Title: Lecture - Machine Learning, PHYS 777

Speakers: Mohamed Hibat Allah

Collection/Series: Machine Learning (Elective), PHYS 777, February 24 - March 28, 2025

Subject: Condensed Matter, Other

Date: March 25, 2025 - 9:00 AM

URL: https://pirsa.org/25030042

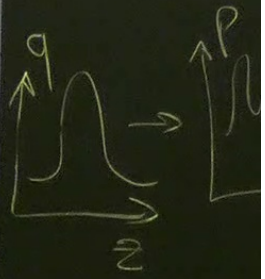
Lecture 11

Today:

↳ Generative modeling: Variational auto-encoders (VAE) and Generative Adversarial Networks (GANs)

↳ Work at the intersection of "Quantum" and ML.

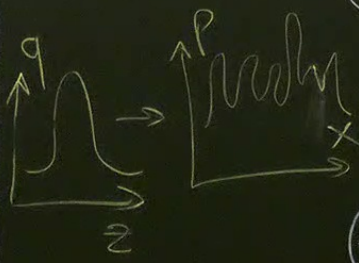
Last



Last time: Generative models

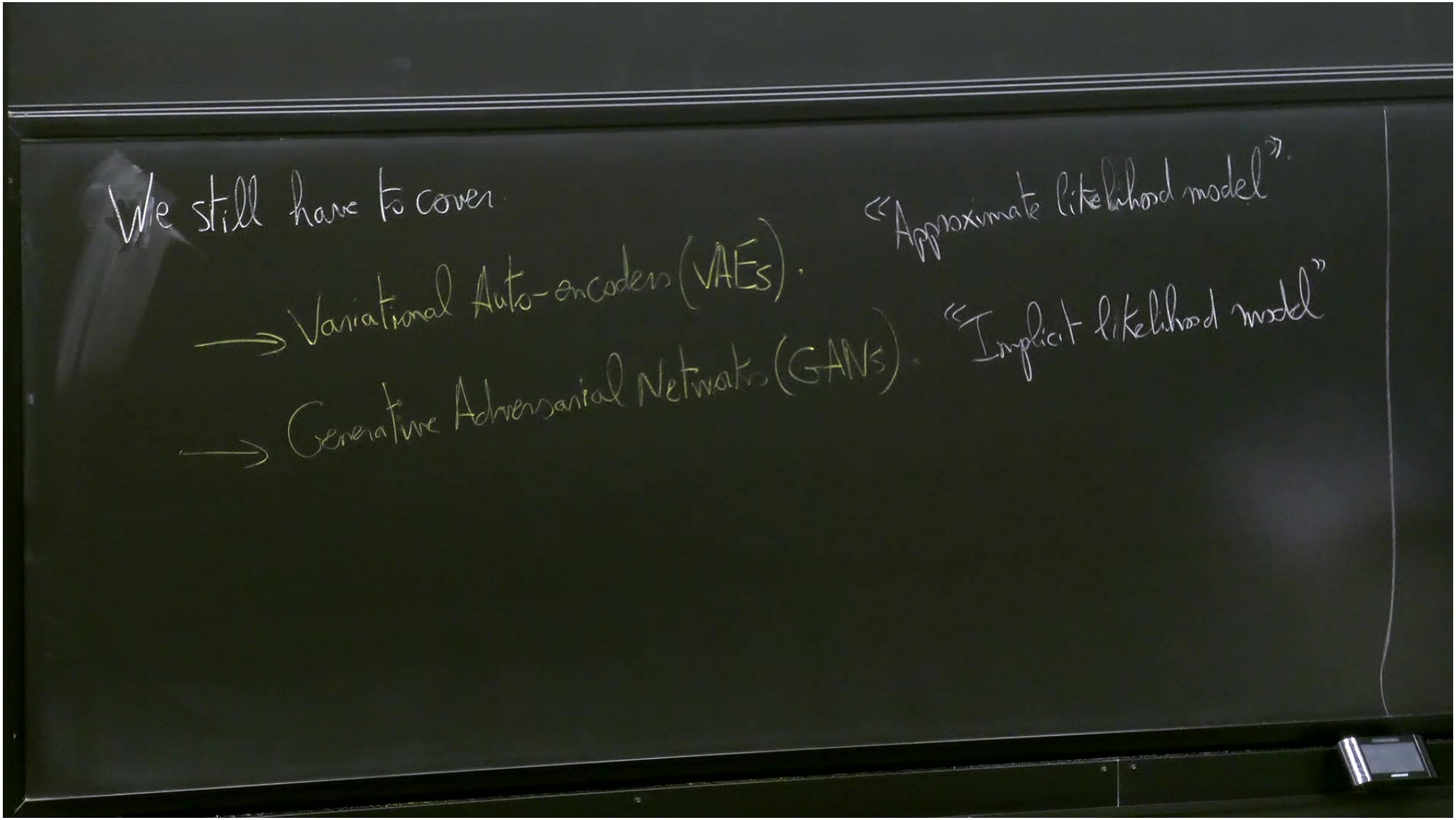
$$P(\vec{x}) = P(x_1)\,P(x_2|x_1)\cdots P(x_N|x_{N-1},\ldots,x_1)$$

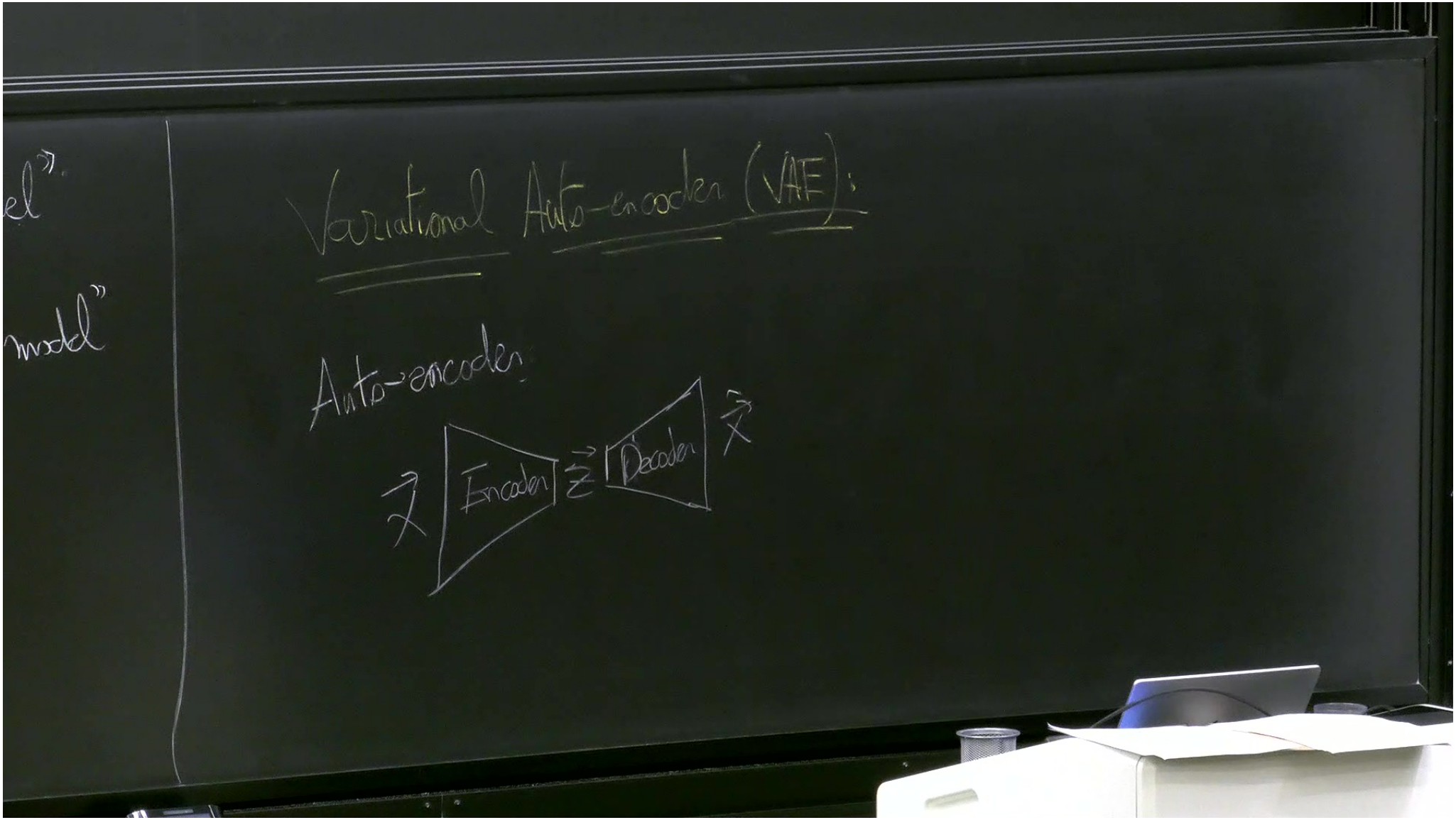
(1) Exact likelihood models
→ Autoregressive models (Discrete data)
→ Normalizing flows (Continuous data)

