E. Generative Modeling

E.1 Generative Modeling: Basic concepts

E.2 Generative Modeling: Variational models

E.3 Generative Modeling: Diffusion models

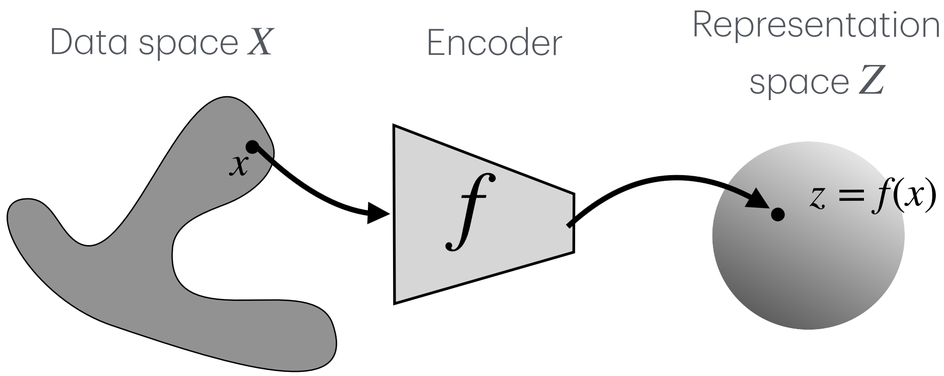
E.4 Generative Modeling: Flow models

Tiam Heydari, PhD
Postdoctoral Fellow, UBC

This lecture is Inspired by:
Generative Image Models and Representation Learning, chapter33. Antonio Torralba, Phillip Isola, and William Freeman

What is generative modeling?

Representation Learning



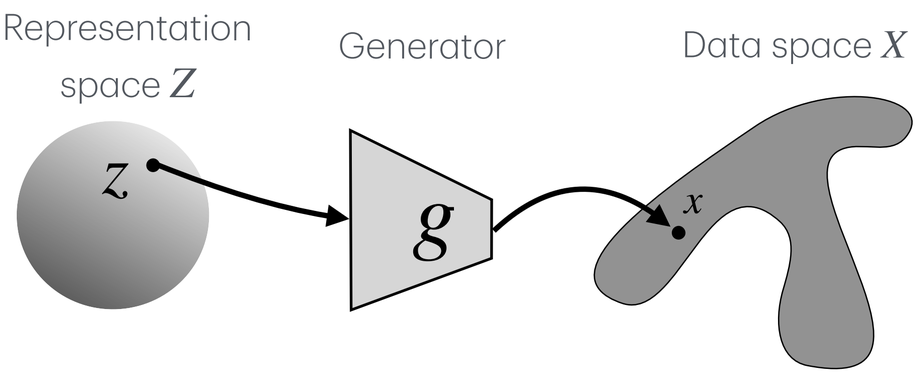
$$f : \mathcal{X} \to Z$$
$$z = f(x)$$

Question: Given data, what representation describes it?

Main direction: data → latent representation

Goal: compress, organize, compare, or predict from data.

Generative Modeling



$$g : Z \to \mathcal{X}$$
$$x = g(z)$$

Question: Given a latent code, what data could it produce?

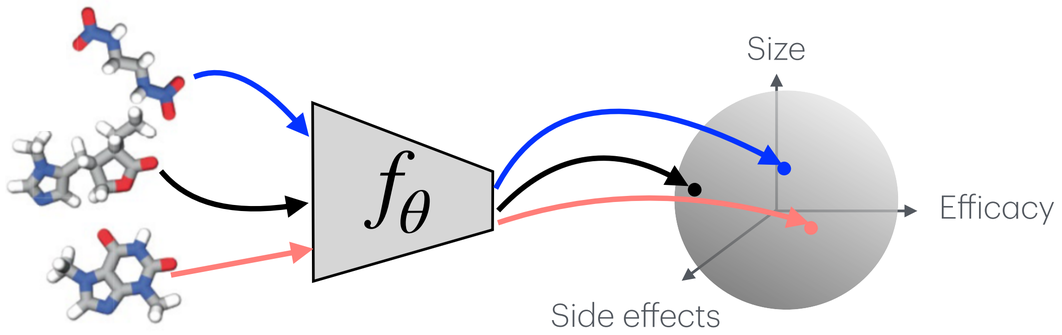
Main direction: latent representation → data

Goal: generate realistic new samples based on some variables

What is generative modeling?

Representation Learning

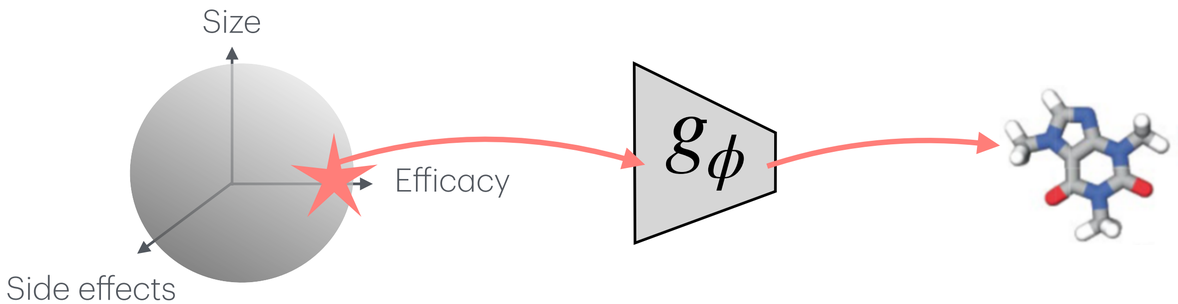
- For example: A successful Representation Learning for a drug molecular graphs we find a good disentangled latent space.



Now, if someone develops a new candidate structure x_{new}, we can use $f_\theta(x_{new})$ to compare it with other drugs in the database.

Generative Modeling

- For example: With a successful Generative Model for a drug molecular graphs we ask what is the drug structure with most efficacy and least side effects.



