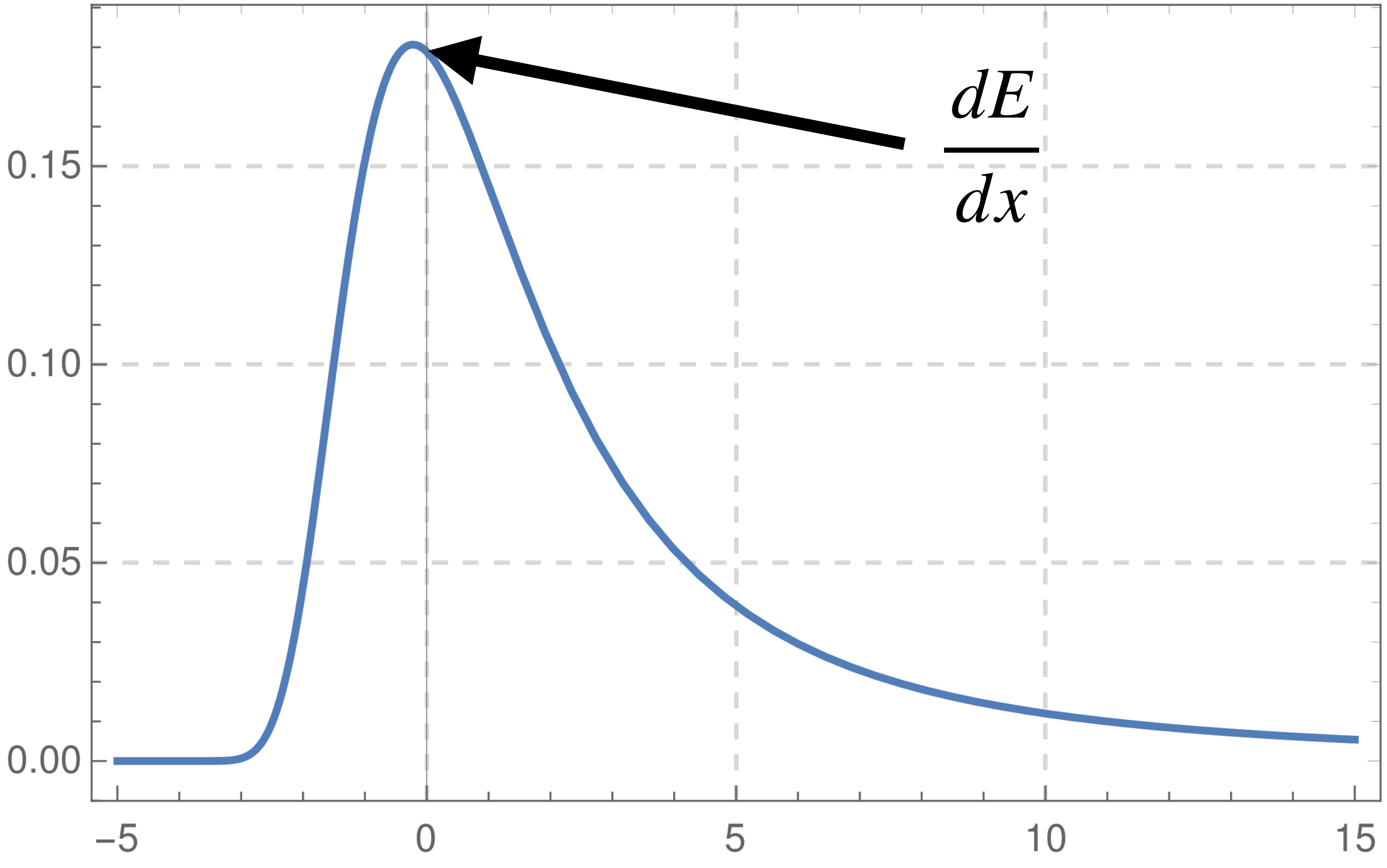
$.05 < \beta\gamma < 500$

$M > m_\mu$

Actual Energy Loss

- As we step along we lose energy by the Landau distribution

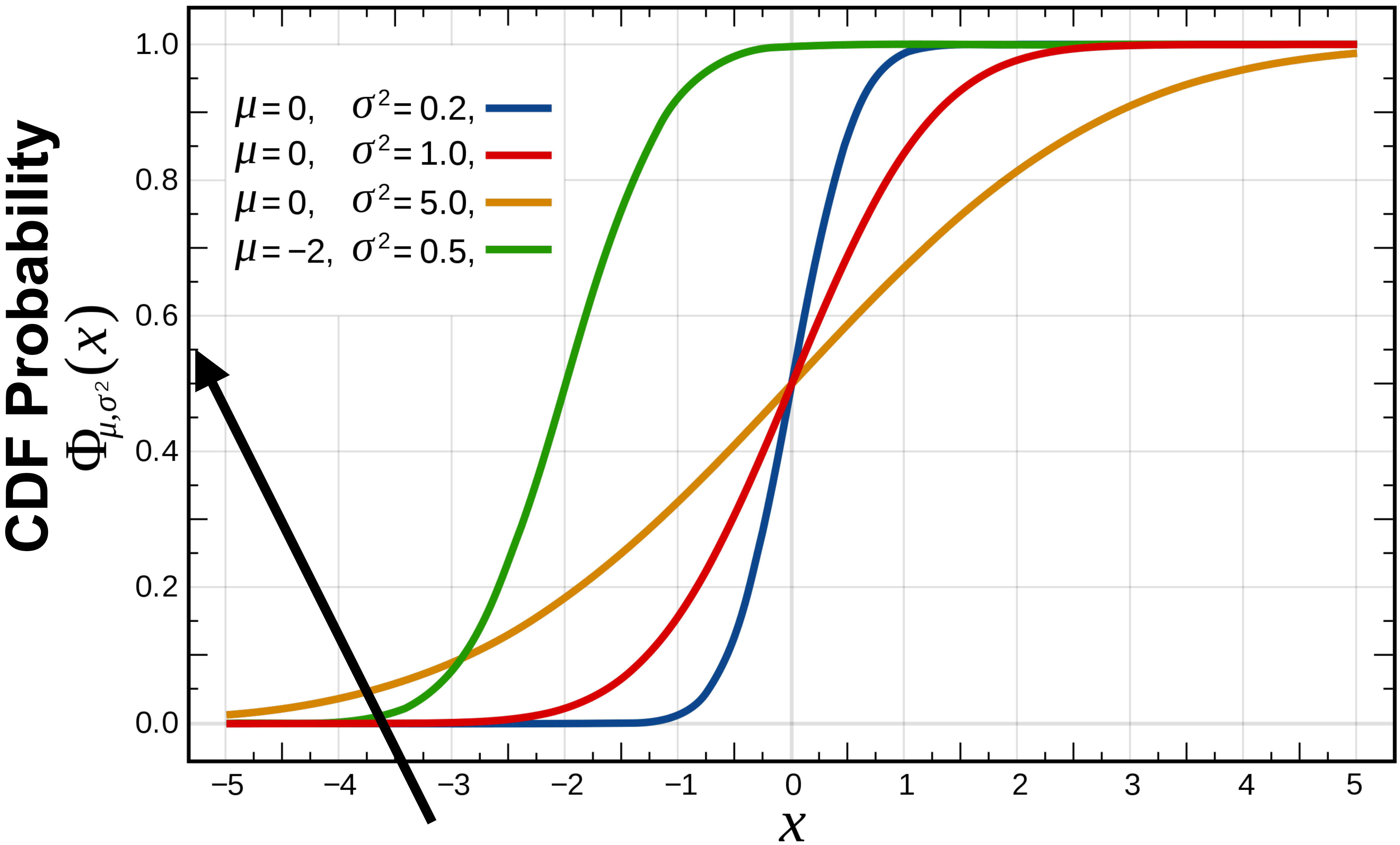
$$p(x) = \frac{1}{2\pi i} \int_{a-i\infty}^{a+i\infty} e^{s\log(s)+xs}\, ds,$$