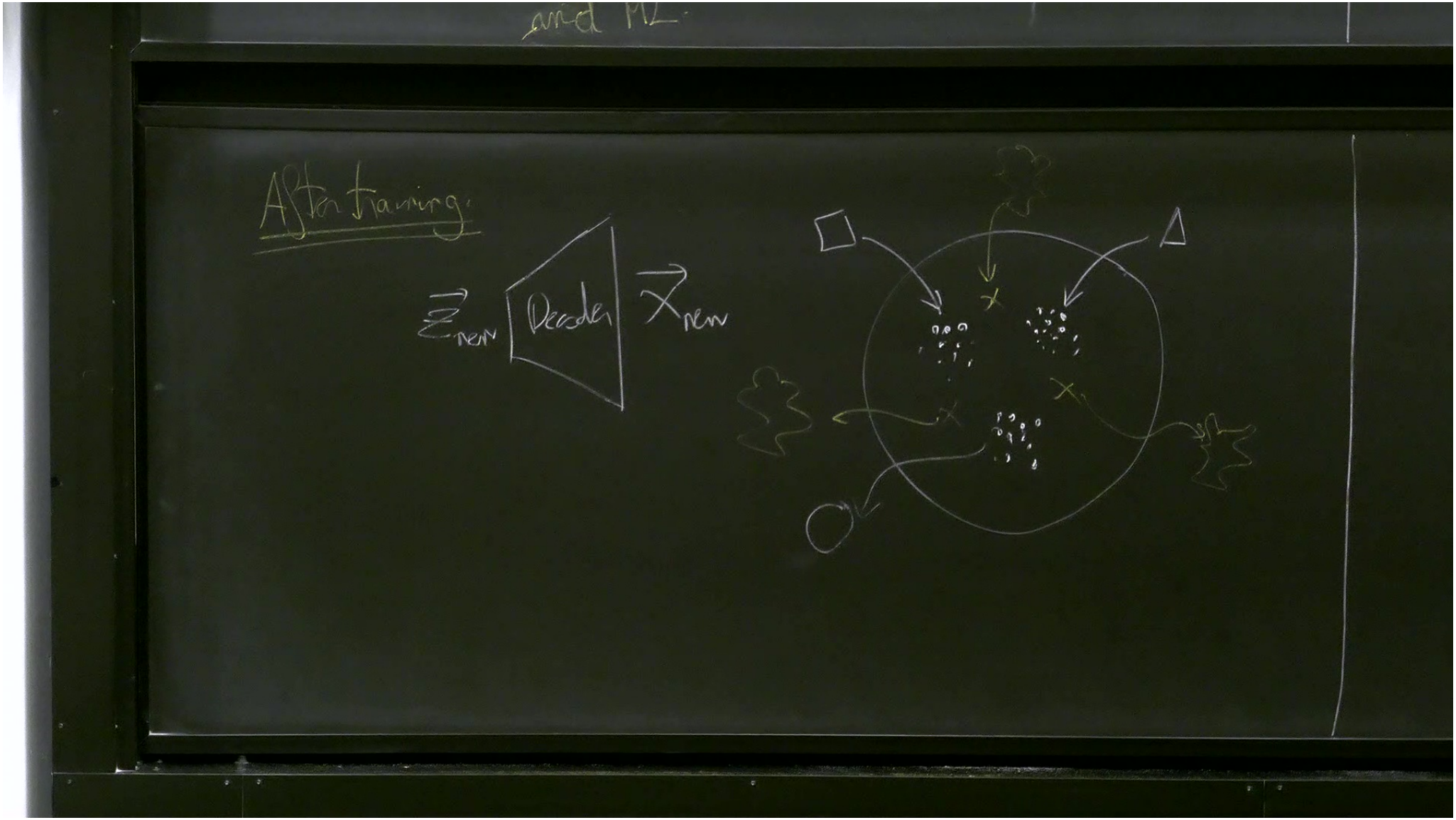


(2) Approximate likelihood models
→ Energy-based models

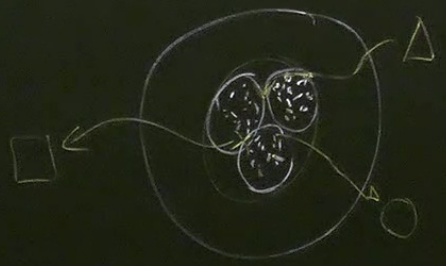
$$P_\theta(\vec{x}) = \frac{\exp(-E_\theta(\vec{x}))}{Z}$$


We still have to cover.
→ Variational Auto-encoders (VAEs). "Approximate likelihood model"
→ Generative Adversarial Networks (GANs). "Implicit likelihood model"


Variational Auto-encoder (VAE):
Auto-encoder:
Encoder
Decoder


and ML.
After training:
$\vec{z}_{new}$
Decoder
$\vec{x}_{new}$

In a VAE.



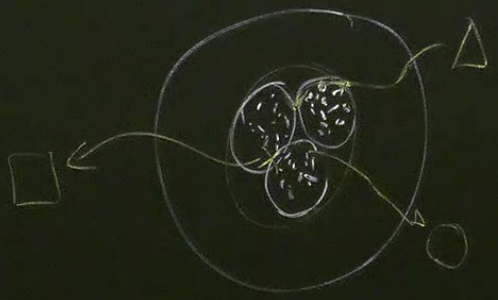
x → Encoder → $\vec{\mu}$, $\vec{\sigma}$ → $\vec{z}$ → Decoder

$\vec{z} \sim N(\vec{\mu}, \vec{\sigma}^2)$

$\vec{z} = \vec{\mu}_\theta + \vec{\sigma}_\theta \odot \vec{\varepsilon}$

$\vec{\varepsilon} \sim N(0, I)$.

In a VAE.

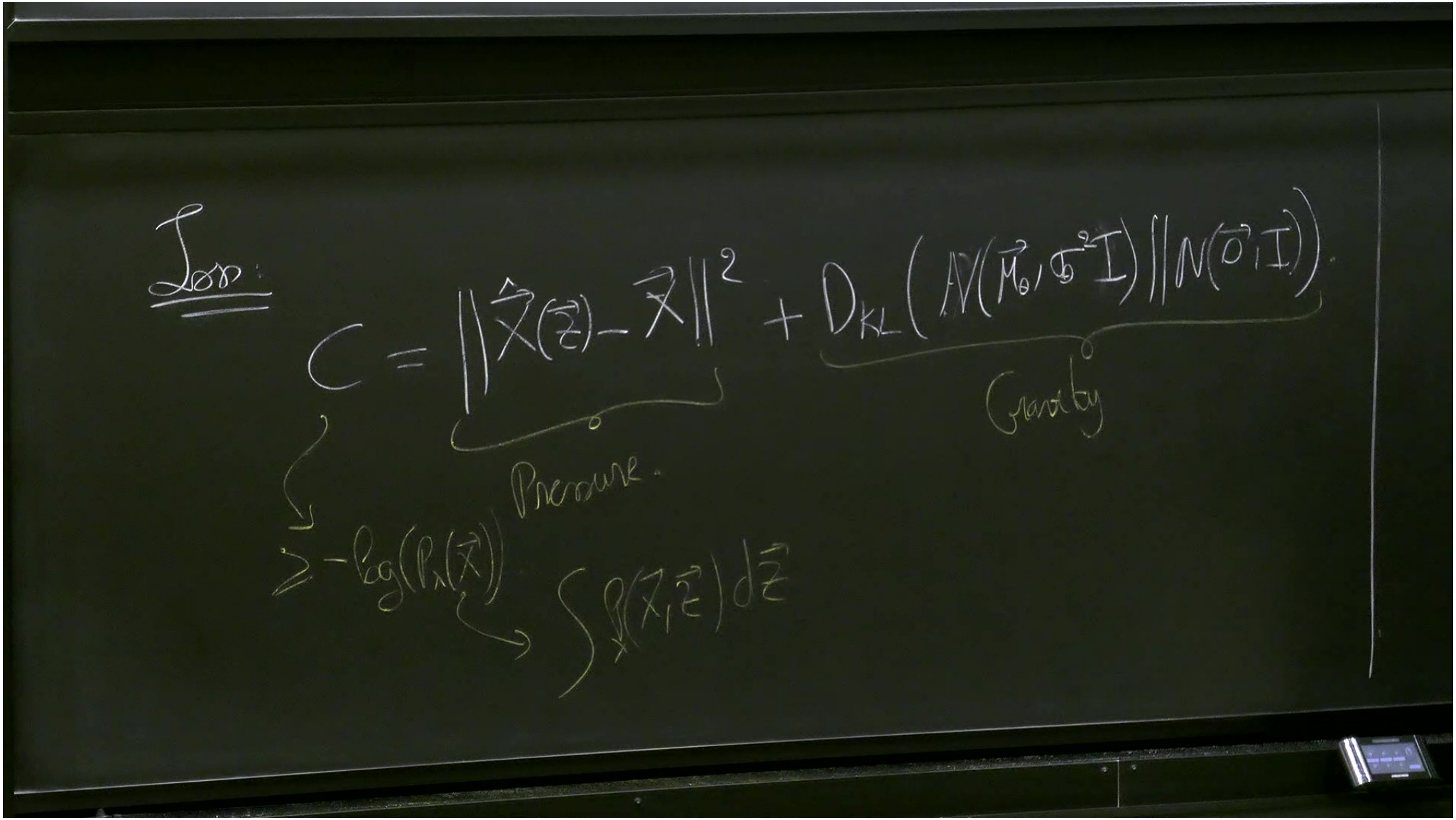


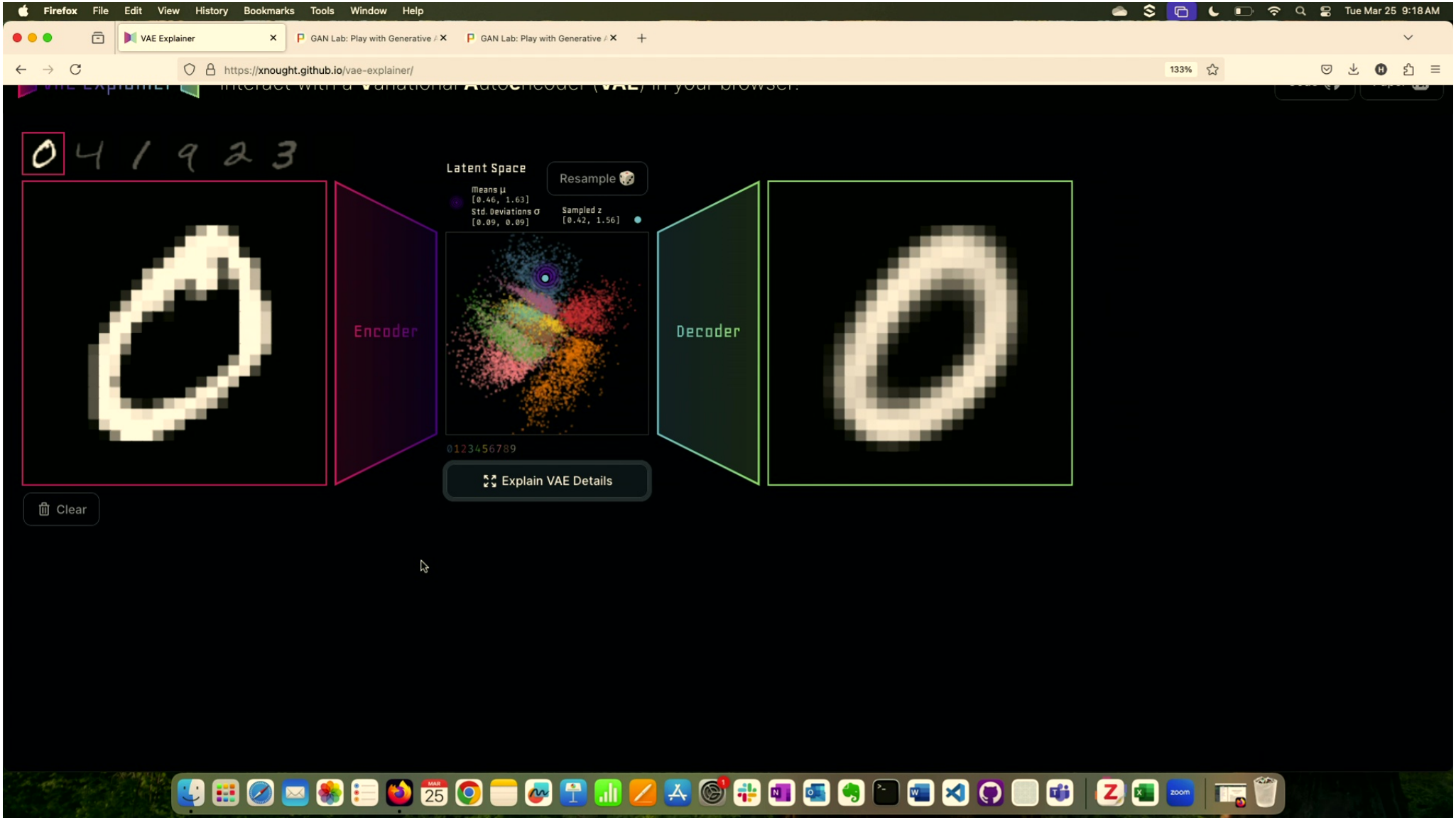
x → Encoder → $\vec{\mu}$, $\vec{\sigma}$ → $\vec{z}$ → Decoder → $\hat{x}$