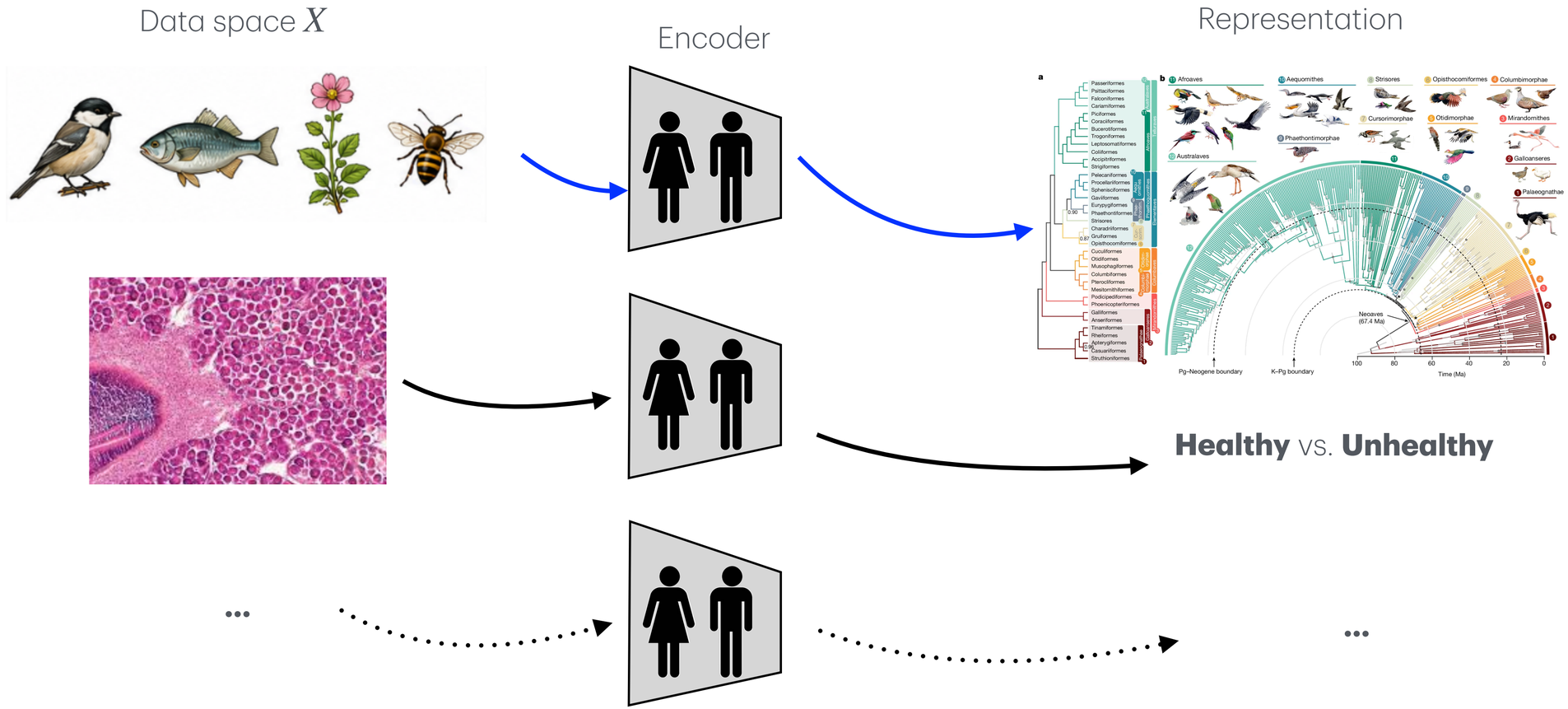
We just need to define which charachteristics in latent space we want z_{new}, and $x_{new} = g_\phi(z_{new})$ generates it *de novo!*

Wang, Hanchen, et al. "Scientific discovery in the age of artificial intelligence." *Nature* 620.7972 (2023): 47-60.

A historical intuition: biology as representation learning

Living systems have many components, dimensions and are complex.

Historically biology organizes observations into useful representations and categories*:



Mayr, E. (1982). The Growth of Biological Thought: Diversity, Evolution, and Inheritance

Brenner, S. (2010). Sequences and consequences.

*Many exceptions exist: Genetics, Later Developmental biology experiments, etc.

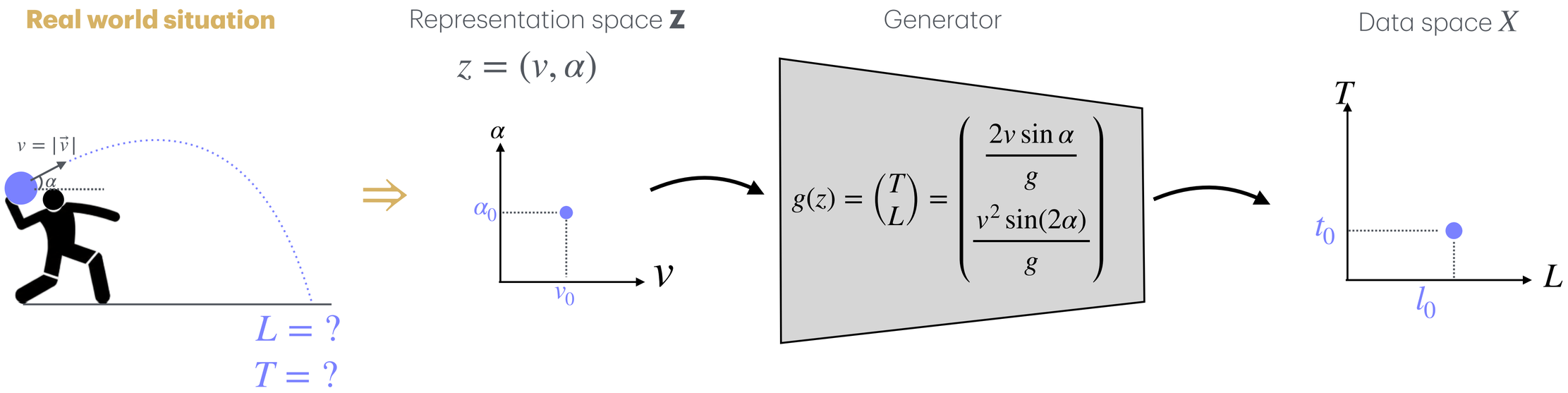
Images adapted from:

Stiller, J., Feng, S., Chowdhury, AA. *et al*. Complexity of avian evolution revealed by family-level genomes. *Nature* **629**, 851–860 (2024).

https://www.qu.edu.qa/en-us/Research/esc/Facilities/labs/Pages/histopathology.aspx

A physics case study: A historical intuition: physics as generative modeling

Physical systems can be complex, but physics often seeks compact laws. Historically, physics aims to obtain equations, symmetries, and initial conditions, then generates observable numerical predictions*.



