Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Jascha Sohl-Dickstein[1], Eric Weiss[2],
Niru Maheswaranathan[3], Surya Ganguli[3]

[1] Google Brain, [2] UC Berkeley, [3] Stanford University

Eric

Niru

Surya

Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Outline

- **Motivation: The promise of deep unsupervised learning**
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

The Promise of Deep Unsupervised Learning

The Promise of Deep Unsupervised Learning

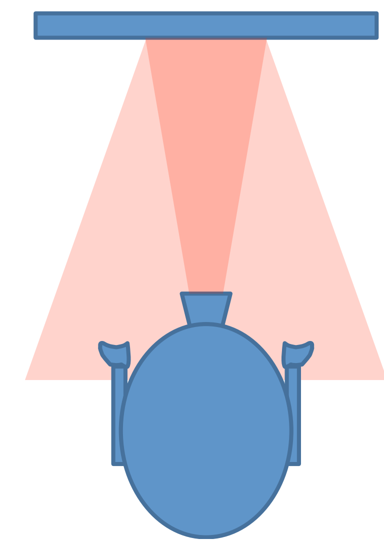
- Unknown features/labels

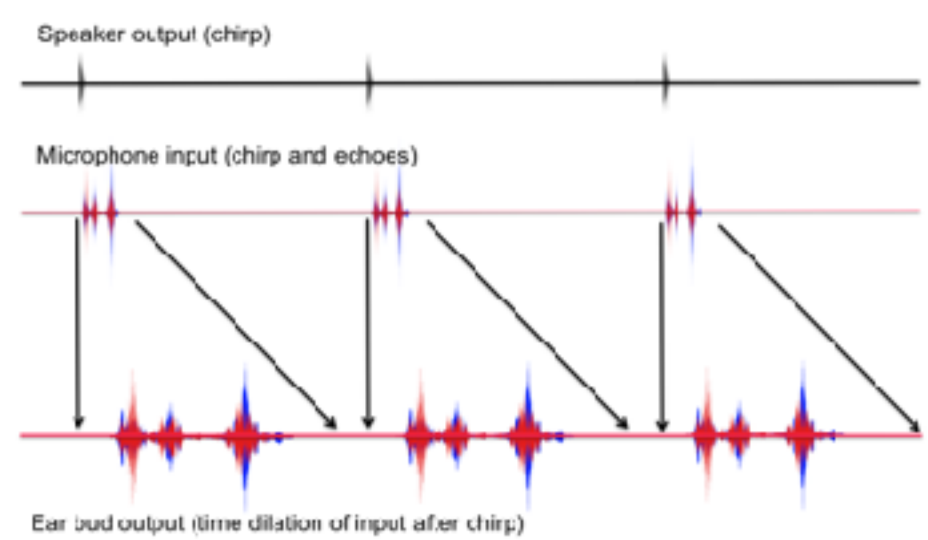
The Promise of Deep Unsupervised Learning

- Unknown features/labels
    - Novel modalities

The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities





[Trans Biomed Eng, 2015]