Average of this distribution gives Bethe-Bloch

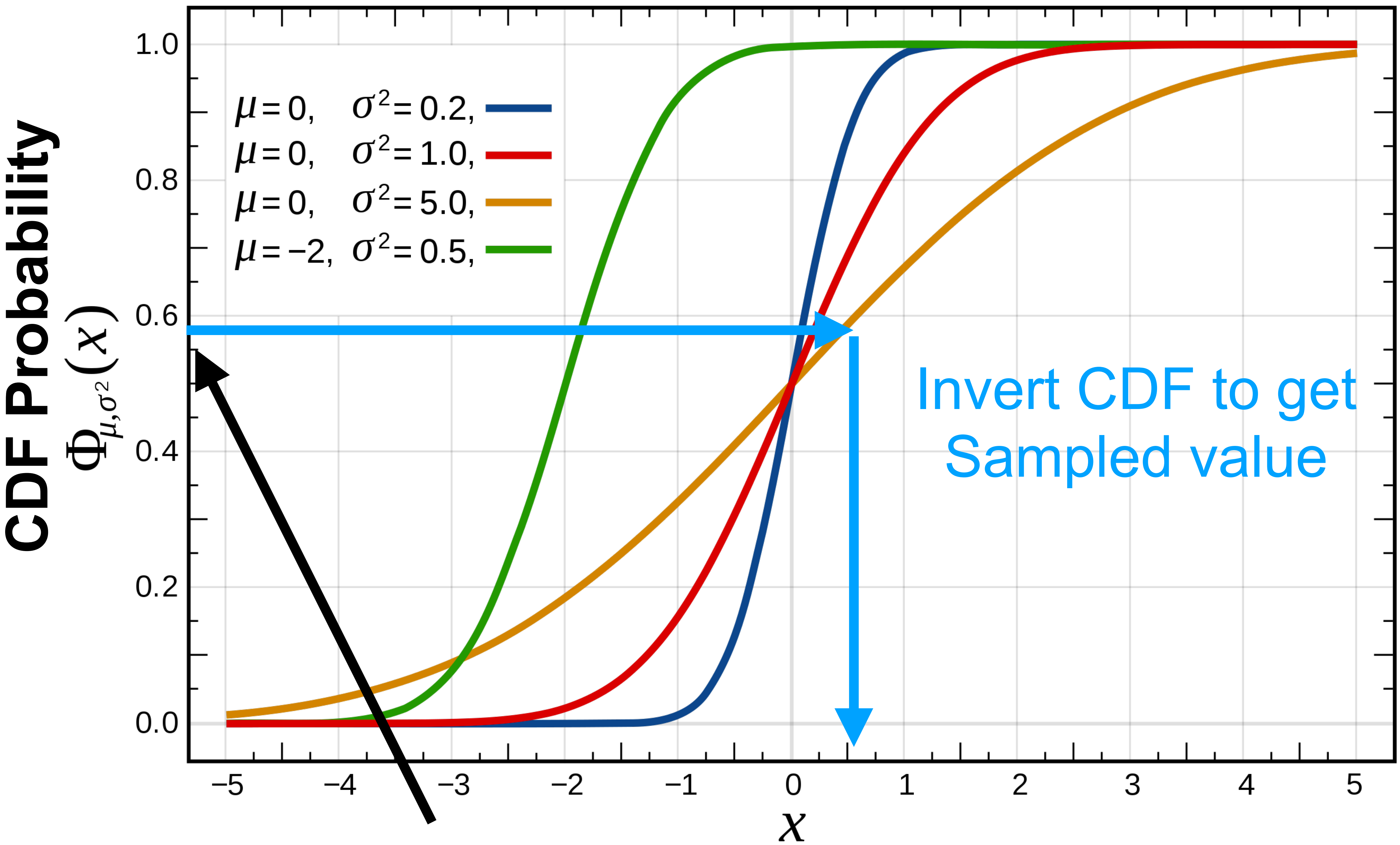
We can sample this
At each step

Sampling a Distribution



Sample from a p-value from 0 to 1 (flat 0 to 1)

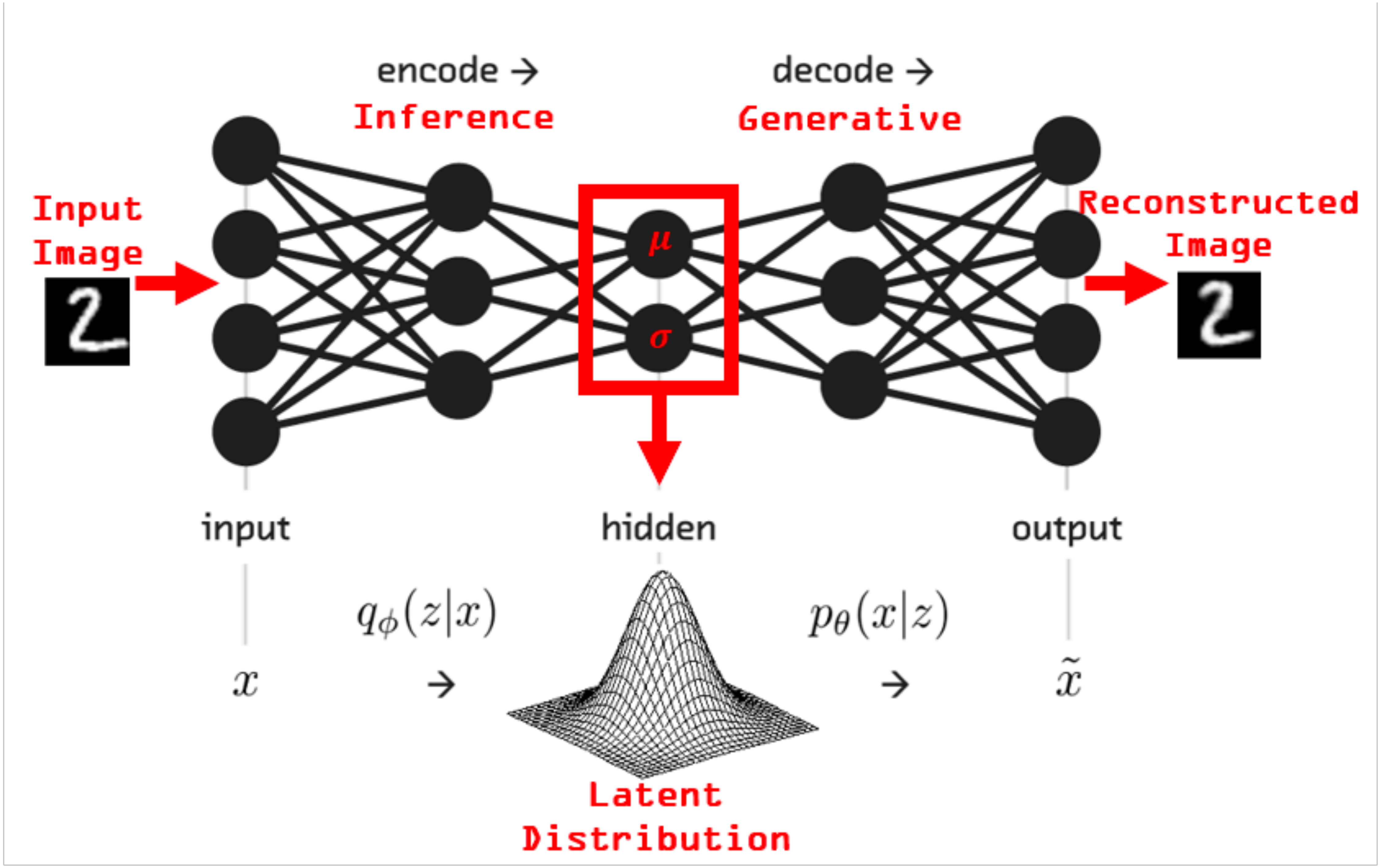
Sampling a Distribution



Sample from a p-value from 0 to 1 (flat 0 to 1)