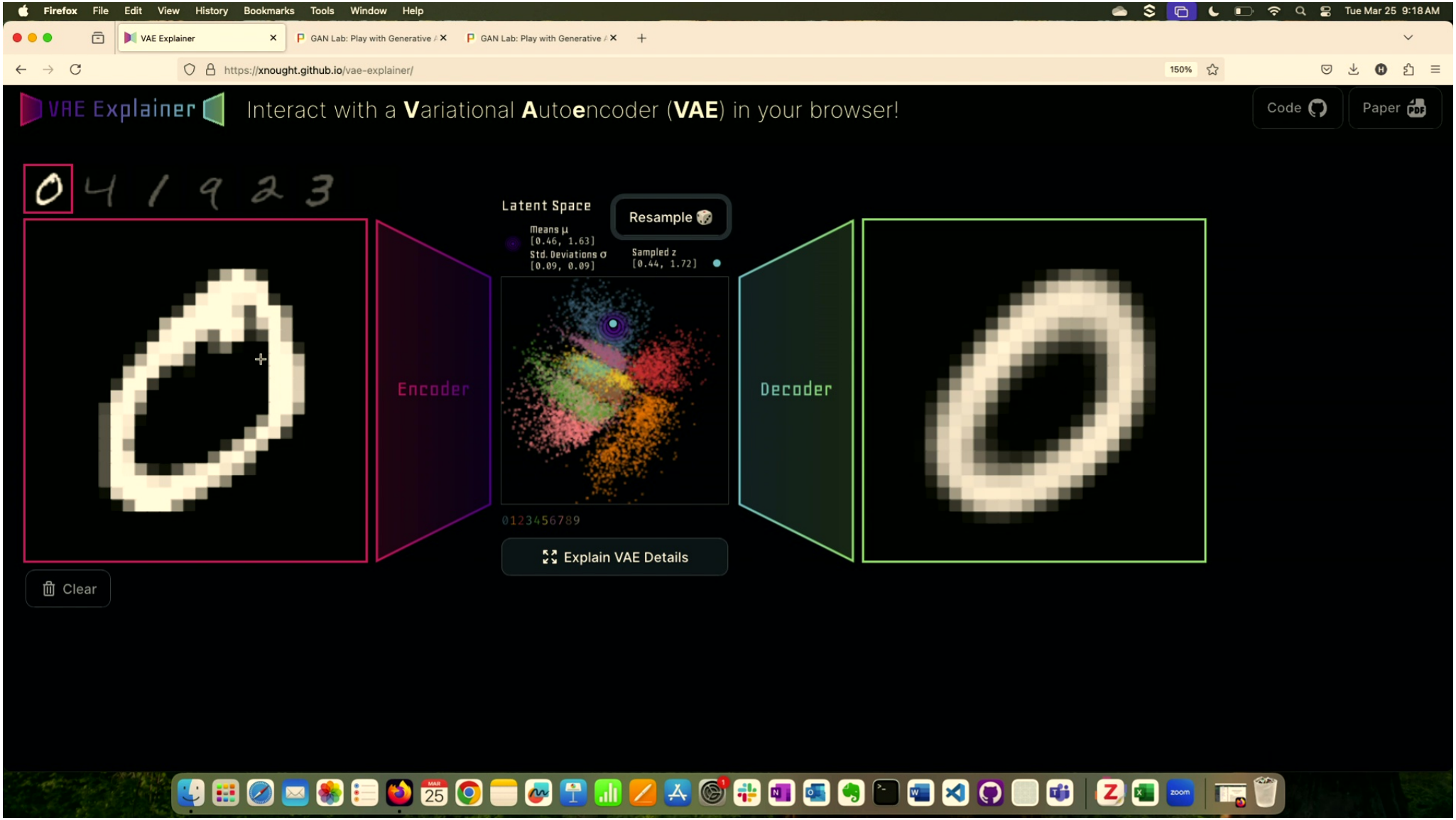
$\vec{z} \sim N(\vec{\mu}, \vec{\sigma}^2) \longrightarrow N(\vec{0}, I)$

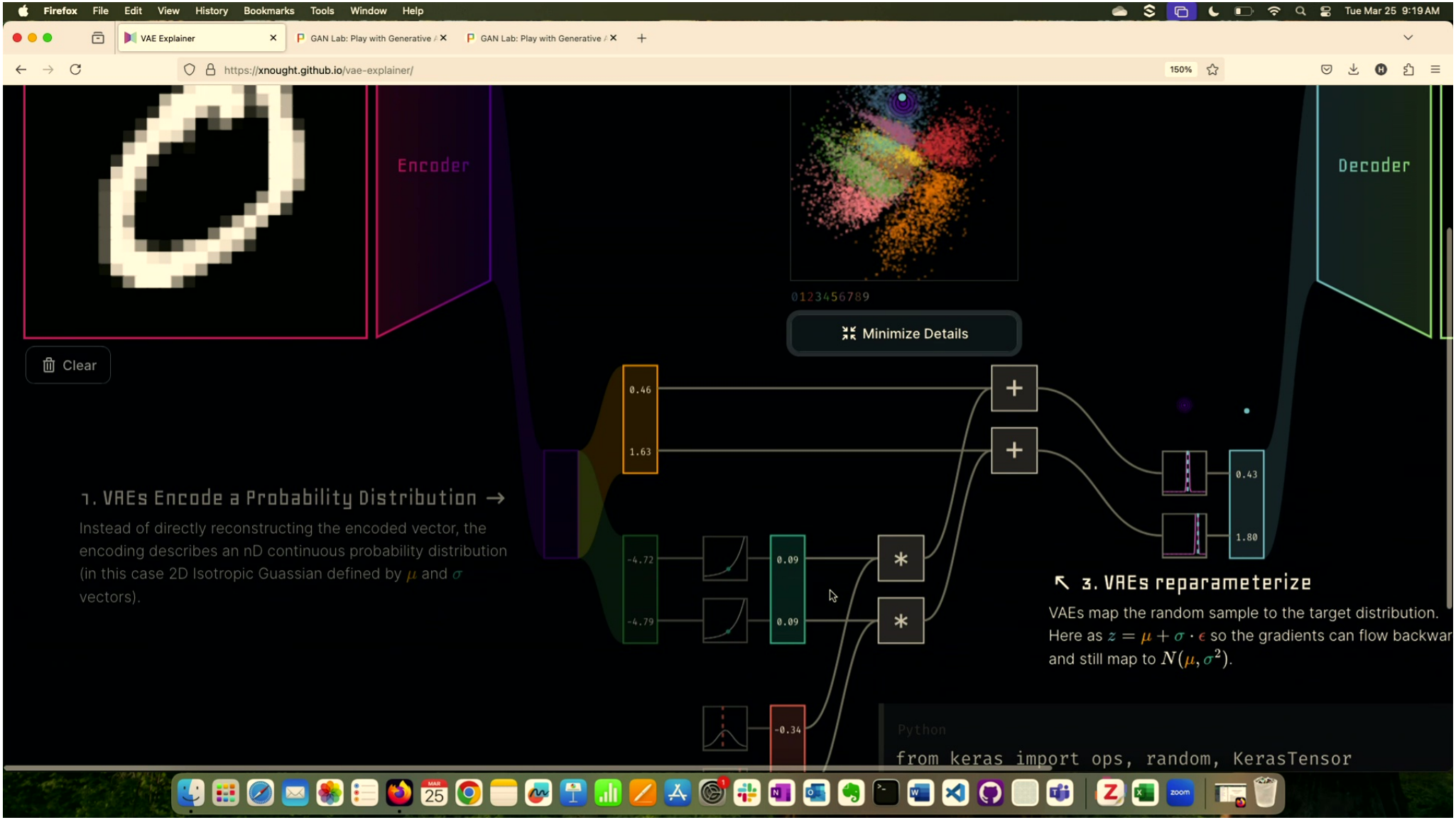
$\vec{z} = \vec{\mu}_\theta + \vec{\sigma}_\theta \odot \vec{\varepsilon}$

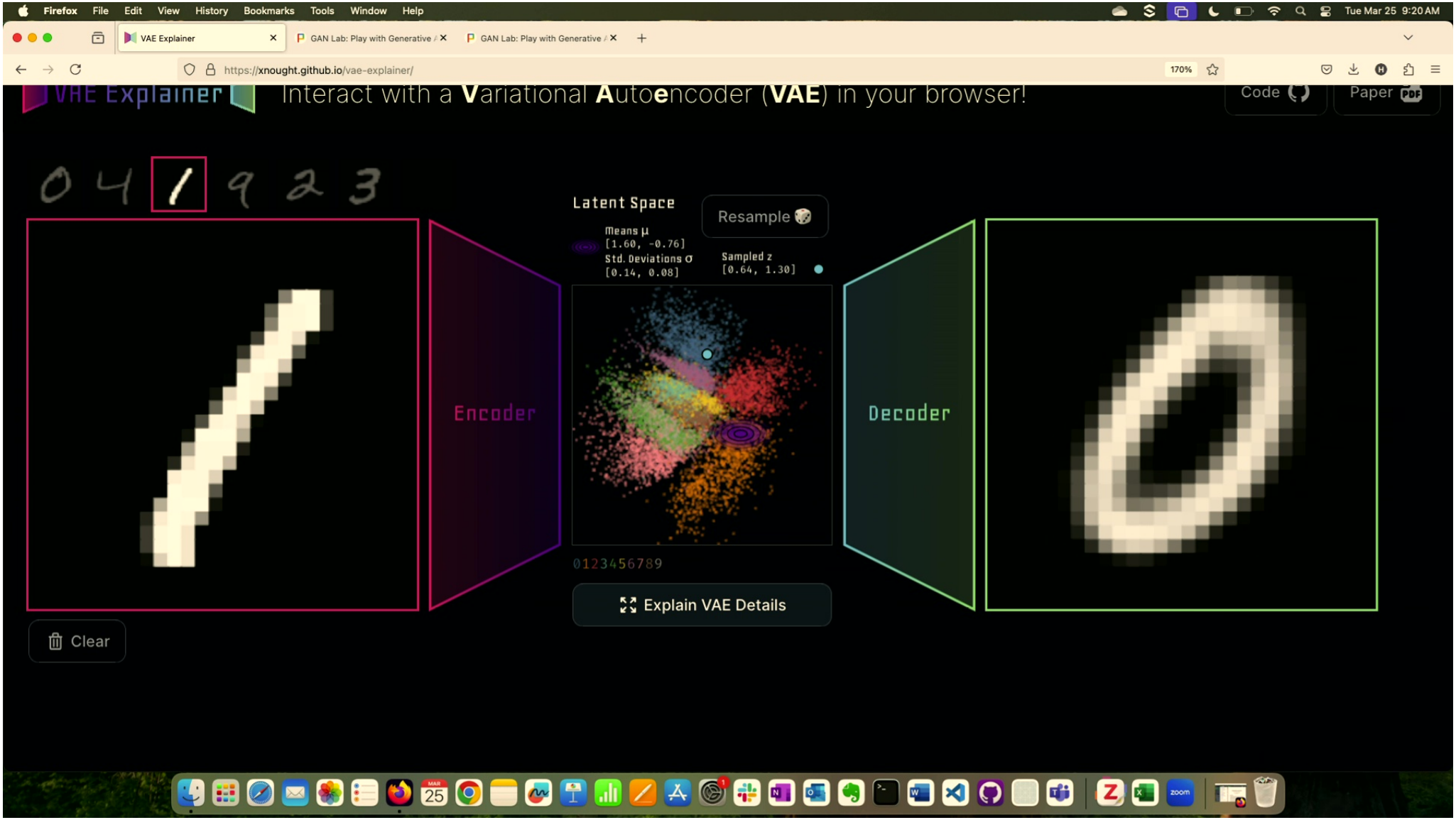
$\vec{\varepsilon} \sim N(\vec{0}, I)$.