The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities

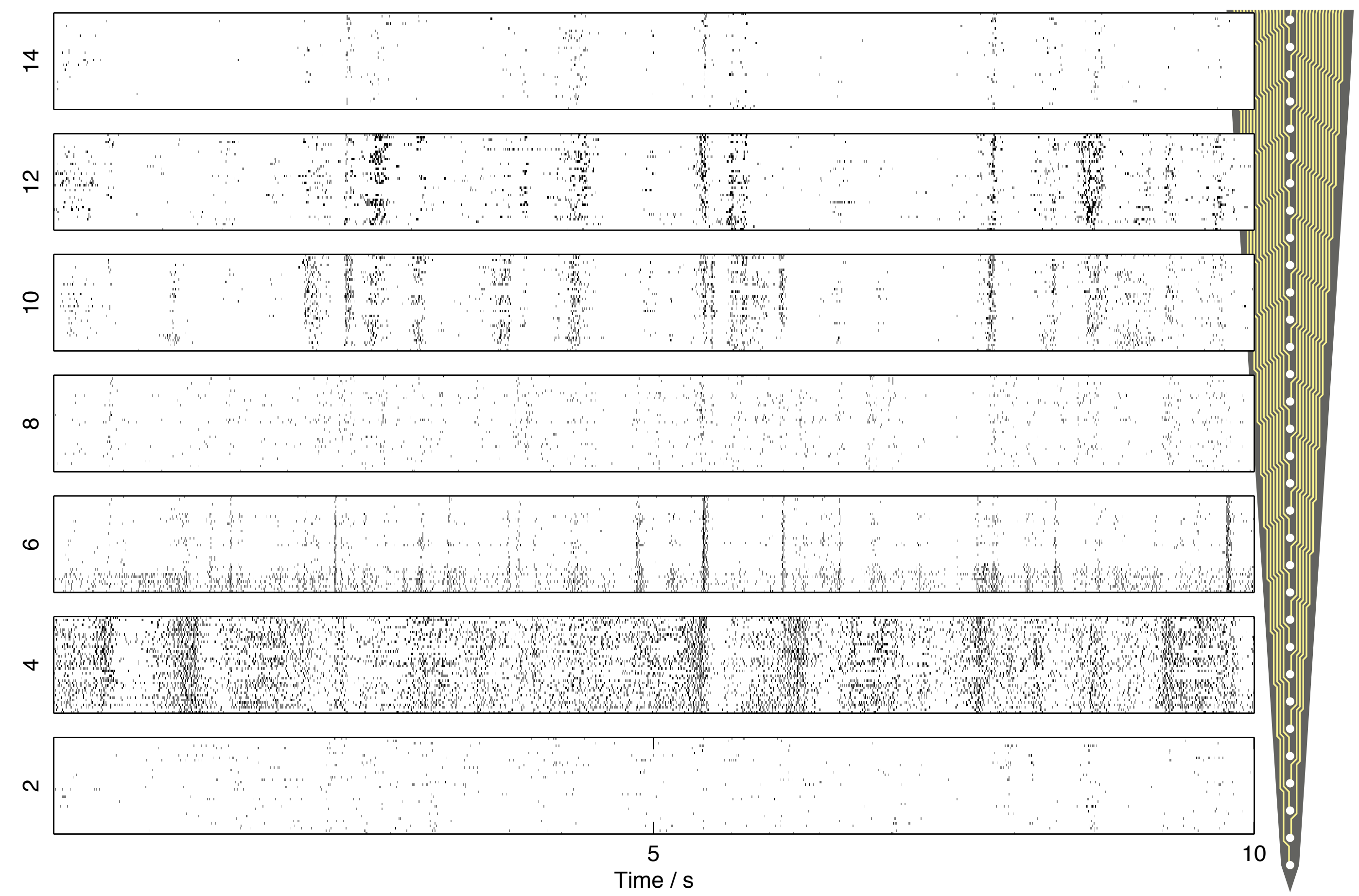
The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities
  - Exploratory data analysis

The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities
  - Exploratory data analysis

7 exemplar multiunits responding to 40 repeated trials of natural video in cat V1



[PLoS Comp Bio 2014] [Neuron 2013]

The Promise of Deep Unsupervised Learning

- Unknown features/labels
    - Novel modalities
    - Exploratory data analysis

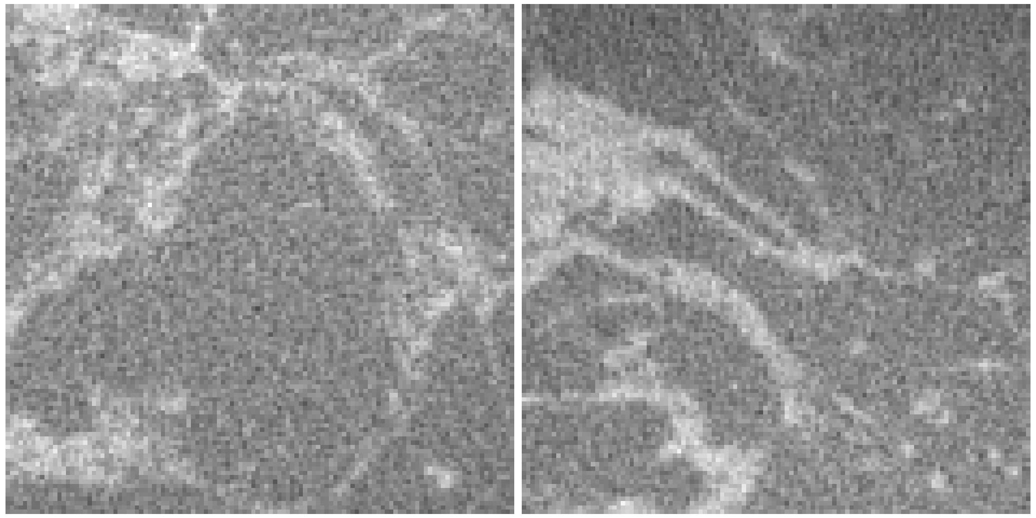
The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities
  - Exploratory data analysis
- Expensive labels