In physics we compress the situation into a few variables, then uses laws to generate predictions.

*Caveat: this is an intuition, not a strict history. Physics also learns representations, and compact laws do not automatically explain higher-level complexity. The literature of emergence is vast and deep in physics. Eg. P. W. “More Is Different.”(1972)

A physics case study:

For pedagogy, let us take the projectile example more seriously! The ideal physics model says:

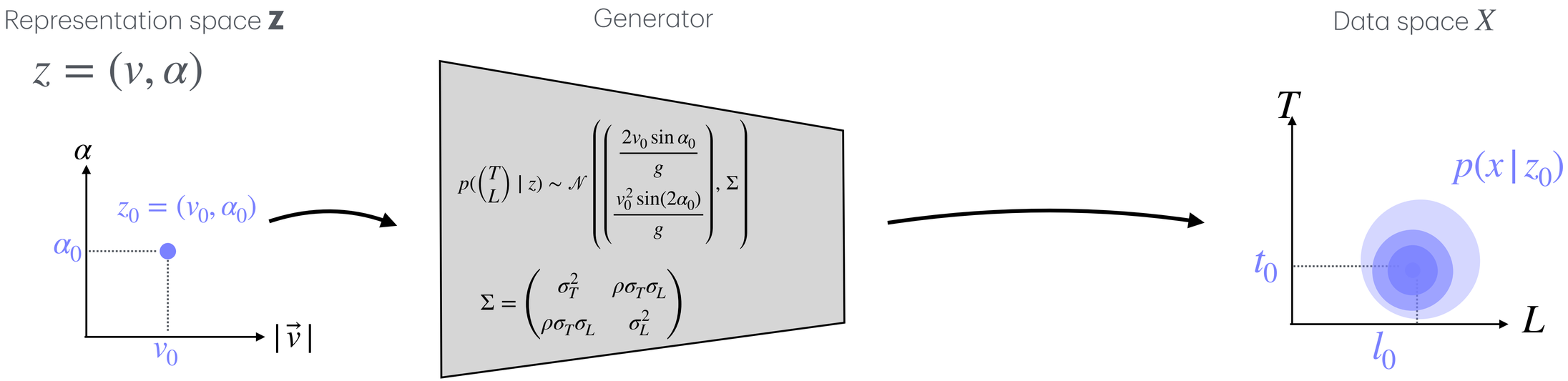
$$\begin{pmatrix} T \\ L \end{pmatrix} = g(z), \qquad z = (v, \alpha)$$

But in the real world, the same apparent input may not always give the same output. It is very common case when we try to convert theoretical physics into observational data.

Idea 1: hidden/unmodeled variables:

Wind, friction, spin, air density, measurement error. Therefore the outputs L and T have uncertainty (randomness) in them, and instead of a single value, we need to use probability density to represent them:

$$\begin{pmatrix} T \\ L \end{pmatrix} = g(z) \rightarrow \sim p(\begin{pmatrix} T \\ L \end{pmatrix} \mid z)$$



A physics case study:

Idea 2: Idea 2: We may not know the input exactly, or might want to study a variety of input values.

Even if the law is fully deterministic, we may not know the input exactly. Instead of one point, $z_0 = (v_0, \alpha_0)$, we may only know a distribution of input values:

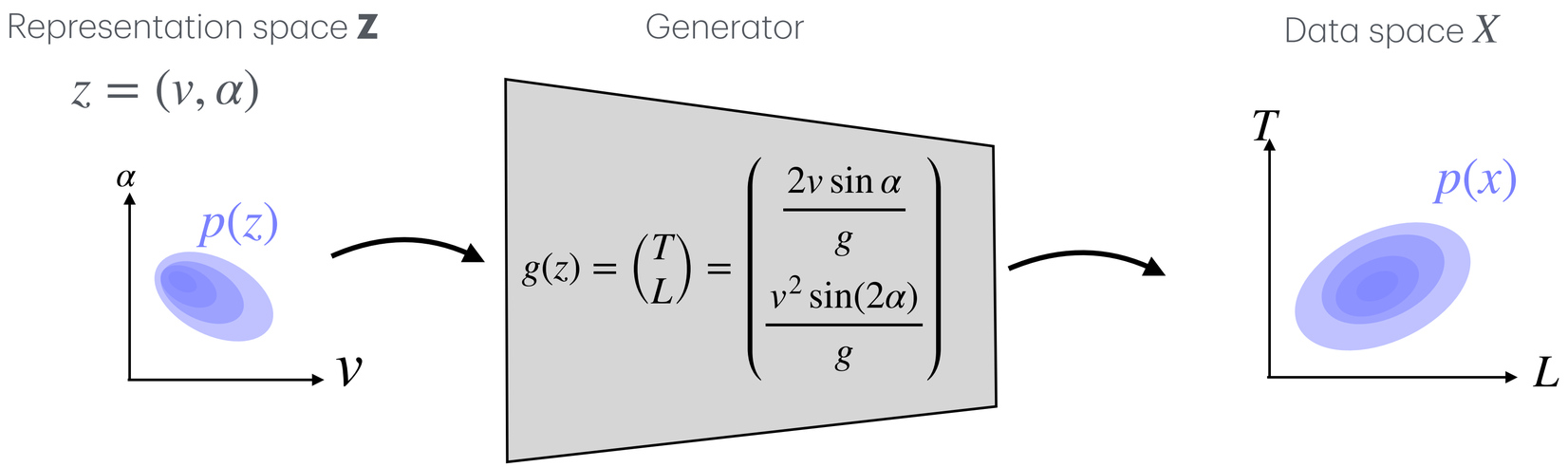
$$z = (v, \alpha) \sim p(z)$$

and then the deterministic map:

$$x = g(z), \qquad x = \begin{pmatrix} T \\ L \end{pmatrix}$$

turns that into an output distribution point by point, even though that the generator is deterministic:

$$z \to p(z) \quad \Longrightarrow \quad x \to p(x)$$



A physics case study: Combining both ideas: from deterministic maps to probabilistic generation