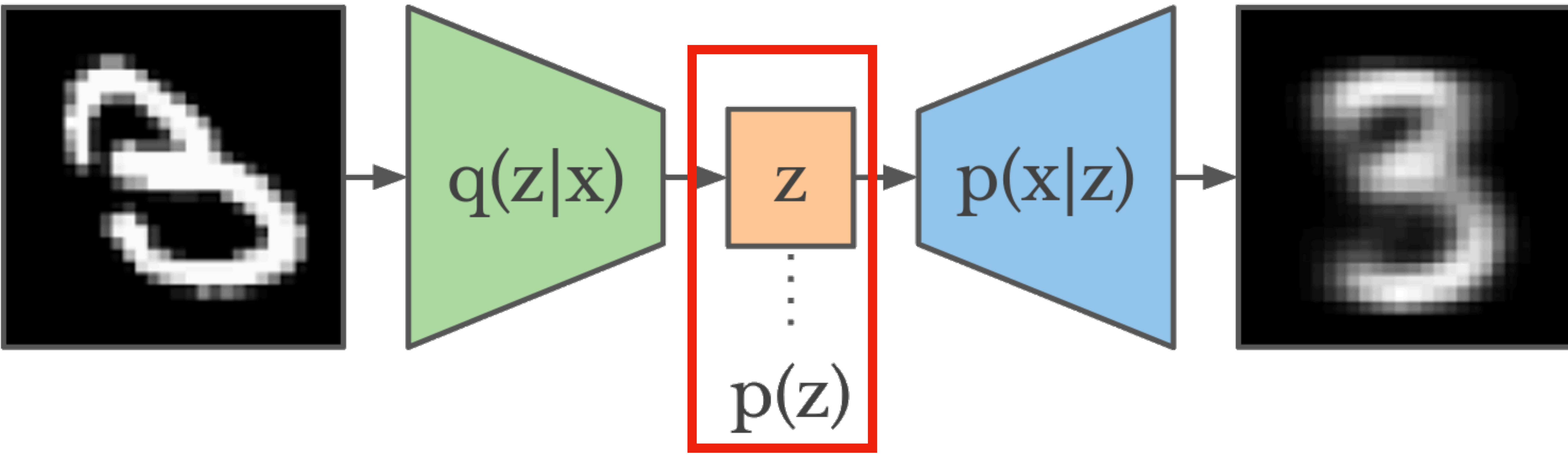
VAE

- Variational Autoencoder is a great way to model objects

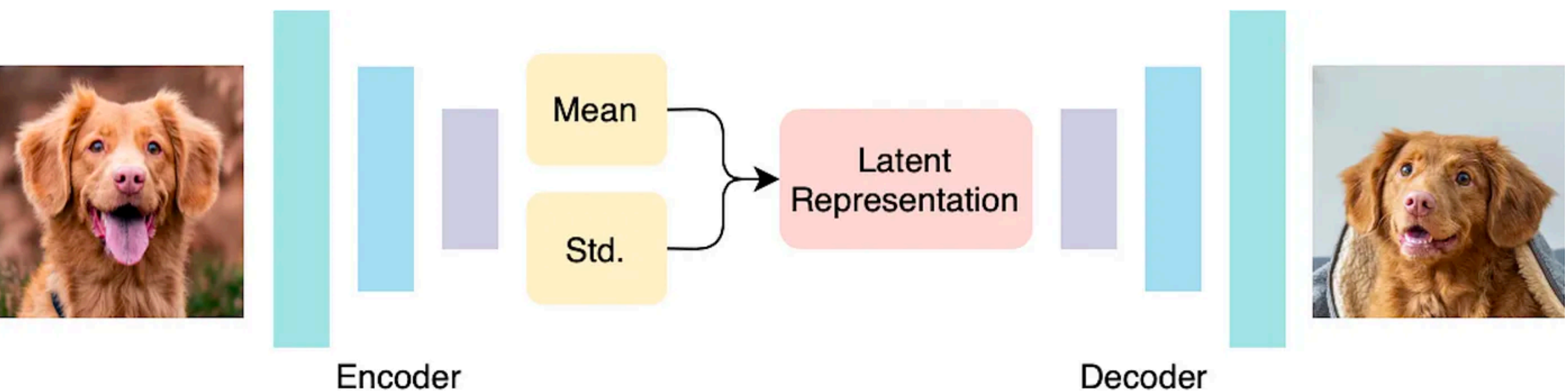


VAE

- Variational Autoencoder is a great way to model objects

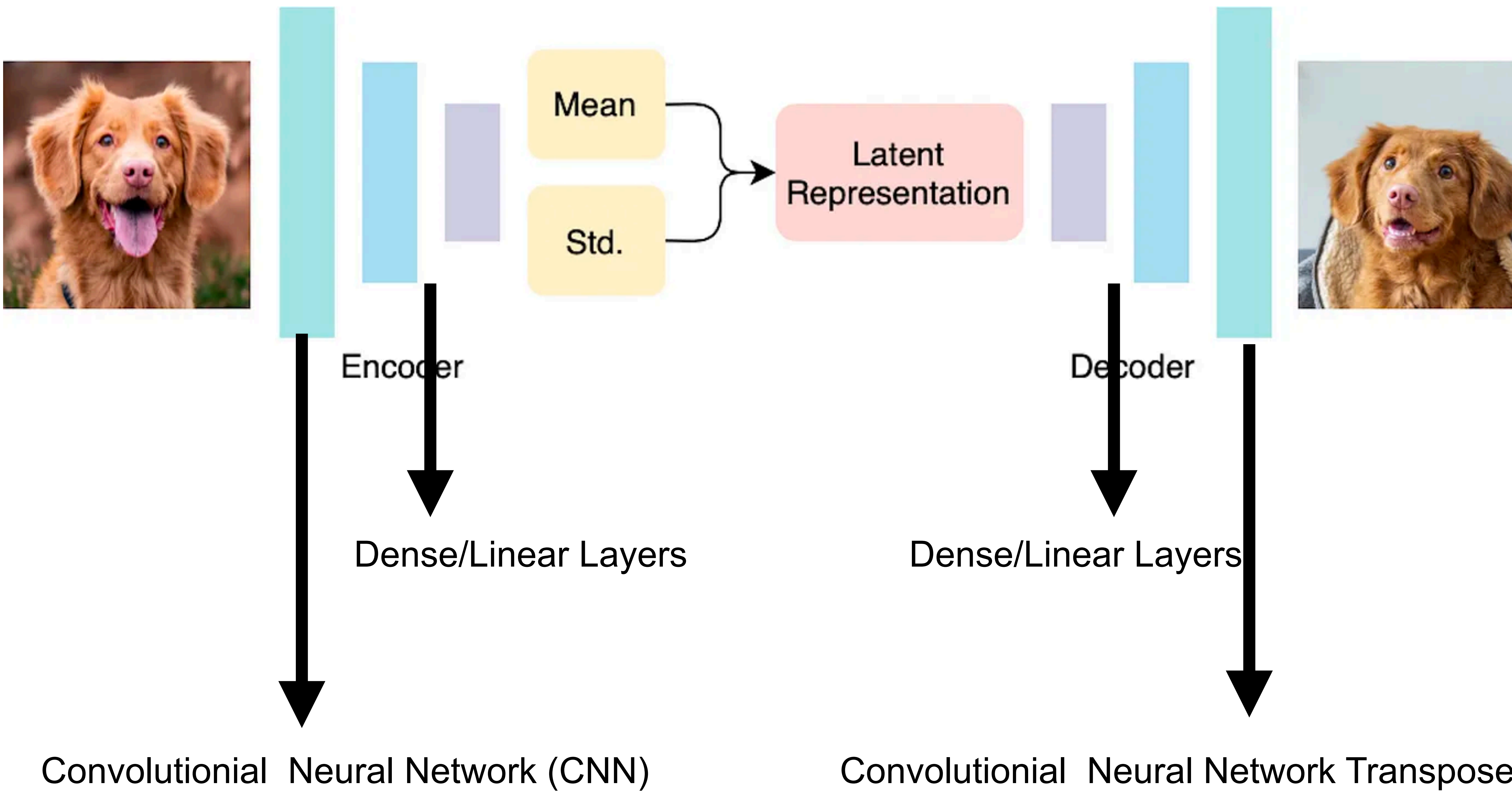