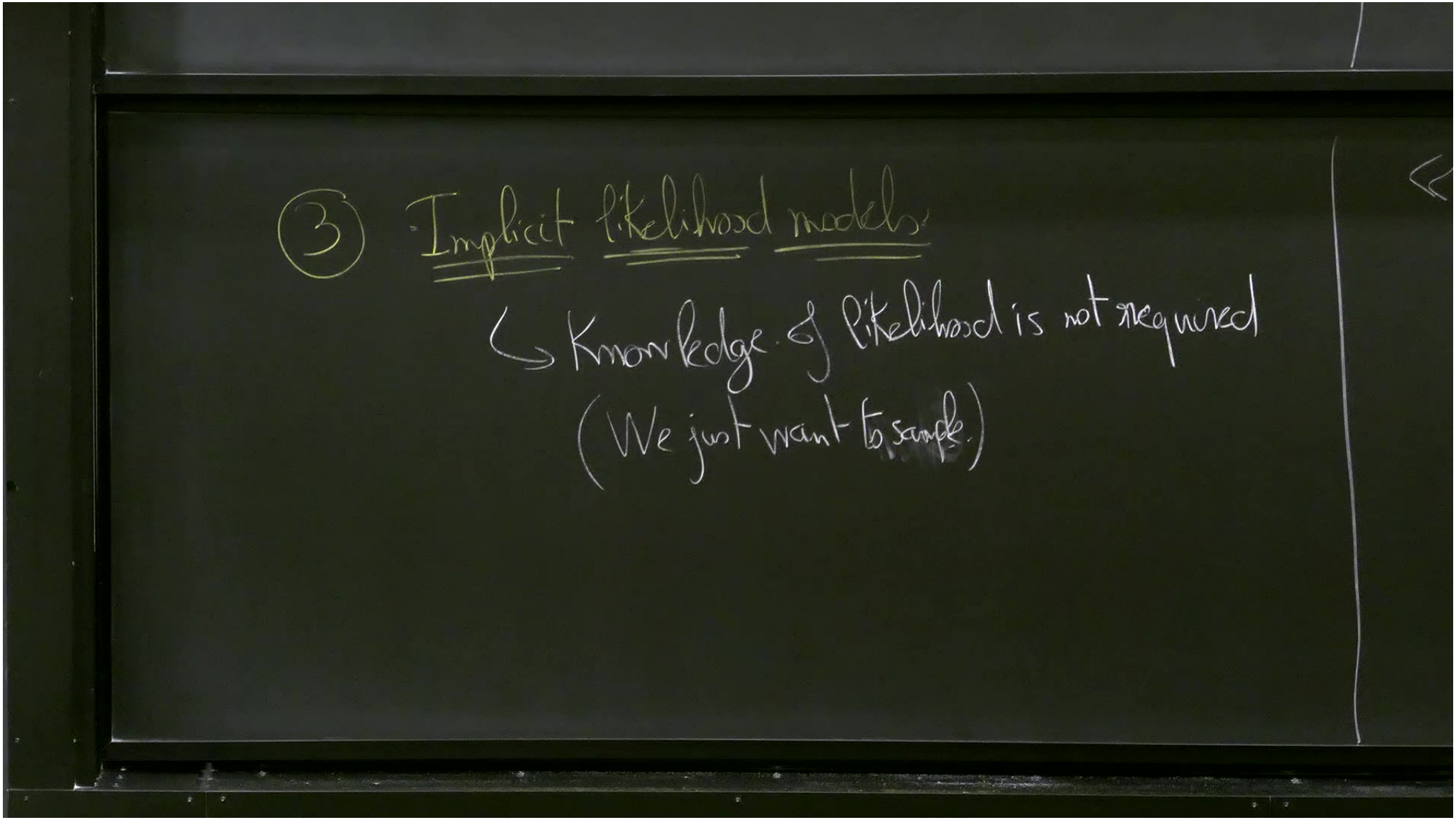

Loss:
$C = \|\hat{\vec{x}}(\vec{z}) - \vec{x}\|^2 + D_{KL}\left(N(\vec{\mu}_\theta, \sigma^2 I) \,\|\, N(\vec{0}, I)\right)$
Pressure
Gravity
$\geq -\log(P_x(\vec{x}))$
$\int P(\vec{x}, \vec{z})\, d\vec{z}$


Firefox File Edit View History Bookmarks Tools Window Help
Tue Mar 25 9:18 AM
VAE Explainer
GAN Lab: Play with Generative
GAN Lab: Play with Generative
https://xnought.github.io/vae-explainer/
133%
0 4 1 9 2 3
Latent Space
Resample
Means μ
[0.46, 1.63]
Std. Deviations σ
[0.09, 0.09]
Sampled z
[0.42, 1.56]
Encoder
Decoder
0123456789
Explain VAE Details
Clear


Firefox File Edit View History Bookmarks Tools Window Help
Tue Mar 25 9:18 AM
VAE Explainer
GAN Lab: Play with Generative
GAN Lab: Play with Generative
https://xnought.github.io/vae-explainer/
150%
VAE Explainer
Interact with a Variational Autoencoder (VAE) in your browser!
Code
Paper
Latent Space
Resample
Means μ
[0.46, 1.63]
Std. Deviations σ
[0.09, 0.09]
Sampled z
[0.44, 1.72]
Encoder
Decoder
0123456789
Explain VAE Details
Clear


Firefox File Edit View History Bookmarks Tools Window Help
Tue Mar 25 9:19 AM
VAE Explainer
GAN Lab: Play with Generative
GAN Lab: Play with Generative
https://xnought.github.io/vae-explainer/
150%
Encoder
Decoder
0123456789
Minimize Details
Clear
0.46
1.63
-4.72
-4.79
0.09
0.09
-0.34
0.43
1.80
1. VAEs Encode a Probability Distribution →
Instead of directly reconstructing the encoded vector, the encoding describes an nD continuous probability distribution (in this case 2D Isotropic Guassian defined by μ and σ vectors).
↖ 3. VAEs reparameterize
VAEs map the random sample to the target distribution. Here as $z = \mu + \sigma \cdot \epsilon$ so the gradients can flow backwar
and still map to $N(\mu, \sigma^2)$.
Python
from keras import ops, random, KerasTensor

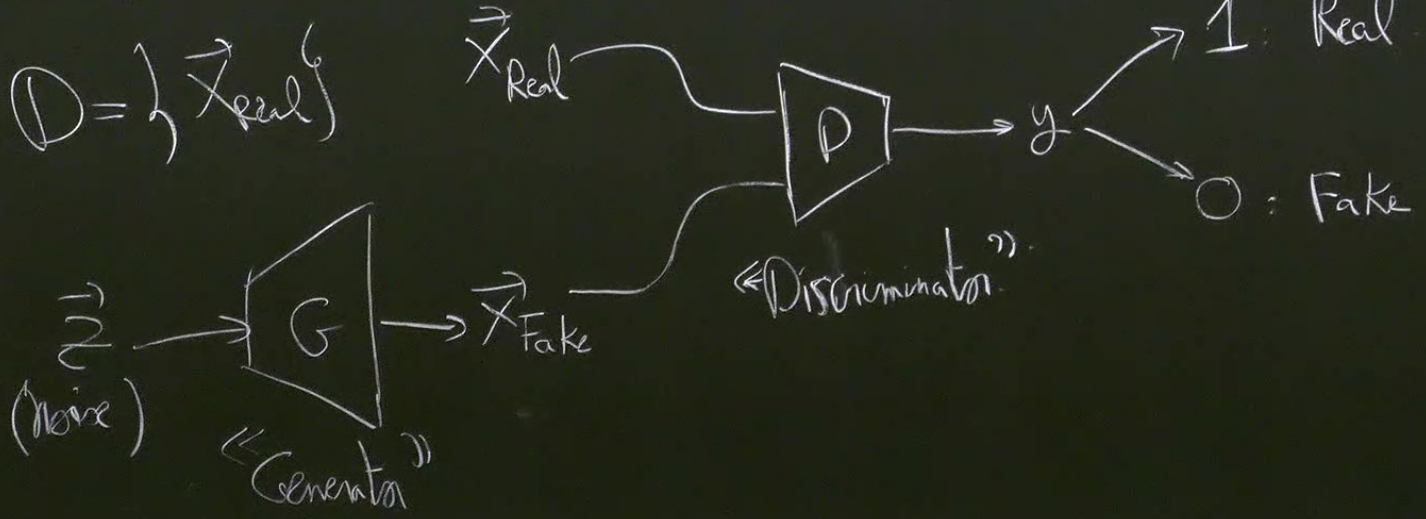

Firefox File Edit View History Bookmarks Tools Window Help
Tue Mar 25 9:20 AM
VAE Explainer
GAN Lab: Play with Generative
GAN Lab: Play with Generative
https://xnought.github.io/vae-explainer/
170%
VAE Explainer
Interact with a Variational Autoencoder (VAE) in your browser!
Code
Paper
Latent Space
Resample
Means μ
[1.60, -0.76]
Std. Deviations σ
[0.14, 0.08]
Sampled z
[0.64, 1.30]
Encoder
Decoder
0123456789
Explain VAE Details
Clear