In reality, both the input and the environment may be uncertain:

(i) **Generation**: Uncertainty and probabilistic outputs for each single point input, $\delta(z = z_0)$ (probabilistic generation):

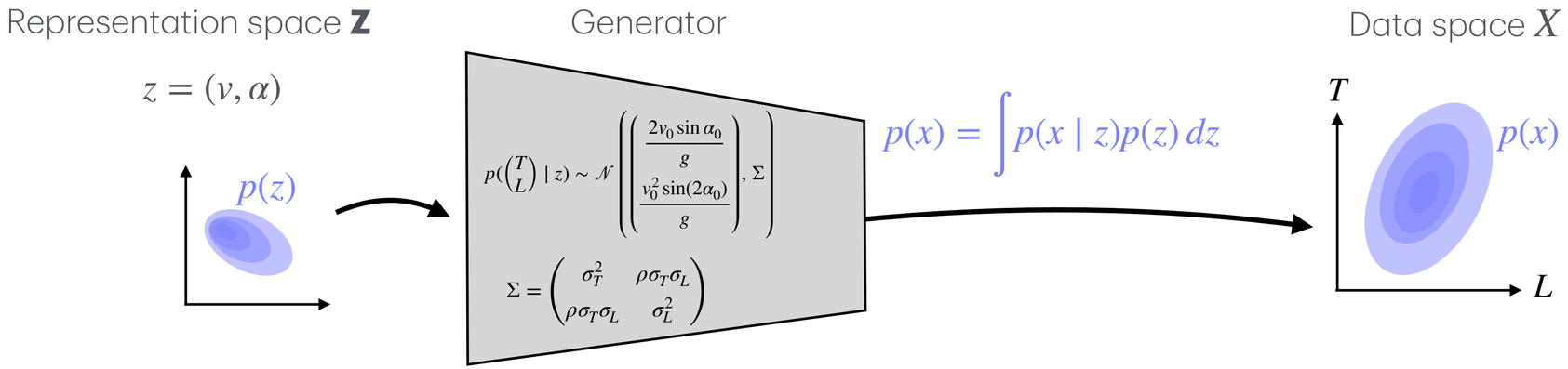
$$p(x) = \int p(x \mid z)\delta(z - z_0)\, dz \quad = p(x \mid z_0)$$

(ii) **Input**: Probabilistic view of the representation z (the input of the generator): $p(z)$

Combining (i) and (ii) leads to how the output $p(x)$ is shaped based on input representation distribution $p(z)$ and generator uncertainty $p(x \mid z)$:

$$p(x) = \int p(x \mid z)p(z)\, dz$$

***Note**: When $p(z)$ and $p(x \mid z)$ are known,$p(x)$ can be estimated by numerical integration or sampling.*



A physics case study: Combining both ideas: from deterministic maps to probabilistic generation

Let's implement this in practice:

Abstractly, we followed this direction: $p(z) \longrightarrow p(x \mid z) \longrightarrow p(x)$

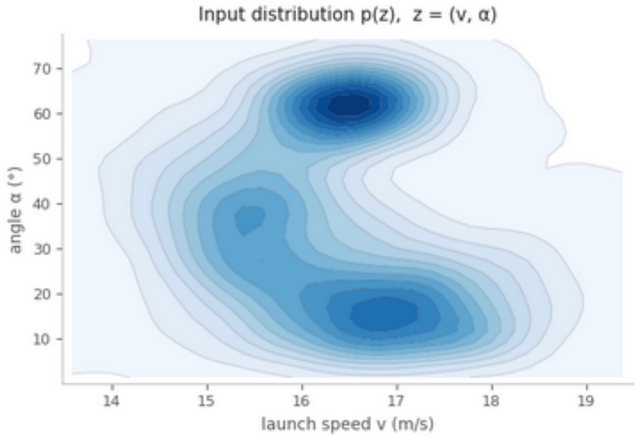
This should be given,
Or we should be able to estimate it!
We will discuss this further

This is our generator, we derived it.
Maps one input point (z) into a probability $p(x \mid z)$

This is driven by knowing the $p(z)$ and knowing the generator:

$$p(x) = \int p(x \mid z)p(z)\, dz$$

Representation space **z**

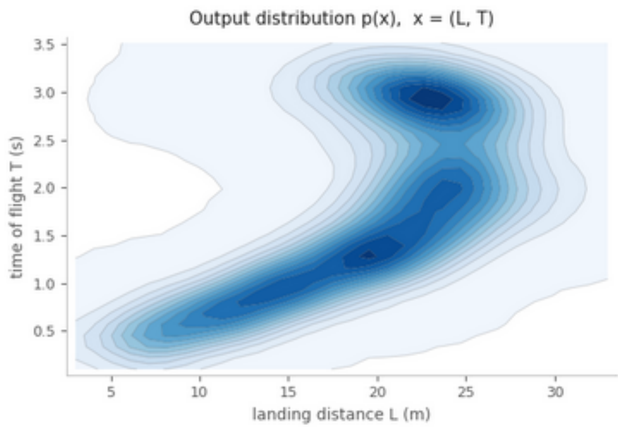


Generator

$$p\left(\begin{pmatrix} T \\ L \end{pmatrix} \mid z\right) \sim \mathcal{N}\left(\begin{pmatrix} \frac{2v_0 \sin\alpha_0}{g} \\ \frac{v_0^2 \sin(2\alpha_0)}{g} \end{pmatrix}, \Sigma\right)$$

$$\Sigma = \begin{pmatrix} \sigma_T^2 & \rho\sigma_T\sigma_L \\ \rho\sigma_T\sigma_L & \sigma_L^2 \end{pmatrix}$$