Randomly sample a normal distribution in this space

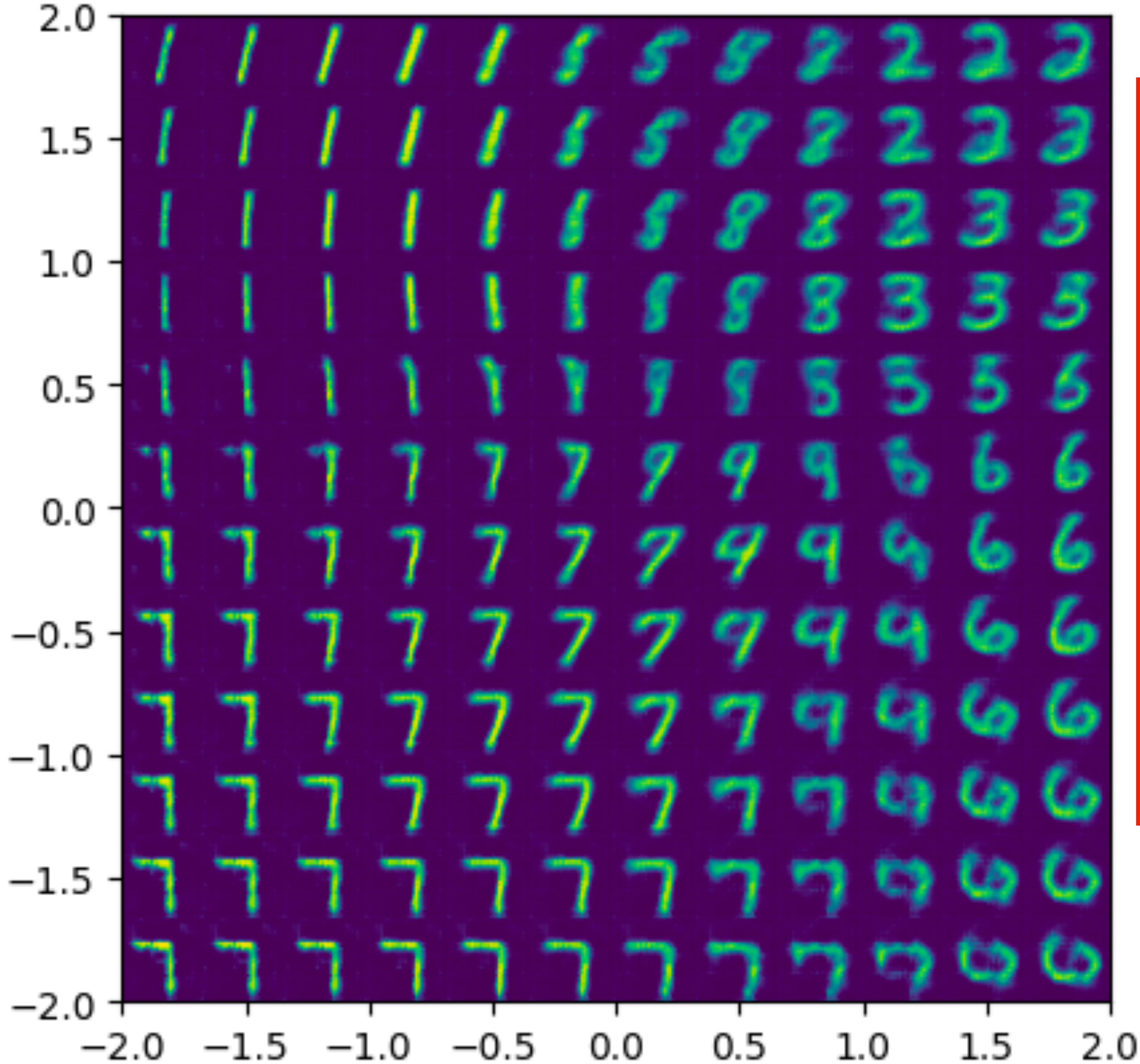


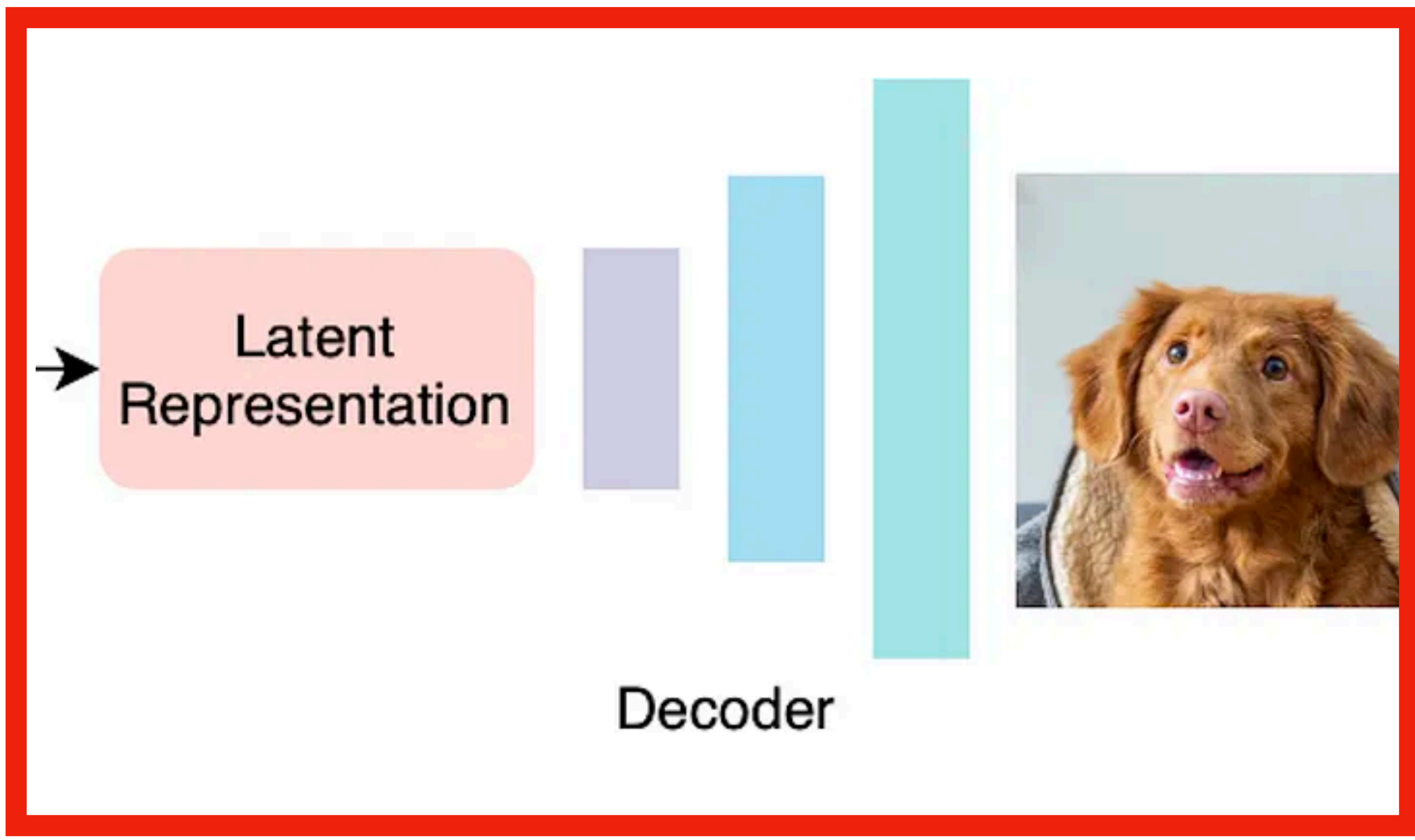
MNIST VAE encoder

- We will use a CNN to encode the data and process it



Exploring the latent space?