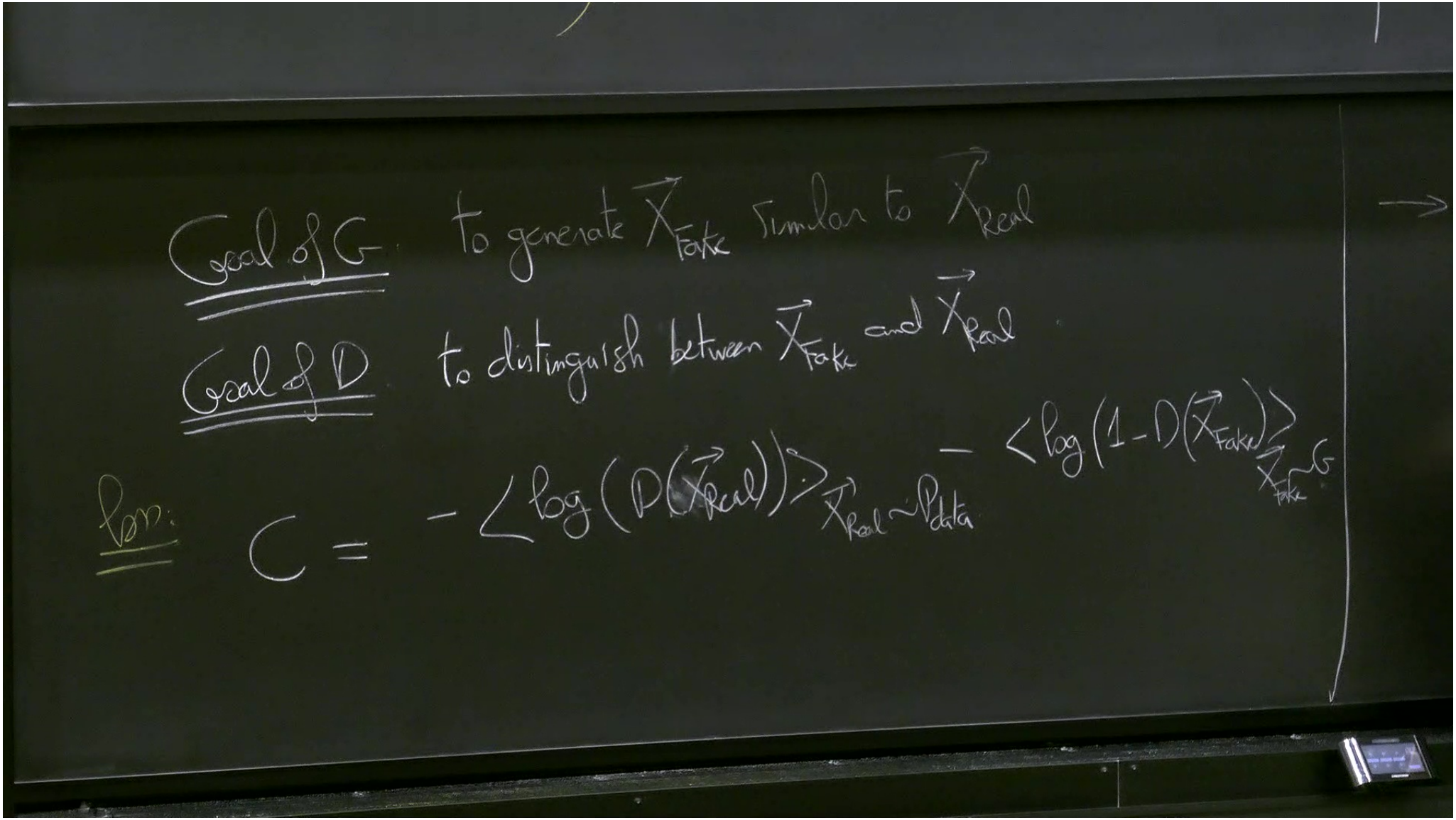

③ Implicit likelihood models
↳ Knowledge of likelihood is not required
(We just want to sample.)

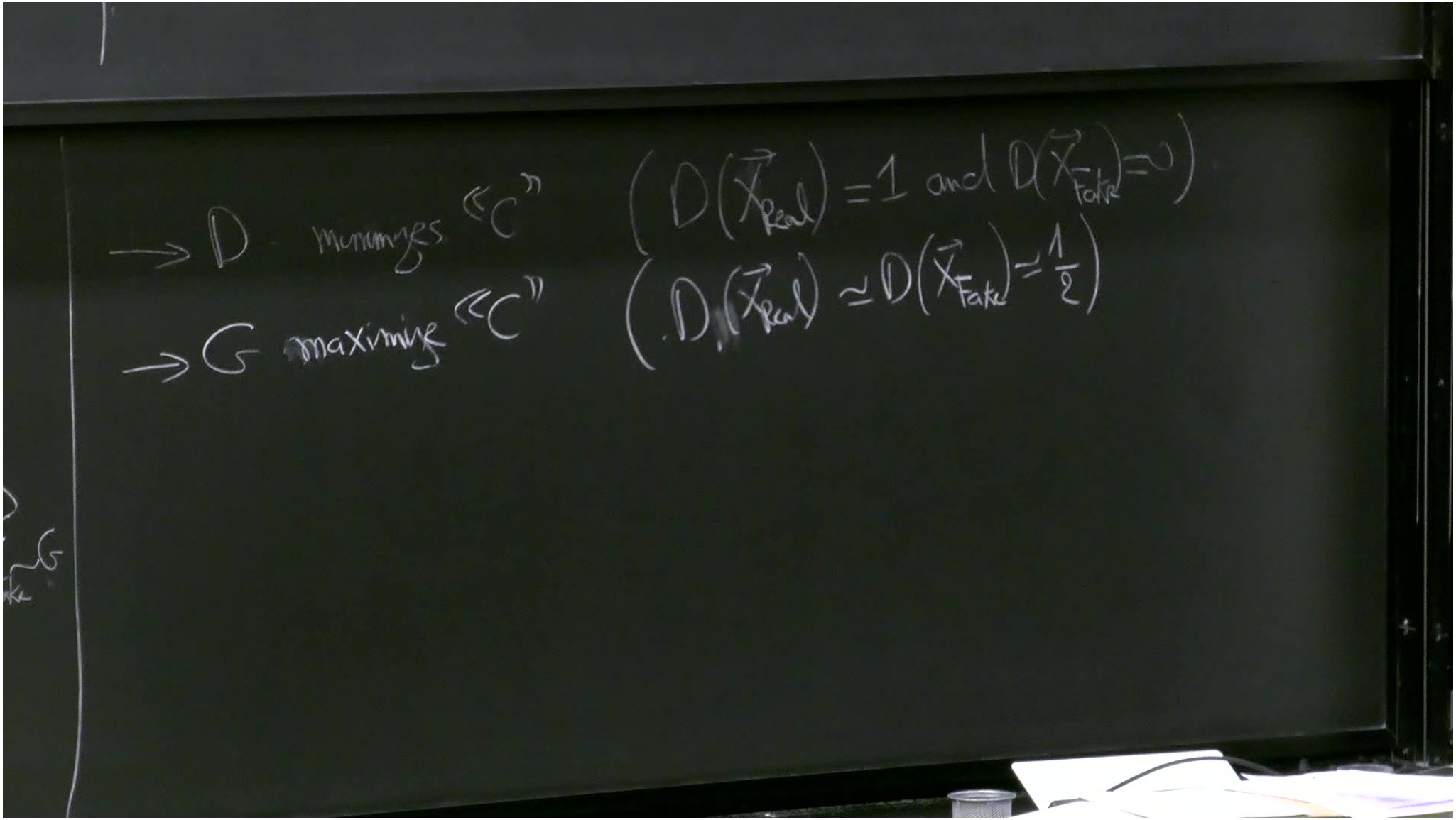
$\varepsilon \sim N(0, I)$.

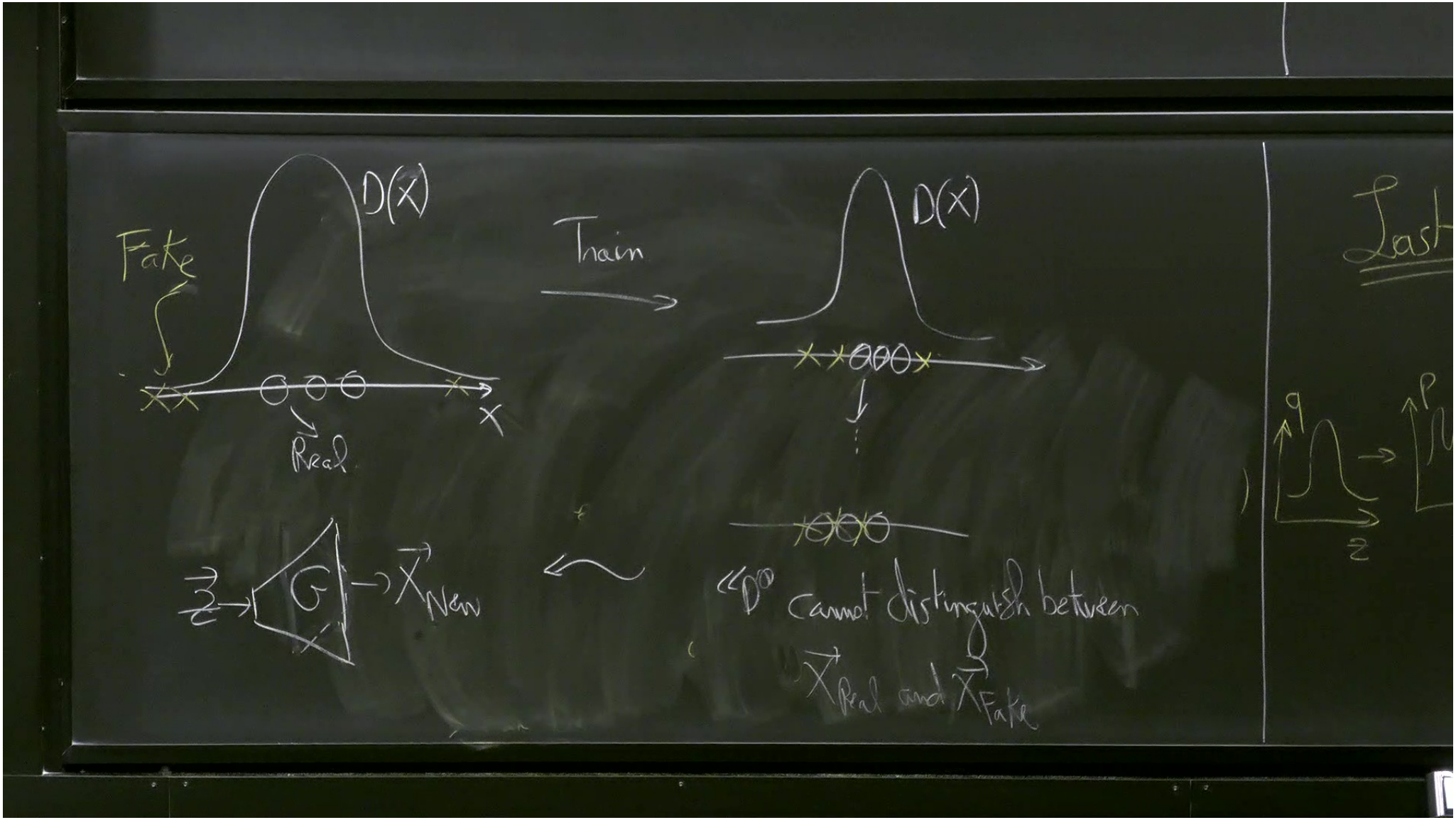
"Generative Adversarial Networks" (GAN).