Data space X

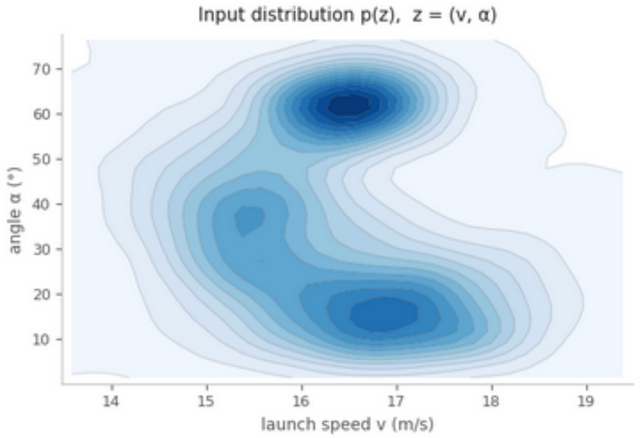


But how did we actually dod it? We did it by sampling actual points from $p(z)$.

Let's explore this next and via this we will introduce some other important concepts of **Generative modelling**.

A physics case study: How did we approximate p(x)?

Representation space **z**



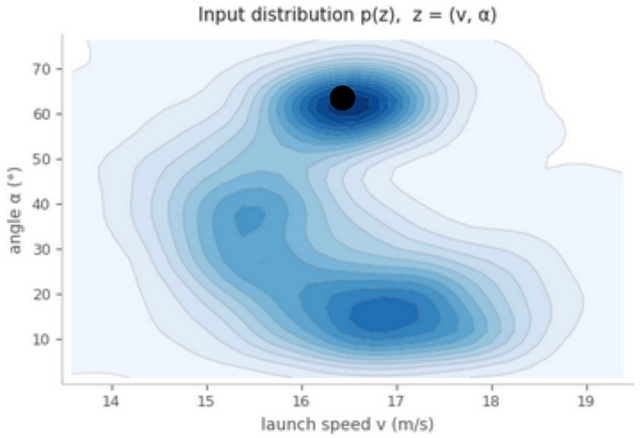
Step 1) Choose an input distribution:

A physics case study: How did we approximate p(x)?

Representation space **z**

Generator

Data space X



Step 1) Choose an input distribution:

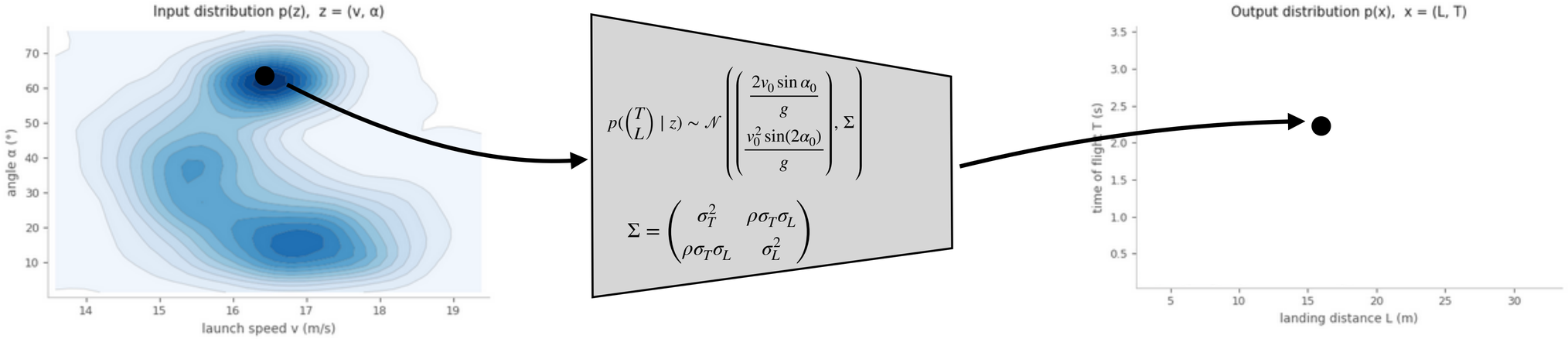
Step 2) Sample one point from $p(z)$

A physics case study: How did we approximate p(x)?

Representation space **z**

Generator

Data space X



Step 1) Choose an input distribution:

Step 2) Sample one point from $p(z)$

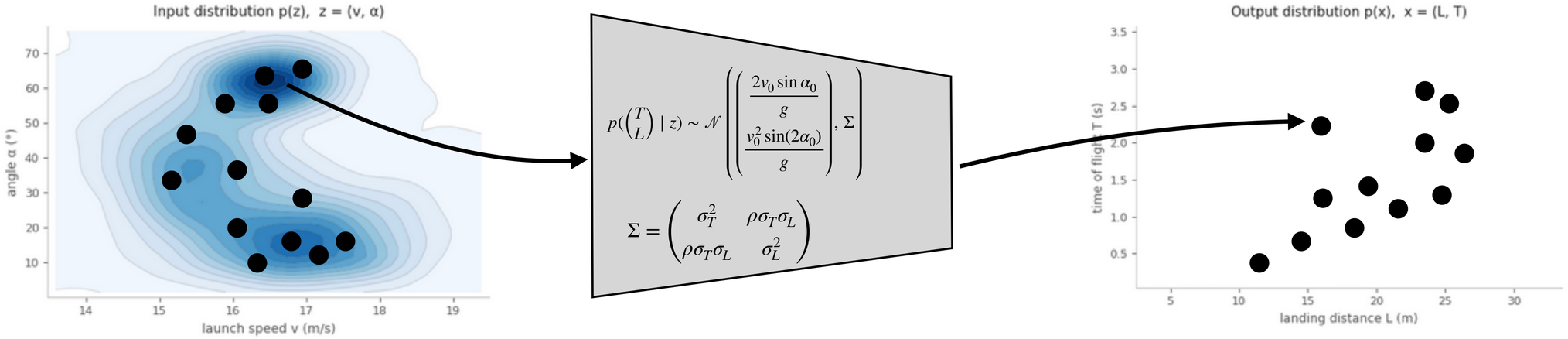
Step 3) Give that point to the generator. For this input, the generator produces one output sample: $x_i \sim p(x \mid z_i)$

A physics case study: How did we approximate p(x)?