The Promise of Deep Unsupervised Learning

- Unknown features/labels
  - Novel modalities
  - Exploratory data analysis
- Expensive labels

Coronal breast CT



[SPIE 2009] [Med Phys 2014]

The Promise of Deep Unsupervised Learning

- Unknown features/labels
    - Novel modalities
    - Exploratory data analysis
- Expensive labels

The Promise of Deep Unsupervised Learning

- Unknown features/labels
    - Novel modalities
    - Exploratory data analysis
- Expensive labels
- Unpredictable tasks / one shot learning

Outline

- Motivation: The promise of deep unsupervised learning

- **Physical intuition: Diffusion processes and time reversal**

- Diffusion probabilistic model: Derivation and experimental results

- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Outline

- Motivation: The promise of deep unsupervised learning
- **Physical intuition: Diffusion processes and time reversal**
    - Destroy structure in data
    - Carefully characterize the destruction
    - Learn how to **reverse time**
- Diffusion probabilistic model: Derivation and experimental results
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Observation 1: Diffusion Destroys Structure

- Dye density represents probability density

Observation 1: Diffusion Destroys Structure

- Dye density represents probability density
- Goal: Learn structure of probability density

Observation 1: Diffusion Destroys Structure

- Dye density represents probability density
- Goal: Learn structure of probability density
- Observation: Diffusion destroys structure

Observation 1: Diffusion Destroys Structure

- Dye density represents probability density
- Goal: Learn structure of probability density
- Observation: Diffusion destroys structure

Observation 1: Diffusion Destroys Structure

- Dye density represents probability density
- Goal: Learn structure of probability density
- Observation: Diffusion destroys structure

Data distribution → Uniform distribution

Core Idea: Recover Structure by Reversing Time

- What if we could reverse time?

Core Idea: Recover Structure by Reversing Time

- What if we could reverse time?

Core Idea: Recover Structure by Reversing Time

- What if we could reverse time?

Data distribution ← Uniform distribution

Core Idea: Recover Structure by Reversing Time

- What if we could reverse time?
- Recover data distribution by starting from uniform distribution and running dynamics backwards