$D = \{\vec{x}_{real}\}$




Goal of G: to generate $\vec{X}_{Fake}$ similar to $\vec{X}_{Real}$
Goal of D: to distinguish between $\vec{X}_{Fake}$ and $\vec{X}_{Real}$
Loss: $C = -\langle \log(D(\vec{X}_{Real})) \rangle_{\vec{X}_{Real} \sim P_{data}} - \langle \log(1 - D(\vec{X}_{Fake})) \rangle_{\vec{X}_{Fake} \sim G}$


→ D minimizes « C »
(D(X_Real) = 1 and D(X_Fake) = 0)
→ G maximizing « C »
(D(X_Real) ≃ D(X_Fake) ≃ 1/2)


Fake
D(x)
Real
x
Train
D(x)
Last
q
p
z
G
$\vec{x}_{New}$
"D" cannot distinguish between
$\vec{x}_{Real}$ and $\vec{x}_{Fake}$

Play with Generative Adversarial Networks (GANs) in your browser!

Fork us on GitHub

GAN Lab

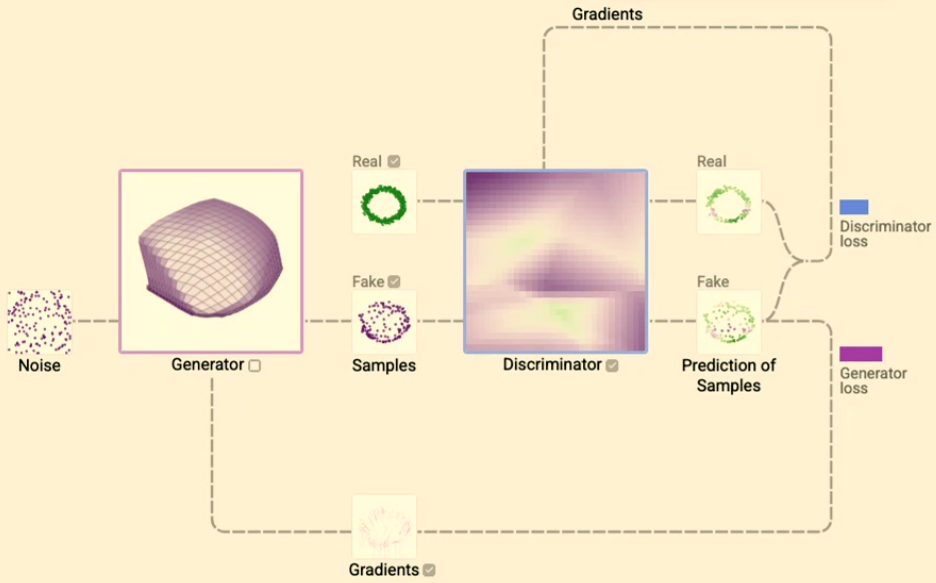
Data Distribution

Use pre-trained model

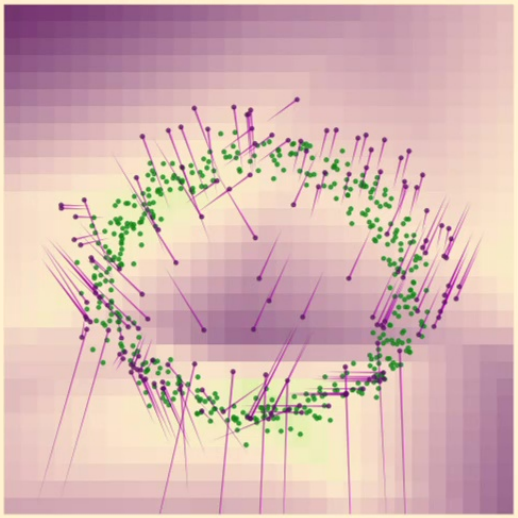
Epoch
001,048

MODEL OVERVIEW GRAPH

Shows the architecture of a model and data flow



LAYERED DISTRIBUTIONS



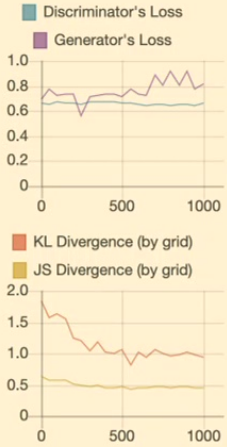
Each dot is a 2D data sample: real samples; fake samples.

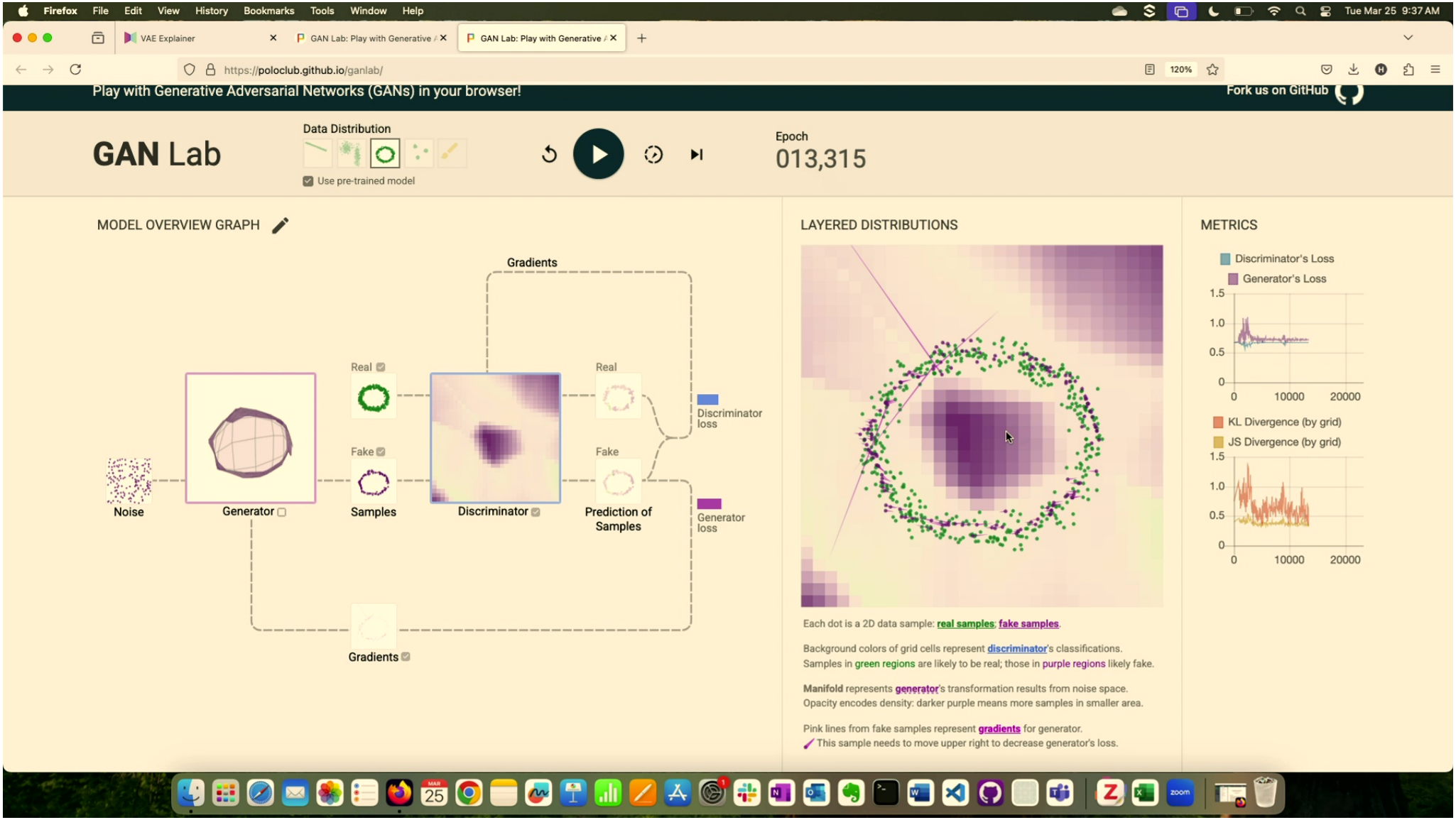
Background colors of grid cells represent discriminator's classifications.
Samples in green regions are likely to be real; those in purple regions likely fake.

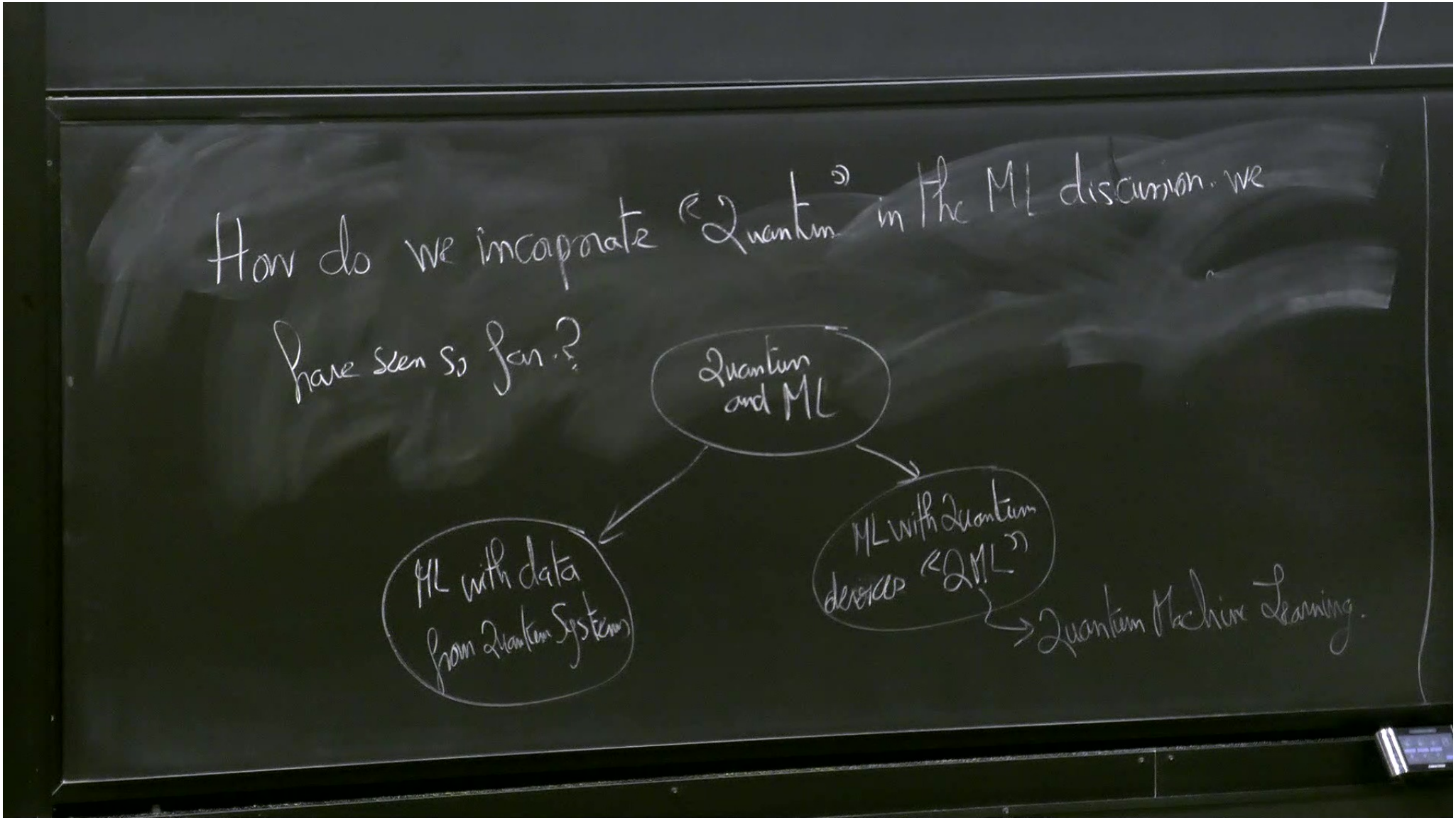
Manifold represents generator's transformation results from noise space.
Opacity encodes density: darker purple means more samples in smaller area.