Representation space **z**

Generator

Data space X



Step 1) Choose an input distribution:

Step 2) Sample one point from $p(z)$

Step 3) Give that point to the generator. For this input, the generator produces one output sample: $x_i \sim p(x \mid z_i)$

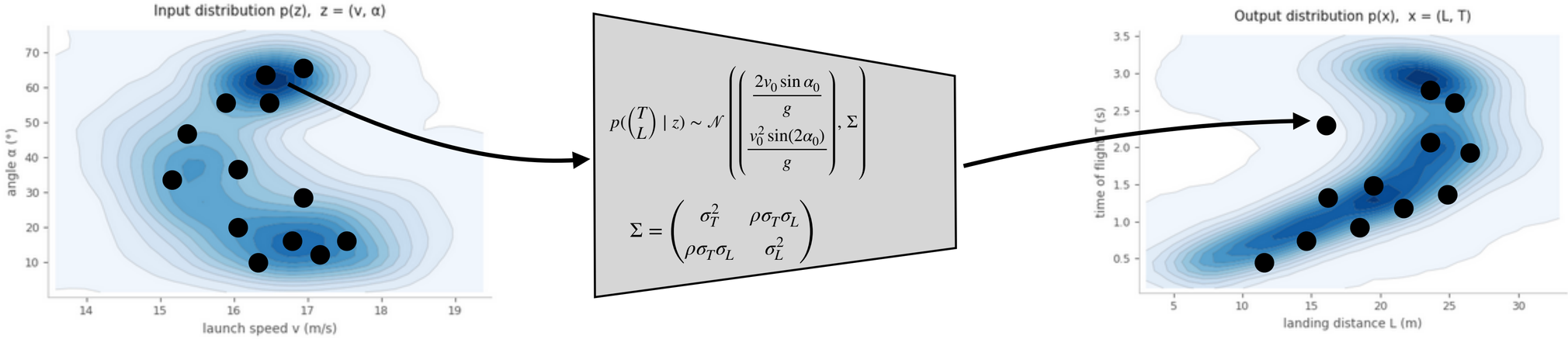
Step 4) Repeat many times

A physics case study: How did we approximate p(x)?

Representation space **z**

Generator

Data space X



Step 1) Choose an input distribution:

Step 2) Sample one point from $p(z)$

Step 3) Give that point to the generator. For this input, the generator produces one output sample: $x_i \sim p(x \mid z_i)$

Step 4) Repeat many times

Step 5) Estimate the output distribution: $\{x_i\}_{i=1}^{N}$ approximates $p(x)$

A physics case study: How did we approximate p(x)?

We want to compute: $p(x)$
But instead of solving the integral directly, we use sampling.

Step 1) Choose an input distribution: First, we defined a probability density in representation space: $z = (v, \alpha), \qquad z \sim p(z)$
This means some launch conditions are more likely than others, and we knew it's exact form

Step 2) Sample one point from $p(z)$: We randomly choose one input point: $z_i \sim p(z)$
Points in high-density regions of $p(z)$ are more likely to be selected.

Step 3) Give that point to the generator. For this input, the generator produces one output sample: $x_i \sim p(x \mid z_i)$
The sample still has randomness because of hidden variability: wind, friction, spin, measurement noise, etc.

Step 4) Repeat many times:

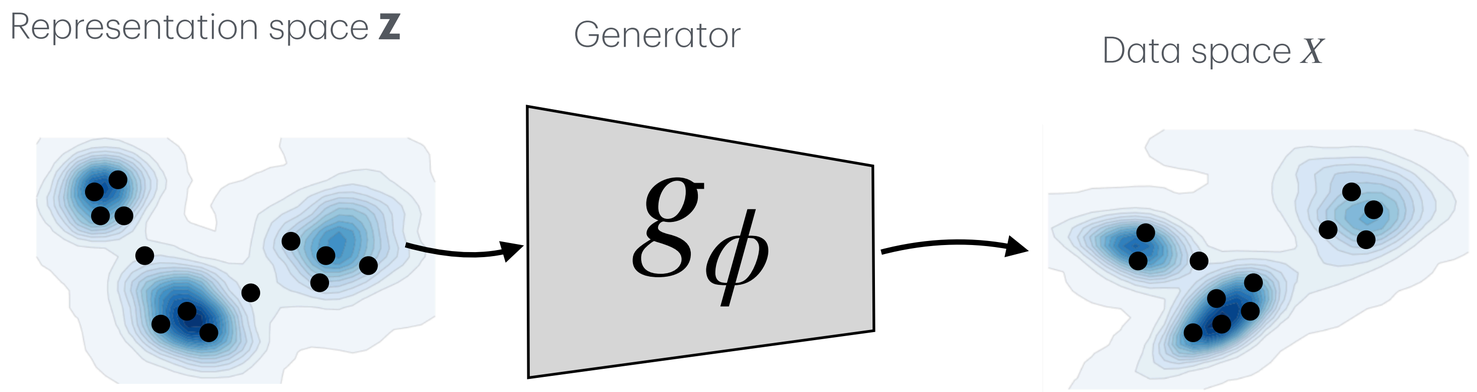
$$z_1, z_2, \ldots, z_N \sim p(z) \qquad \rightarrow \qquad x_1, x_2, \ldots, x_N, \qquad x_i \sim p(x \mid z_i)$$

These $\{x_i\}$ are samples from the output distribution $p(x)$.

Step 5) Step 5) Estimate the output distribution After many repetitions: $z_i \sim p(z), \qquad x_i \sim p(x \mid z_i),$
the generated samples are samples from the output distribution $p(x)$: $\{x_i\}_{i=1}^{N} \approx p(x)$
In another word, we can approximate $p(x) = \int p(x \mid z)p(z)\, dz$ by generating samples from $p(x)$

This is called Monte Carlo (MC) approximation, we can approximate distributions by repeated sampling.

A physics case study: The generator function can be any function g_ϕ.