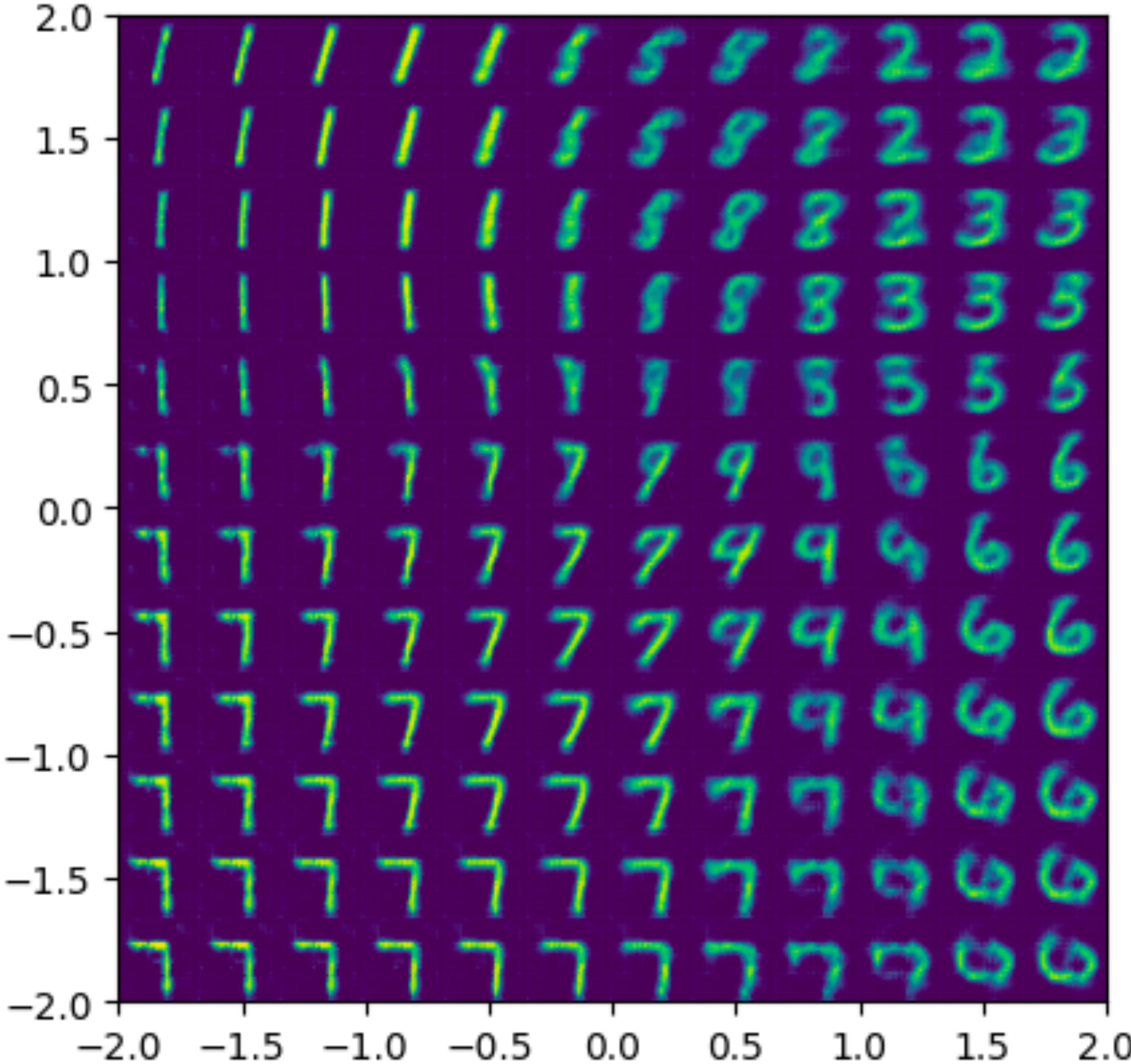


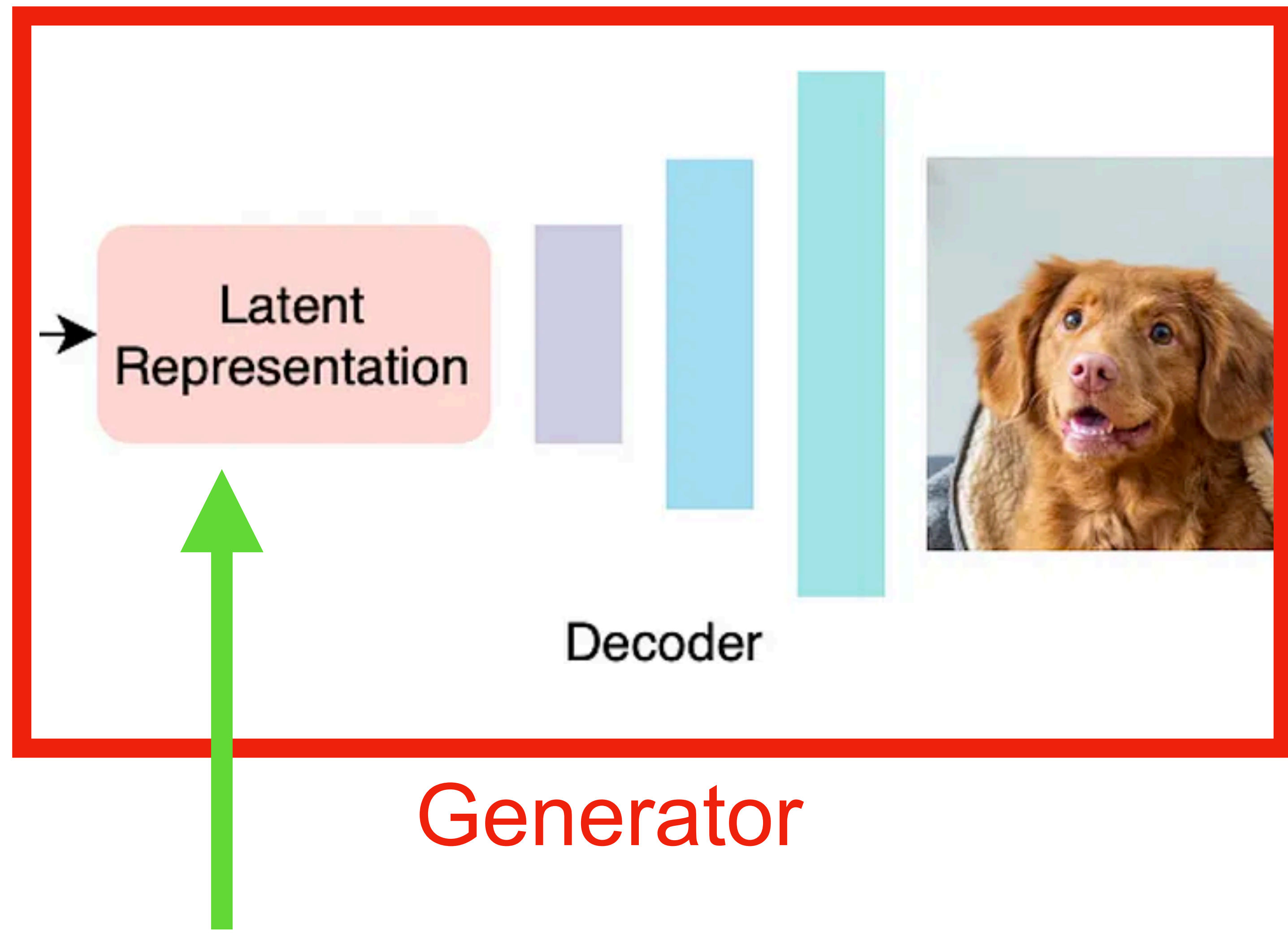


Generator

- We can sample the latent space as a generator

Conditional VAE



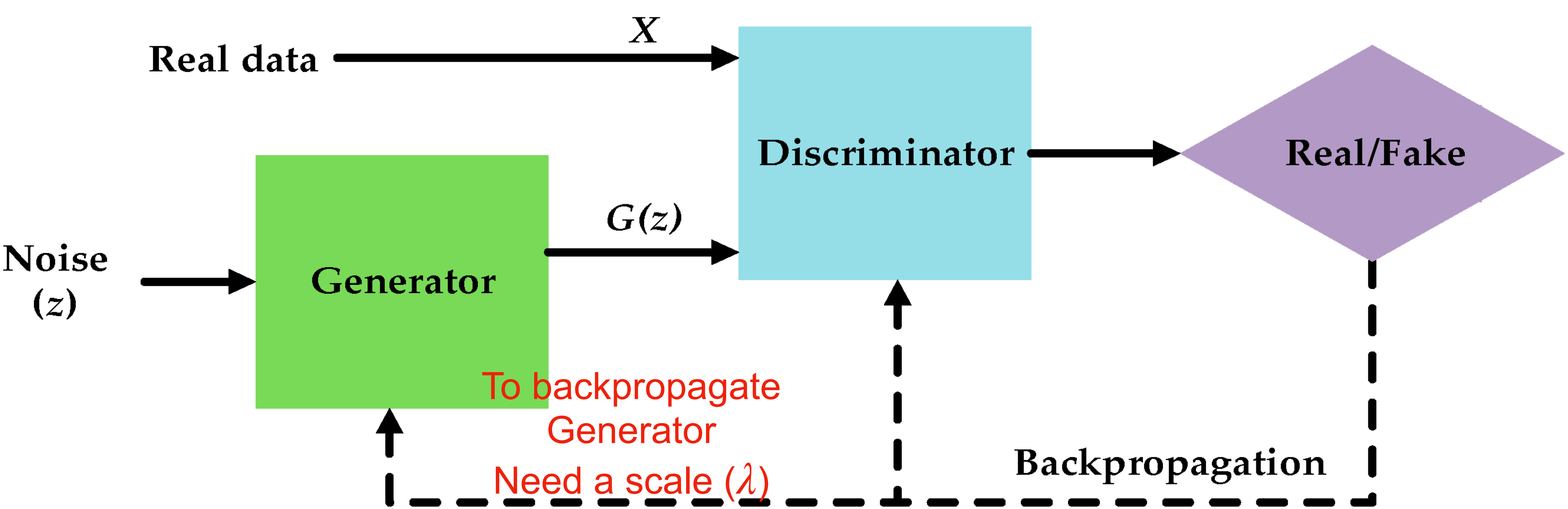


Variables that we would like to condition on

- Force known inputs into the VAE
  - That way our latent space has explicit knowledge of what is going on

GAN

- The other AI way of generating events uses
    - Generative Adversarial Networks (GANs)
    - I personally do not like these networks
        - They need too much tuning, **don't recommend them**