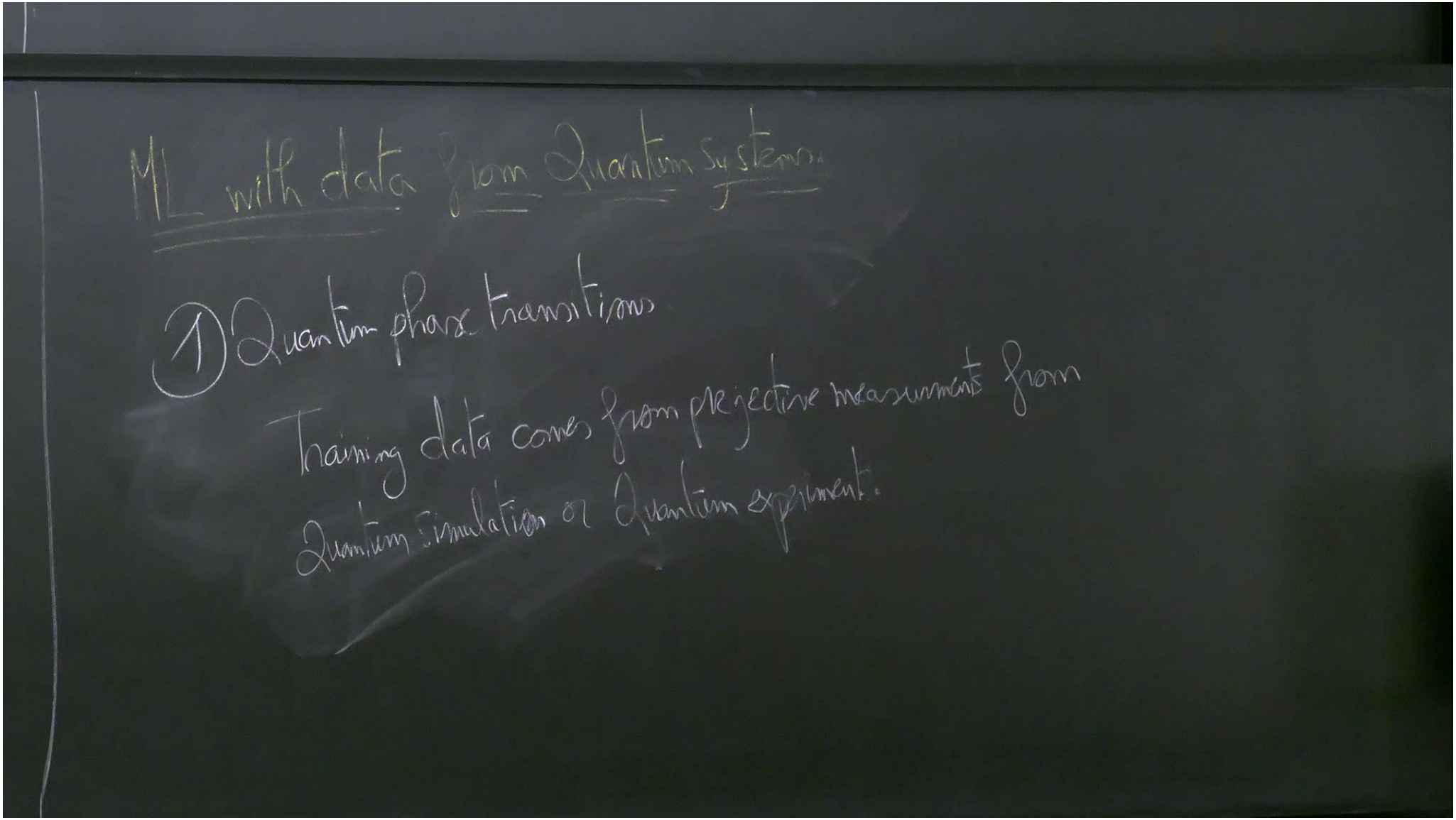
Pink lines from fake samples represent gradients for generator.
This sample needs to move upper right to decrease generator's loss.