Step 1) If the input distribution in the representation space $p(z)$ is known or give

Step 2) Sample one point from $p(z)$

Step 3) Give that point to the generator. For this input, the generator produces one output sample: $x_i \sim p(x \mid z_i)$

Step 4) Repeat many times

Step 5) Estimate the output distribution: $\{x_i\}_{i=1}^{N}$ approximates $p(x)$

This is a general recipe for if we know $p(z)$, and g_ϕ already, which is not the case when we move out of the domain of physics. Generative AI aims to partially address this as we will learn. Before that, we will review some basic probability language.

Background: probability

A) Probability theory: Basics

Random variables, samples, and densities:

Random variable:

$X \in \mathcal{X}$

$X \sim p(x)$: Here $p(x)$ describes how likely different values of X are.

$X_i =$ the random variable before we observe it

$x_i =$ the actual observed value after sampling

$\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$ is a dataset

Mass vs density:

For discrete data:

$p(x) = P(X = x)$ probability

$$\sum_{x \in \mathcal{X}} p(x) = 1$$

For continuous data:

$p(x) \geq 0$ probability density,

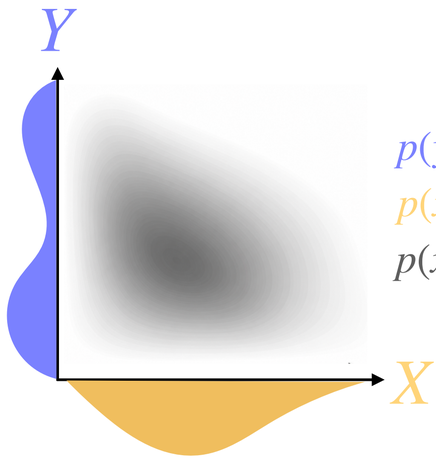
$$\int_{\mathcal{X}} p(x)\, dx = 1$$

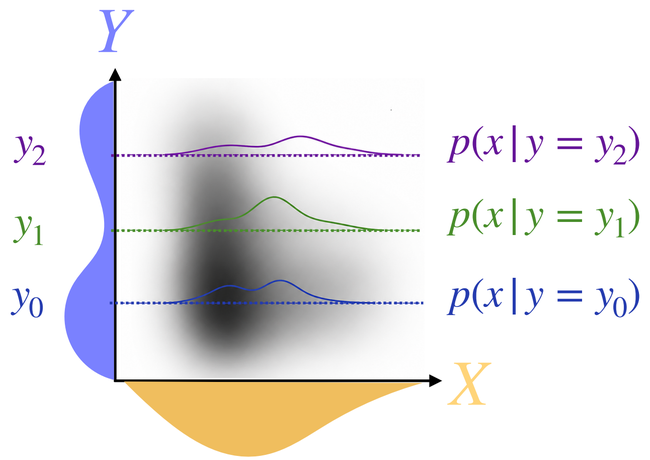
For a region A: $P(X \in A) = \int_A p(x)\, dx$: probability

A) Probability theory: Conditional, joint, and marginal distributions

Let X and Y be two random variables. Their observed values are:

$$x \in \mathcal{X}, \qquad y \in \mathcal{Y}$$





Marginal distributions: Describe each variable alone:

Marginal distribution of X: $p(x)$

Marginal distribution of Y: $p(y)$

Joint distribution: Describes both variables together:

Jointly seeing $X = x$ and $Y = y$: $p(x, y)$

Conditional distribution: describes one variable when the other's value is known:

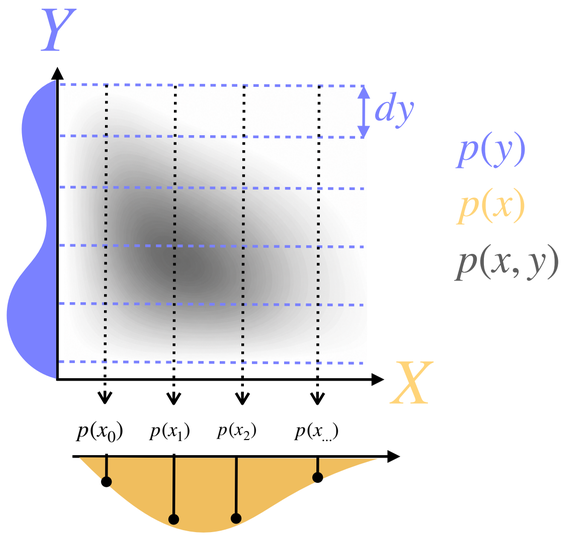
probability of seeing $X = x$, when we know $Y = y$: $p(x \mid y)$

Product rule: The joint can be written as:

$$p(x, y) = p(x \mid y)p(y) = p(y \mid x)p(x)$$

Marginalization: To get $p(x)$, we sum/integrate over all possible y:

$$p(x) = \int p(x, y)\, dy \qquad \text{or} \rightarrow p(x) = \int p(x \mid y)p(y)\, dy$$



$$p(x) = \int p(x, y)\, dy$$

B) Probability application: learning a density model from “data points”

Data: Lets assume we have observed some data points:

$$\mathcal{D} = \{x_1, \ldots, x_N\}$$

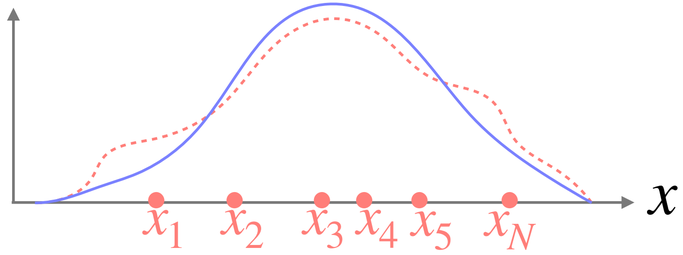
We assume these are coming from a distribution that we don’t have access to:

$x_i \sim p_{data}(x)$, and p_{data} is unknown to us

Where should we start?

p_{data} : Unknown

p_θ : We want to estimate from $\{x_i\}$



Model: We can try to build a *parametrized model* p_θ with limited number of parameters θ to model p_{data}:

$$p_\theta(x) \approx p_{data}(x)$$

So we want these two be similar, but how can we define similarity of distributions?