Data distribution ← Uniform distribution

Core Idea: Recover Structure by Reversing Time

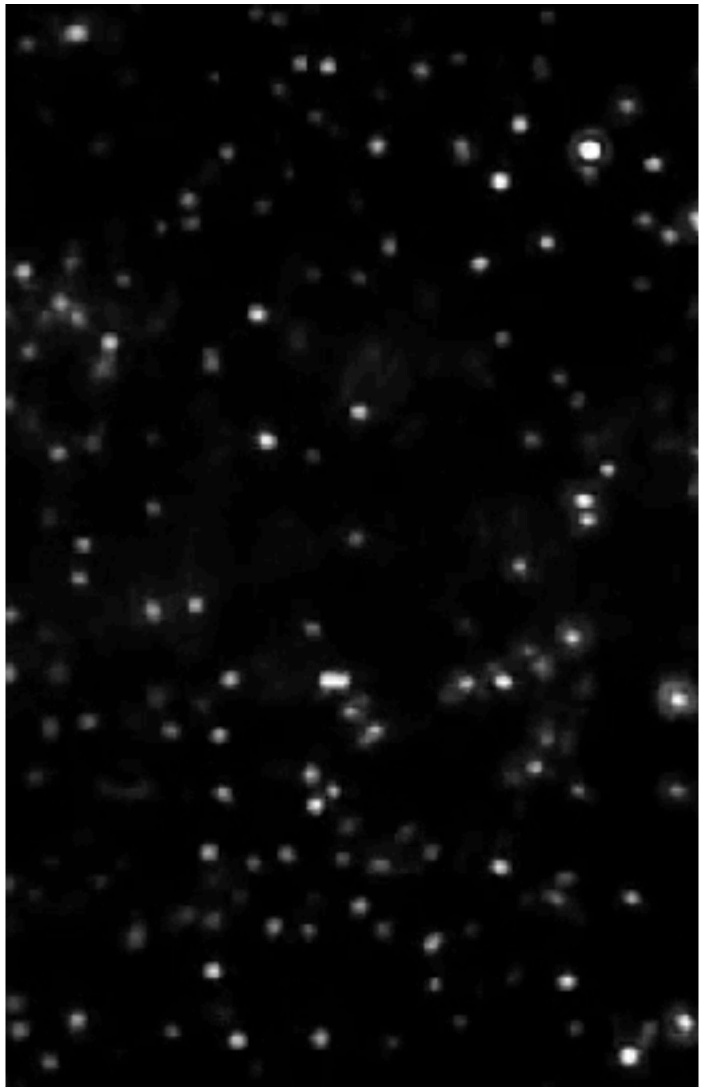
- What if we could reverse time?
- Recover data distribution by starting from uniform distribution and running dynamics backwards

Data distribution ← Uniform distribution

Core Idea: Recover Structure by Reversing Time

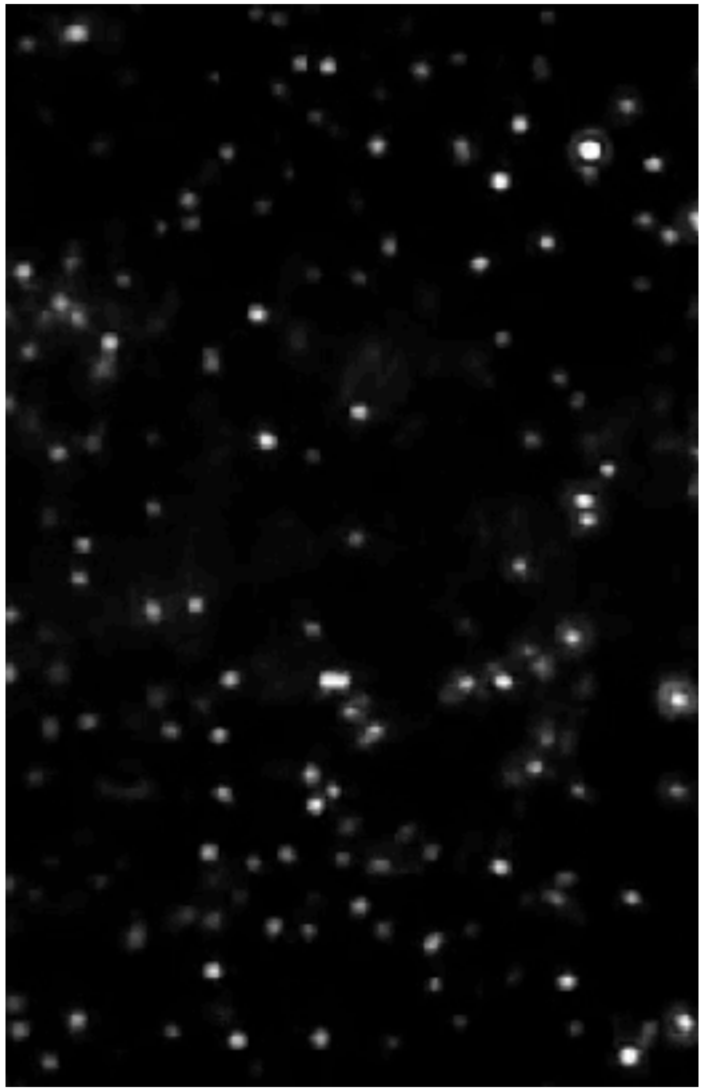
Core Idea: Recover Structure by Reversing Time