Power of GANs

- GANs became popular for generating fake human faces
  - Not as a hard a problem as typical physics problems
  - Our proton problem is probably already too hard for it

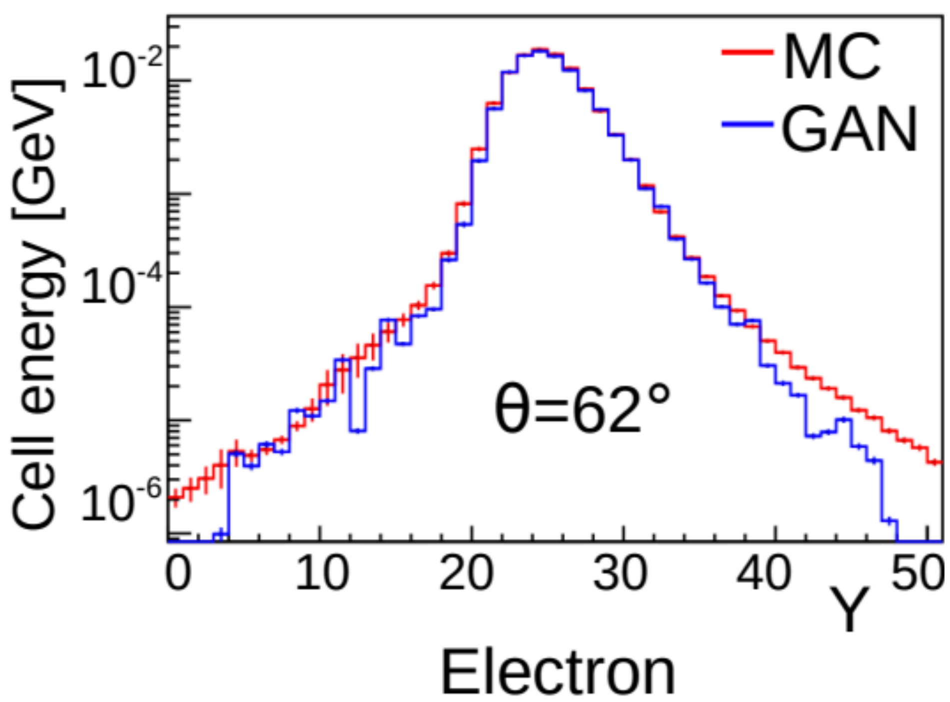
GAN

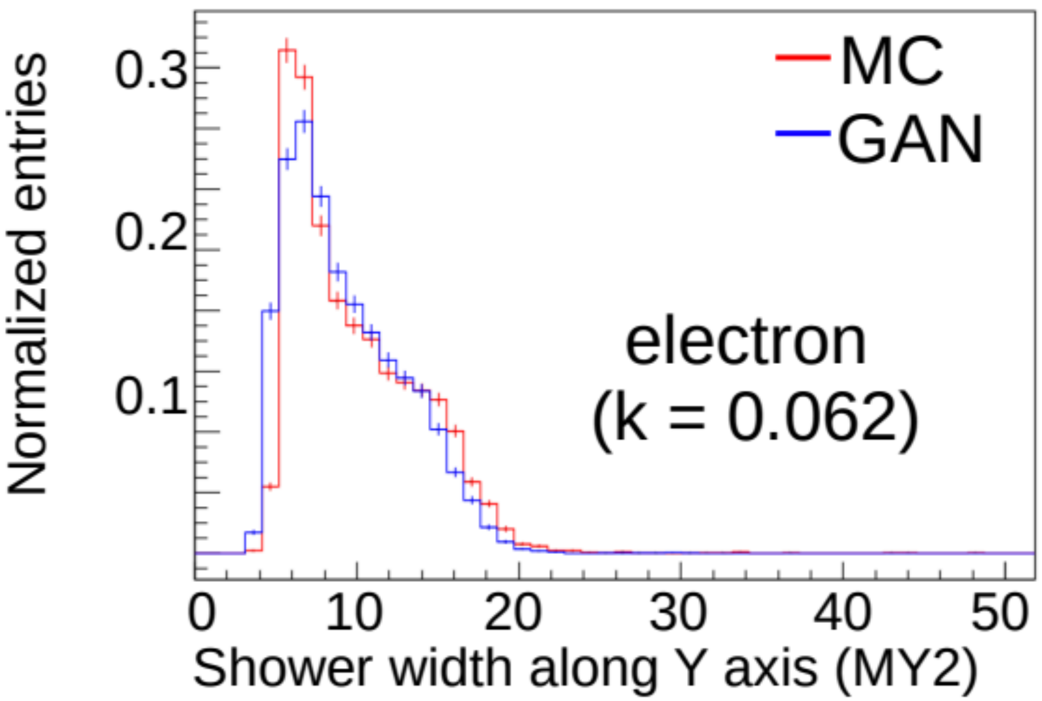
Autoencoder

NeRF

Weakness of GANs

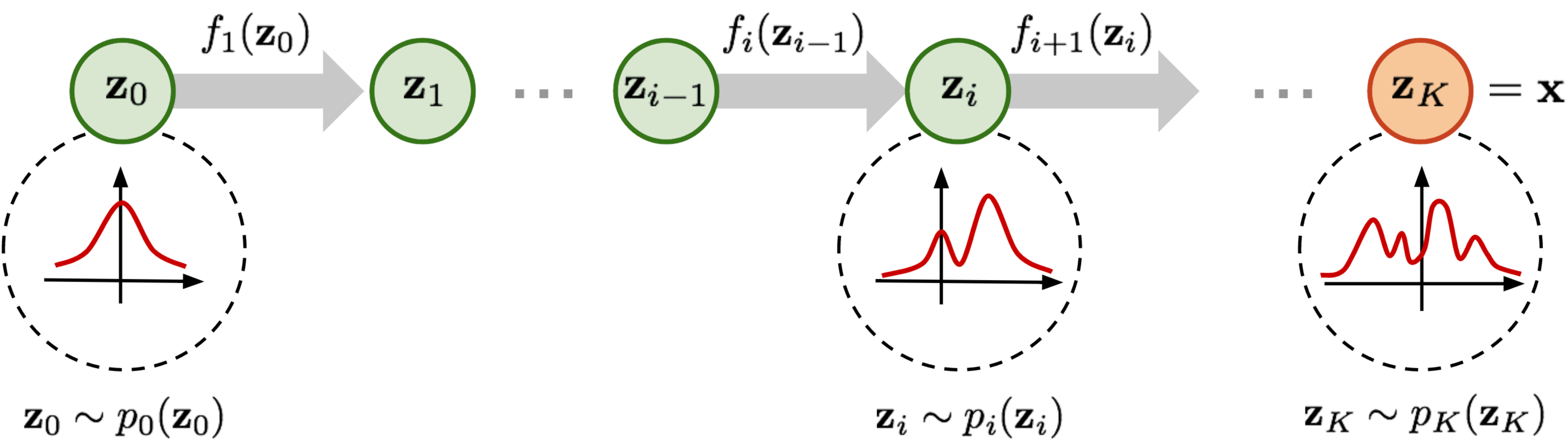
- GANs have not been great for physics
  - While they get the bulk of simulation ok
  - Struggle with the many orders of magnitude needed
  - Also they are pain to tune