Distance of distributions: If $p_1(x)$ and $p_2(x)$ be two probability distributions over the same domain $x \in \mathcal{X}$, KL divergence measures their distance:

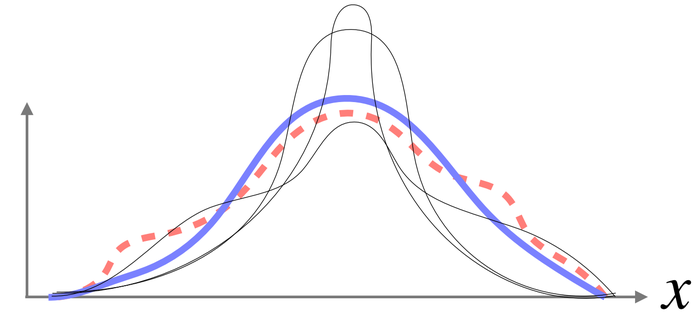
$$D_{\mathrm{KL}}(p_1 \,\|\, p_2) = \int_{\mathcal{X}} p_1(x) \log \frac{p_1(x)}{p_2(x)}\, dx \geq 0$$

Our objective: We to find model parameter θ that makes the model distribution close to the data distribution:

$$\theta^* = \arg\min_\theta D_{\mathrm{KL}}\left(p_{data} \,\|\, p_\theta\right)$$

p_θ for optimal θ^* that minimizes $D_{\mathrm{KL}}\left(p_{data} \,\|\, p_\theta\right)$

p_θ for non optimal θ



The problem is p_{data} is not known to be used in the D_{KL} formula!

B) Probability application: learning a density model from "data points"

Our objective: $\theta^* = \arg\min_\theta D_{\mathrm{KL}}\left(p_{data} \,\|\, p_\theta\right)$. We don't have the p_{data}, we have $\{x_i\}$.

How to solve it?

Step1: Open up the KL:

$$D_{\mathrm{KL}}\left(p_{data}\|p_\theta\right) = \int p_{data}(x)\log\frac{p_{data}(x)}{p_\theta(x)}dx = \int p_{data}(x)\log p_{data}(x)dx - \int p_{data}(x)\log p_\theta(x)dx$$

Step2: The first term does not depend on θ:

$$\theta^* = \arg\min_\theta D_{\mathrm{KL}}\left(p_{data}\|p_\theta\right) = \arg\max_\theta \int p_{data}(x)\log p_\theta(x)dx$$

Step3: Replace the integral with sample average (remember MC):

$$\theta^* = \arg\max_\theta \int p_{data}(x)\log p_\theta(x)dx \;\; = \arg\max_\theta \mathbb{E}_{x\sim p_{data}}[\log p_\theta(x)] \approx \arg\max_\theta \frac{1}{N}\sum_{i=1}^{N}\log p_\theta(x_i)$$

Now we can conclude it for: A given set of data points $\{x_i\}$ + A parameterized function $p_\theta(x)$

Note: if we define the likelihood of the dataset as: $\mathscr{L}(\theta) = \prod_{i=1}^{N} p_\theta(x_i)$, then the log-likelihood is:

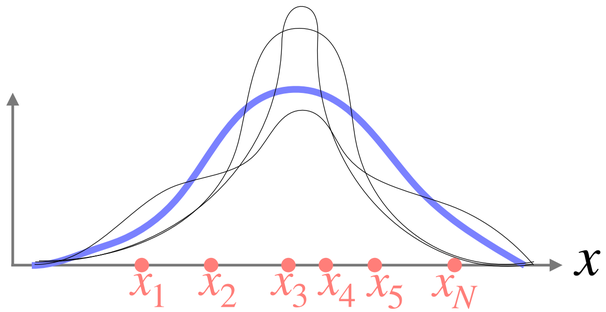
$$\log \mathscr{L}(\theta) = \sum_{i=1}^{N}\log p_\theta(x_i)\,.$$

And the whole process above (steps 1-3) is called maximizing log-likelihood of the data.

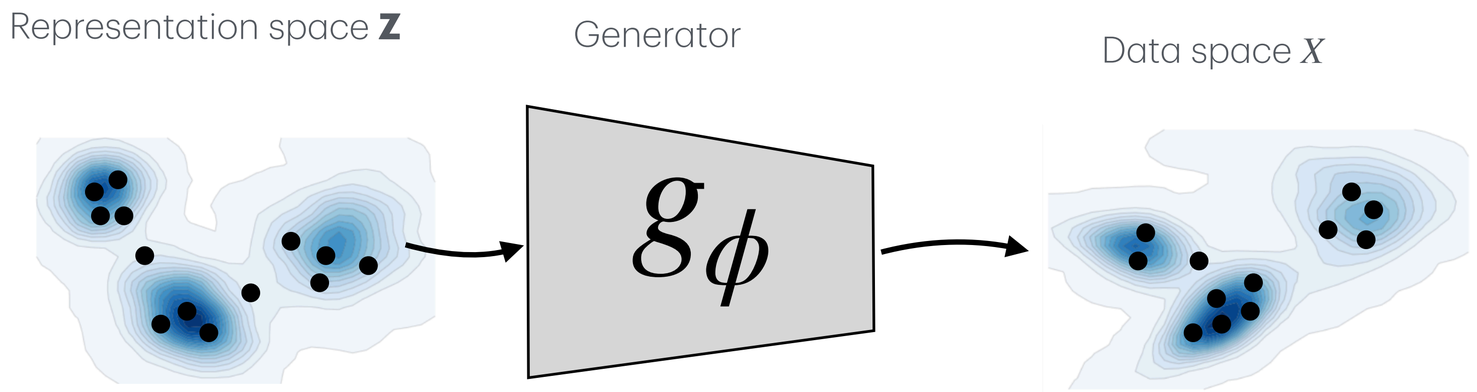
p_θ for non optimal θ

p_θ^*: θ^* that maximizes likelihood:

$\approx \arg\max_\theta \frac{1}{N}\sum_{i=1}^{N}\log p_\theta(x_i)$



In the next lecture we will explore this problem:



We don't know $p(z)$

We don't know g_ϕ

We don't know p_{data}

Add we have is some samples in Data space $\{x_i\} \in X$