Observation 2: Microscopic Diffusion is Time Reversible



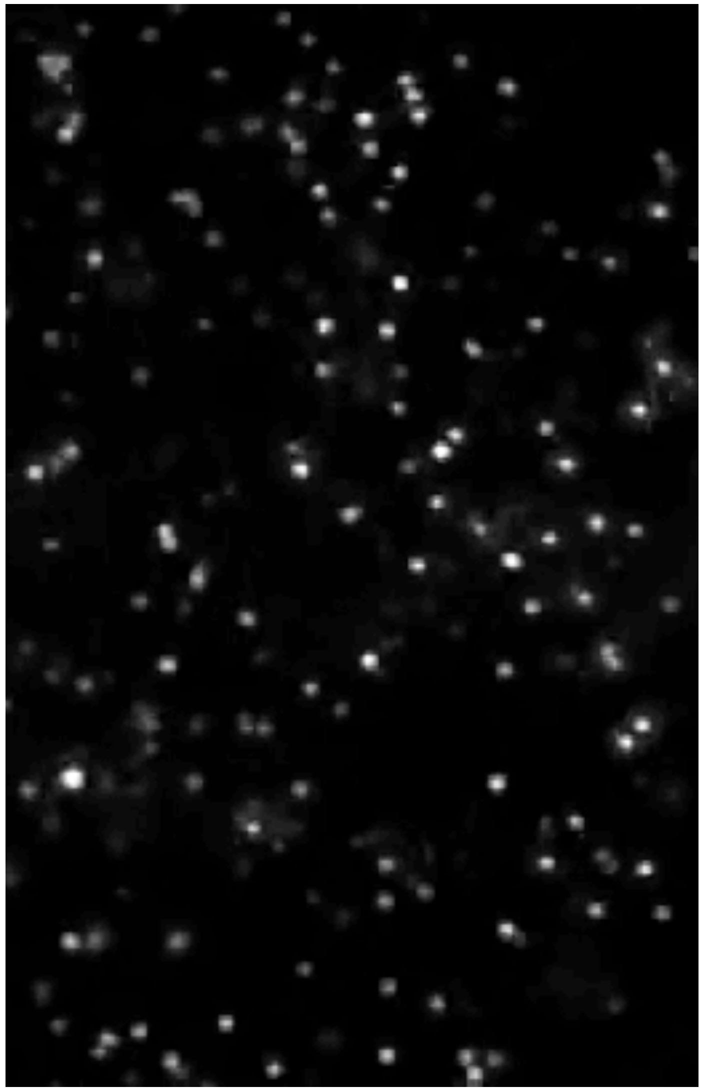
- Microscopic view
- Brownian motion

Observation 2: Microscopic Diffusion is Time Reversible



- Microscopic view
- Brownian motion

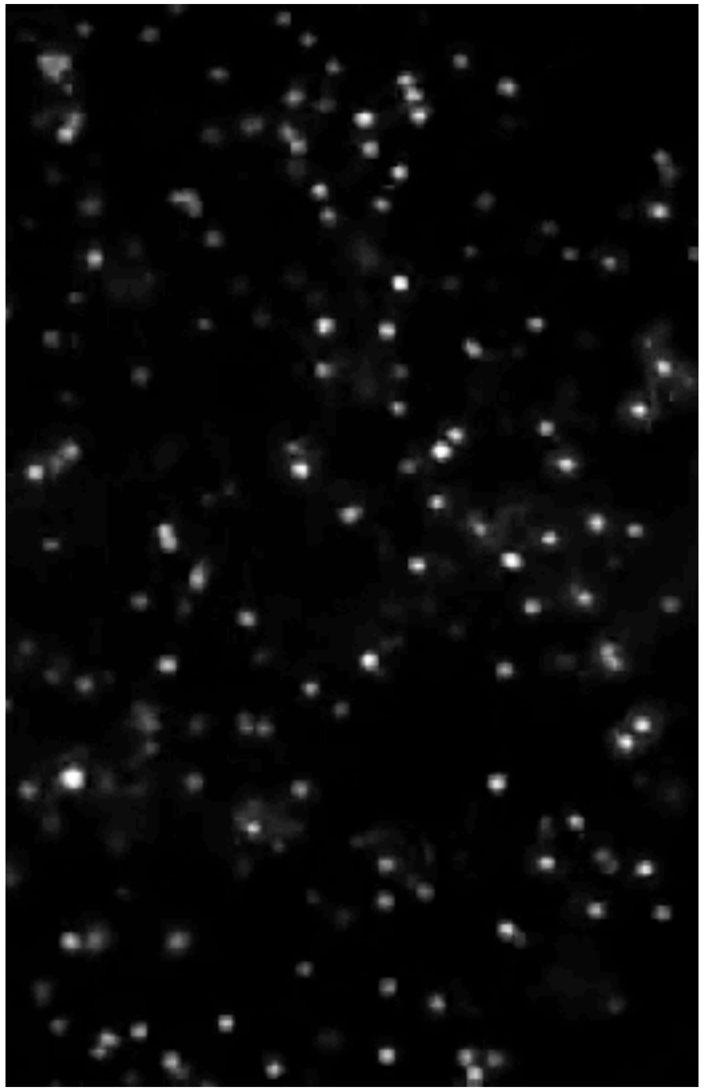