Normalizing Flows

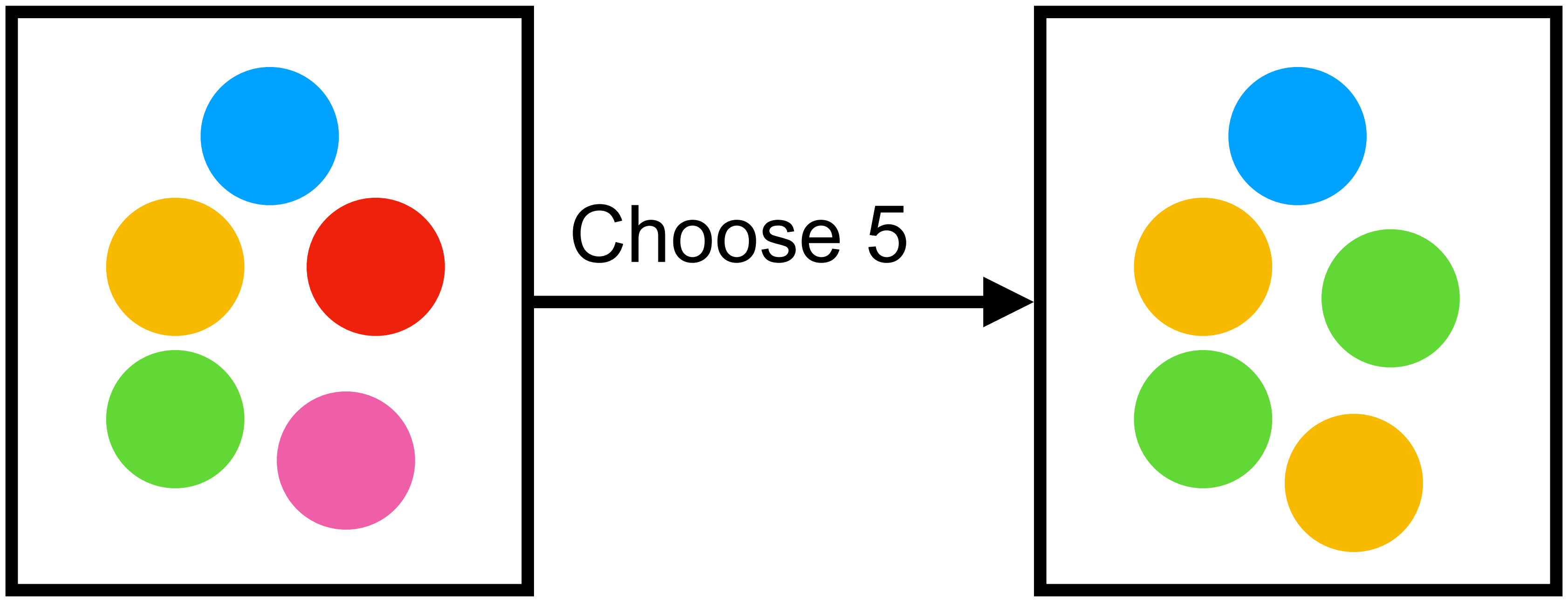
- While I don't show it here
  - The normalizing flow is next evolution of VAE
  - Allows for a more expressive latent space
  - Normalizing flow transforms flows to other spaces



Bootstrapping

- Lets say you have a fixed dataset ($\vec{x}$)
  - You cannot generate more
  - You compute something very complicated on it
    - For example $\vec{y} = \mathrm{NN}(\vec{x})$
  - You want to know the uncertainty on $\vec{y}$ given by $\sigma_{\vec{y}}$
    - However your function $\dfrac{d\mathrm{NN}(\vec{x})}{dx} \notin \text{exists}$

Bootstrapping: Strategy



You will
Get Duplicates

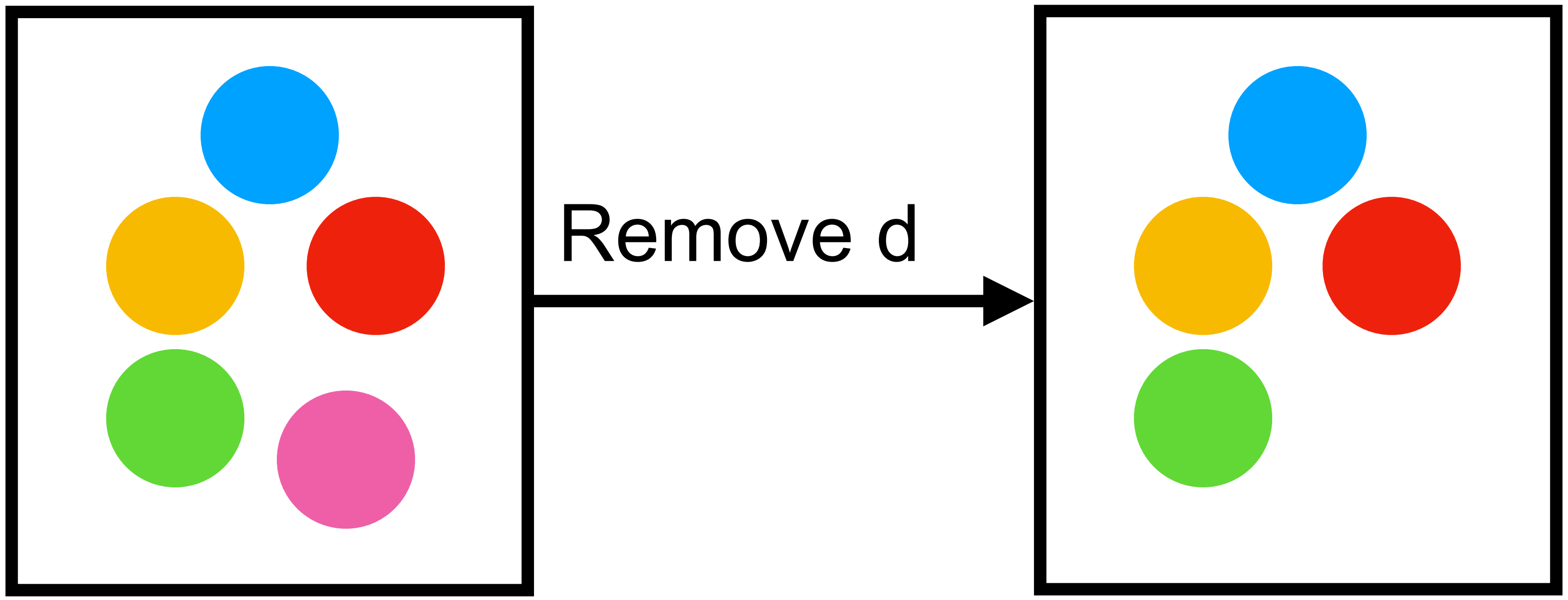
However if
things
Are truly
poission its ok

- Randomly sample the pool of events with the same size
  - Now use this sample to compute all our observables
- Variance over our random samples will be the uncertianty

$$\sigma_{\text{boot}} = \frac{1}{N_{\text{samples}} - 1} \boxed{\sum_{i}^{N_{\text{samples}}} \left(\mathcal{O} - \bar{\mathcal{O}}\right)^2}$$

We can do this
with any
observable

Delete-D Jackknife



- Like what we did with Bootstrapping
  - However, doesn't duplicate events

d=removed samples

$$\sigma_{\text{Jackknife}} = \frac{N_s - d}{\left({}_{N_s}C_d\right) d} \sum_{i}^{{}_{N_s}C_d} \left(\mathcal{O} - \bar{\mathcal{O}}\right)^2$$

Total Unc
From Jackknife

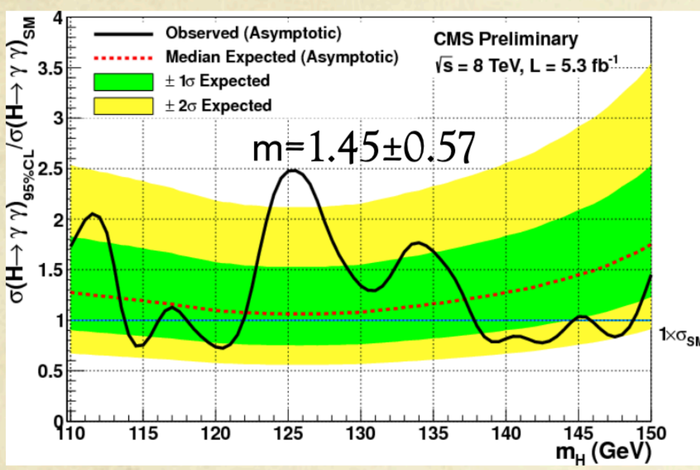
Fun Story

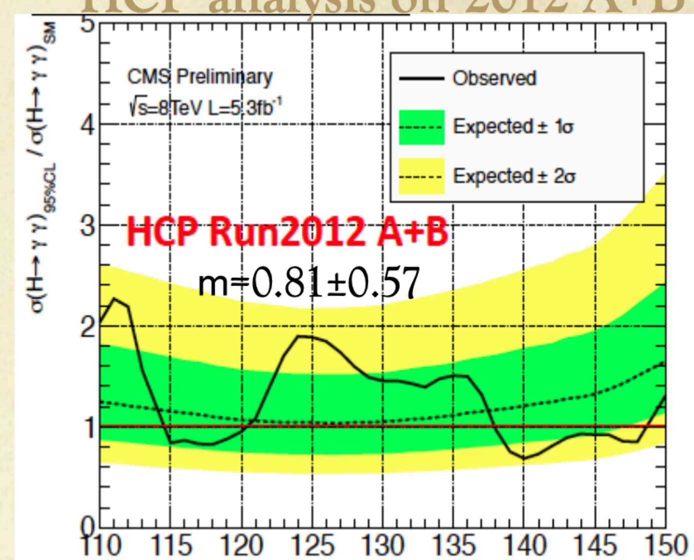
Higgs to diphoton result changed dramatically in 2012
On exactly the same data