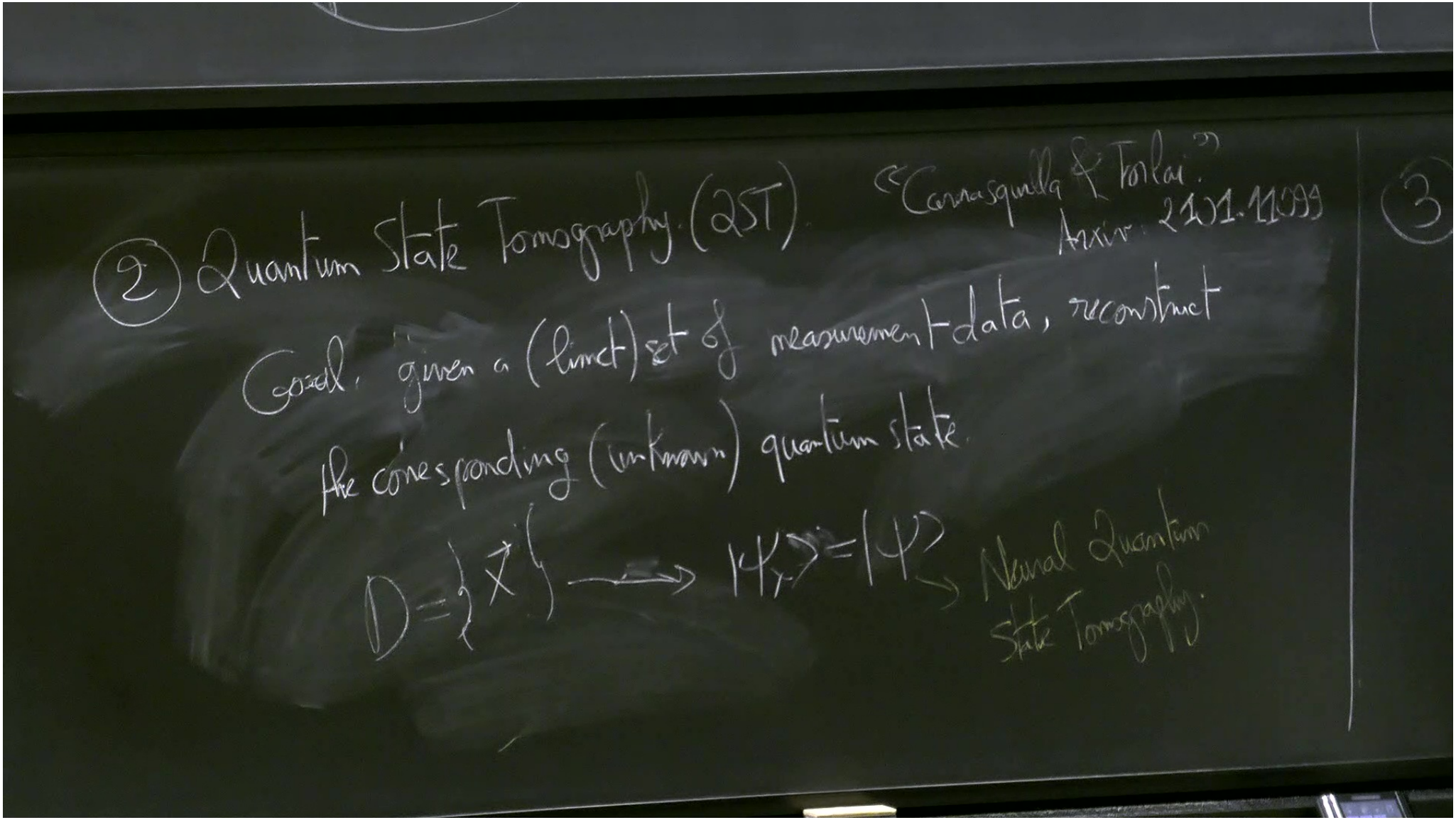
METRICS

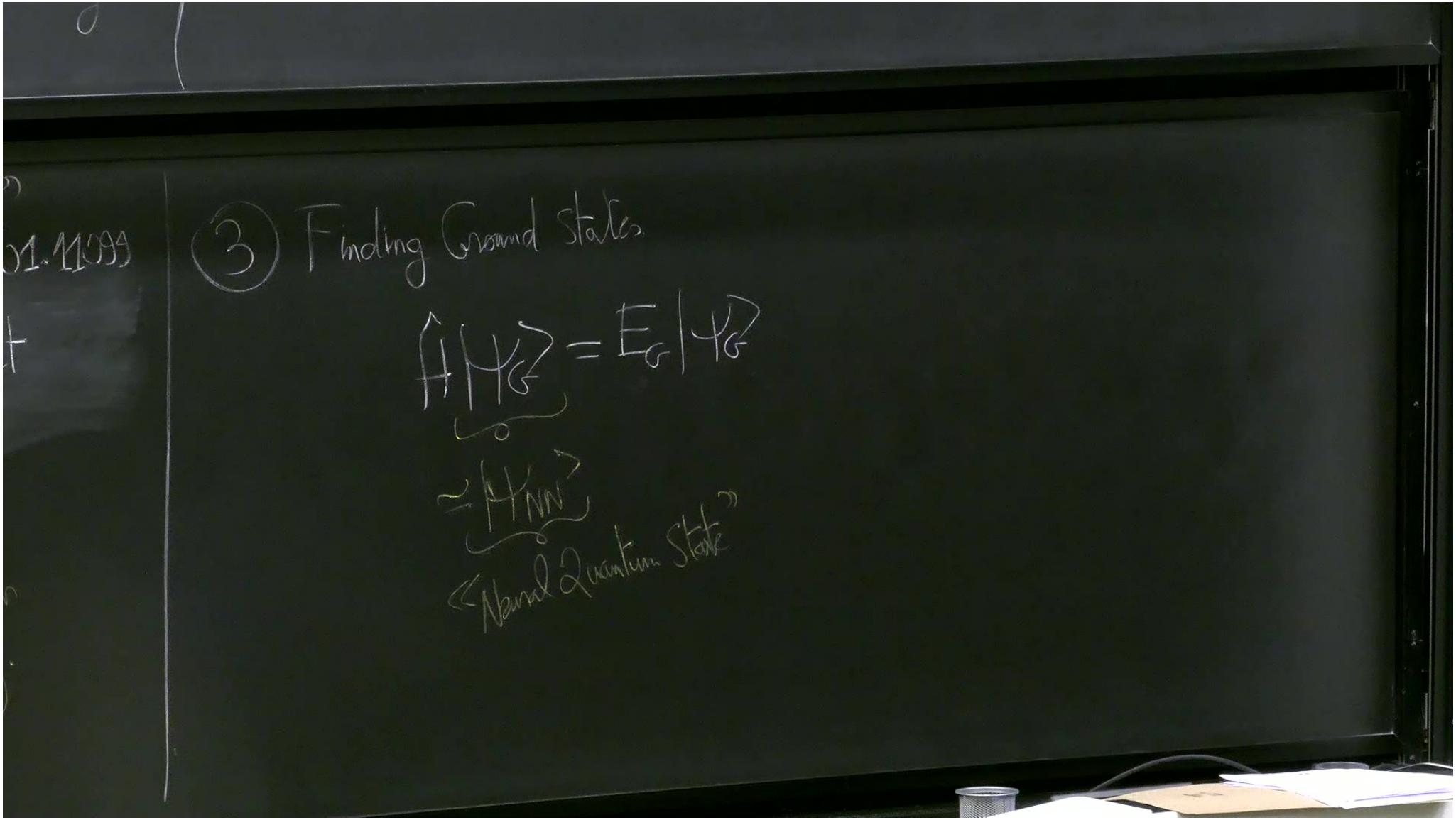



Firefox File Edit View History Bookmarks Tools Window Help
Tue Mar 25 9:37 AM
VAE Explainer
GAN Lab: Play with Generative
GAN Lab: Play with Generative
https://poloclub.github.io/ganlab/
120%
Play with Generative Adversarial Networks (GANs) in your browser!
Fork us on GitHub
GAN Lab
Data Distribution
Use pre-trained model
Epoch
013,315
MODEL OVERVIEW GRAPH
Gradients
Real
Fake
Noise
Generator
Samples
Discriminator
Real
Fake
Prediction of Samples
Discriminator loss
Generator loss
Gradients
LAYERED DISTRIBUTIONS
Each dot is a 2D data sample: real samples; fake samples.
Background colors of grid cells represent discriminator's classifications.
Samples in green regions are likely to be real; those in purple regions likely fake.
Manifold represents generator's transformation results from noise space.
Opacity encodes density: darker purple means more samples in smaller area.
Pink lines from fake samples represent gradients for generator.
This sample needs to move upper right to decrease generator's loss.
METRICS
Discriminator's Loss
Generator's Loss
1.5
1.0
0.5
0
0
10000
20000
KL Divergence (by grid)
JS Divergence (by grid)
1.5
1.0
0.5
0
0
10000
20000


How do we incorporate "Quantum" in the ML discussion. We have seen so far.?
Quantum and ML
ML with data from Quantum Systems
ML with Quantum devices "QML"
→ Quantum Machine Learning.


ML with data from Quantum Systems
1) Quantum phase transitions
Training data comes from projective measurements from
Quantum simulation or Quantum experiment.


② Quantum State Tomography (QST)
"Carrasquilla & Torlai"
Anxiv: 2101.11099
Goal: given a (finite) set of measurement data, reconstruct
the corresponding (unknown) quantum state.
$D = \{\vec{x}_i\} \longrightarrow |\psi_\lambda\rangle = |\psi\rangle$
Neural Quantum
State Tomography.
③


01.11099
3 Finding Ground states
$\hat{H}|\Psi_G\rangle = E_G|\Psi_G\rangle$
$\simeq |\Psi_{NN}\rangle$
"Neural Quantum State"

→ Quantum Machine Learning (QML)

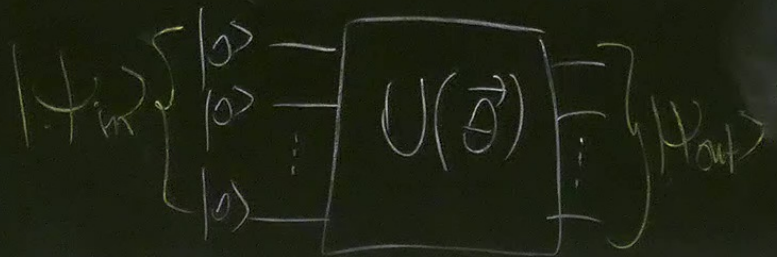
Goal: Use quantum computers/Devices to help with (speed or innovate) ML tasks

Generator

Variational Circuits → Depend on parameters $\theta = \{\theta_1, \theta_2, \dots\}$

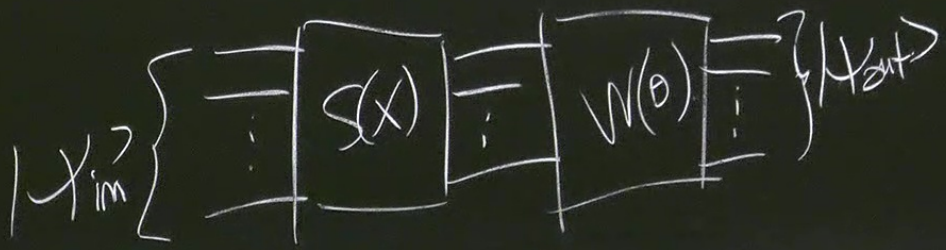
$$|\psi_{in}\rangle = |00\dots0\rangle$$

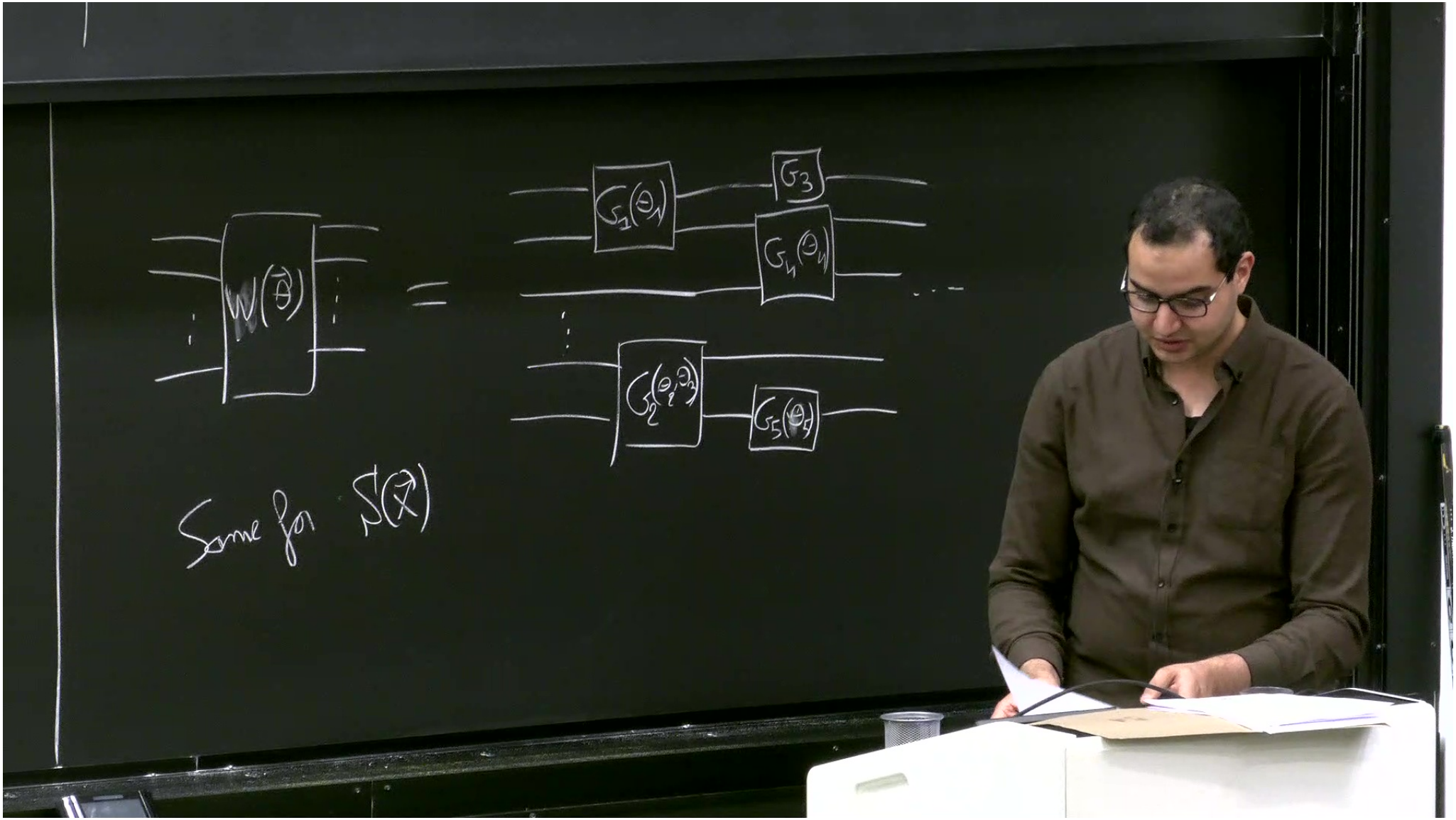
$$|\psi_{out}\rangle = U(\vec{\theta}) \cdot |\psi_{in}\rangle$$



$|\psi_{out}\rangle = U(\vec{x},\vec{\theta})\,|\psi_{in}\rangle$

$U(\vec{x},\vec{\theta}) = \underbrace{W(\vec{\theta})}_{\text{Processing}}\ \underbrace{S(\vec{x})}_{\text{Encoding}}$




W(θ)
G1(θ1)
G3
G4(θ4)
G2(θ2,θ3)
G5(θ5)
Same for S(x)