The Observed Difference I

ICHEP analysis on 2012 A+B

HCP analysis on 2012 A+B



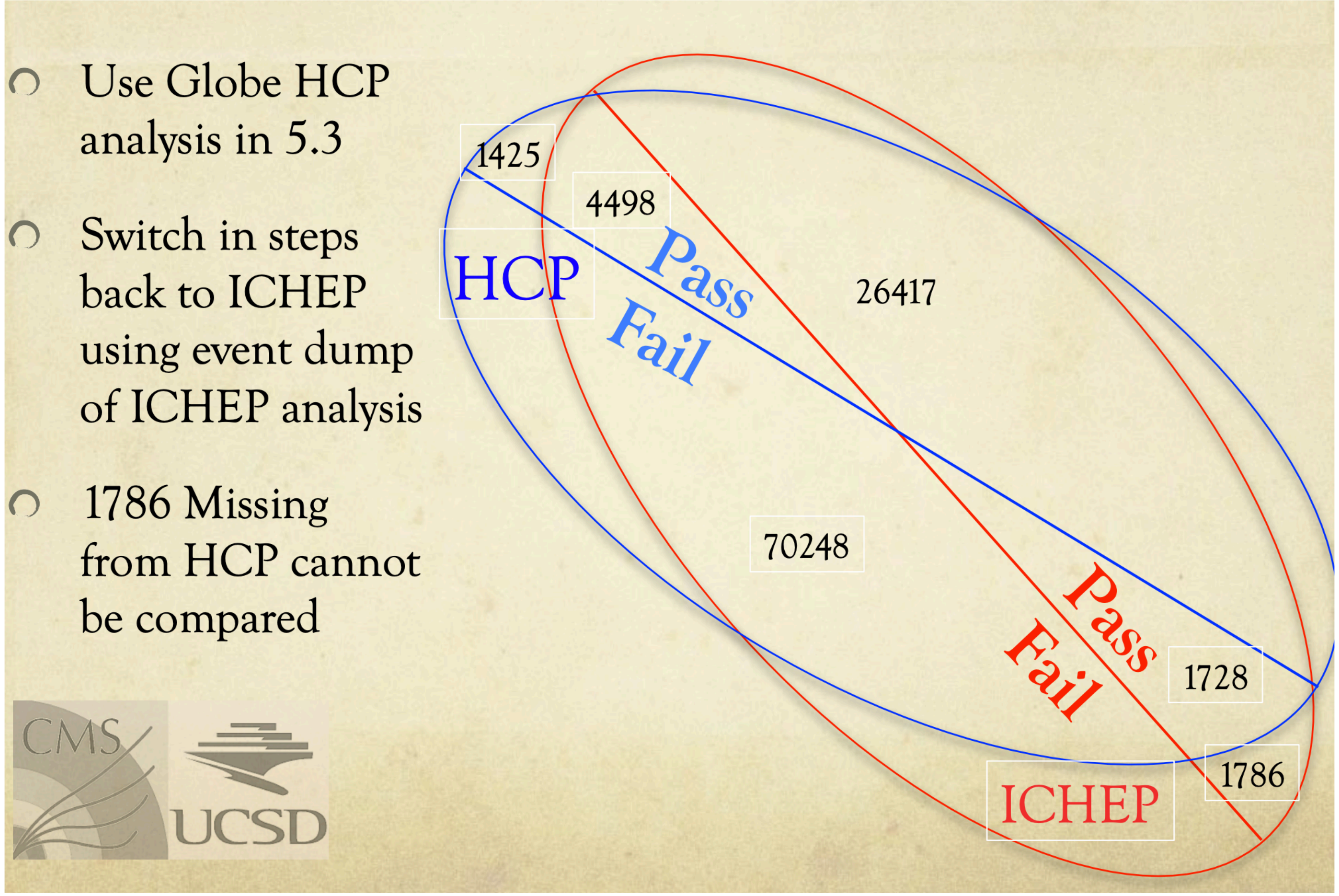


Difference = 0.79 σ_{diff}, but analysis and events are correlated

All checks concentrate on the difference between ICHEP and HCP Analyses on 2012 A+B.

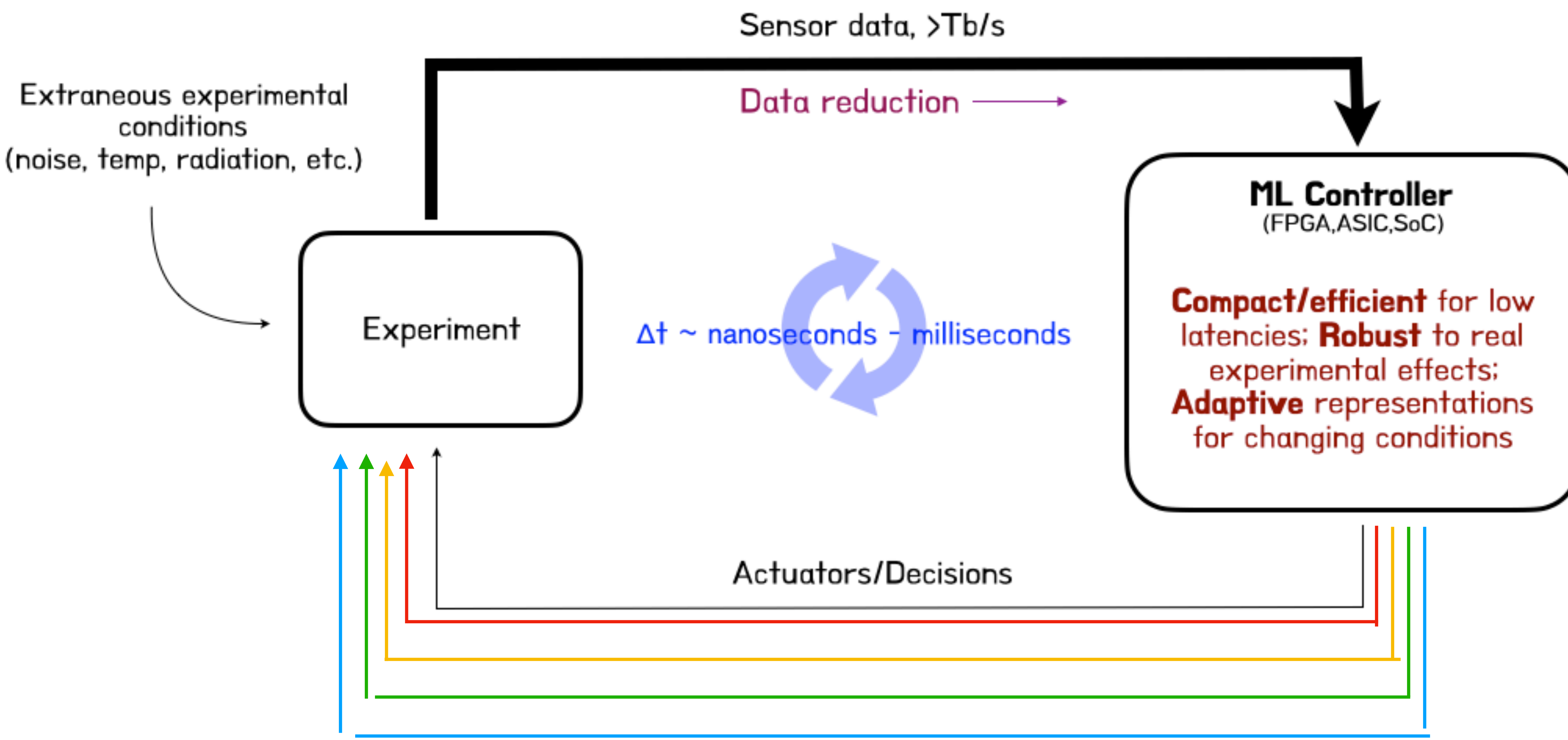
Fun Story

Higgs to diphoton result changed dramatically in 2012
On exactly the same data



Delete-d Jackkinfe was the way we were able to analyze the unc.

Title Text



Sensor data, >Tb/s
Data reduction
Extraneous experimental conditions (noise, temp, radiation, etc.)
Experiment
Δt ~ nanoseconds - milliseconds
ML Controller
(FPGA,ASIC,SoC)
Compact/efficient for low latencies; Robust to real experimental effects; Adaptive representations for changing conditions
Actuators/Decisions