Observation 2: Microscopic Diffusion is Time Reversible



© Rutger Saly

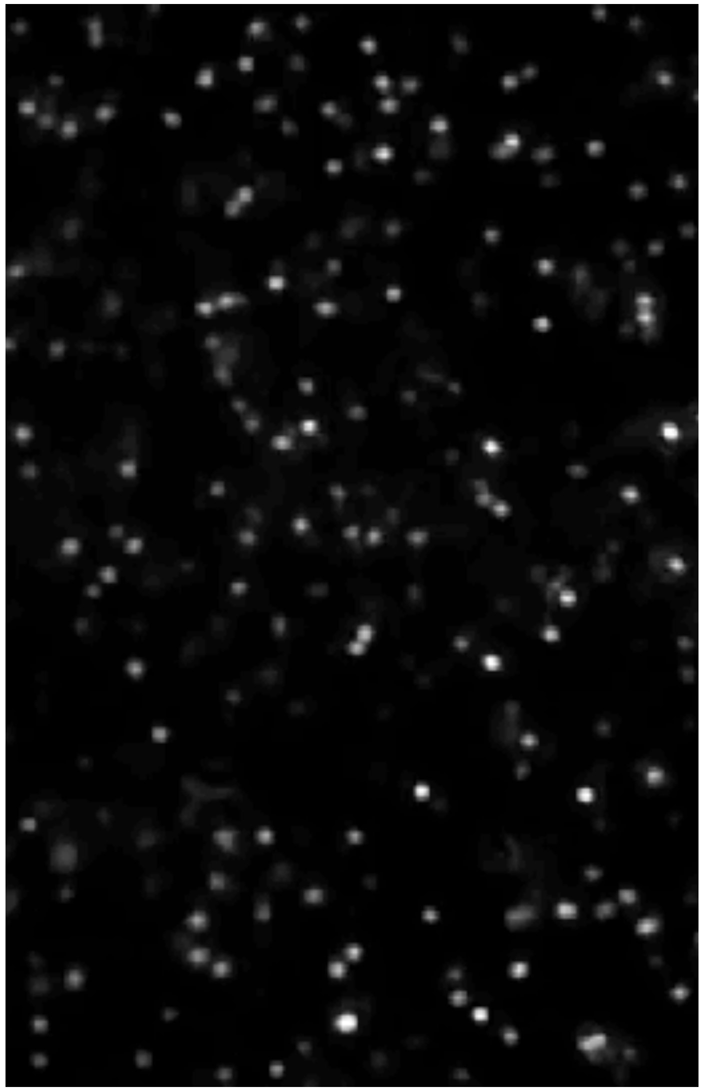
- Microscopic view
- Brownian motion

Observation 2: Microscopic Diffusion is Time Reversible



- Microscopic view
- Brownian motion

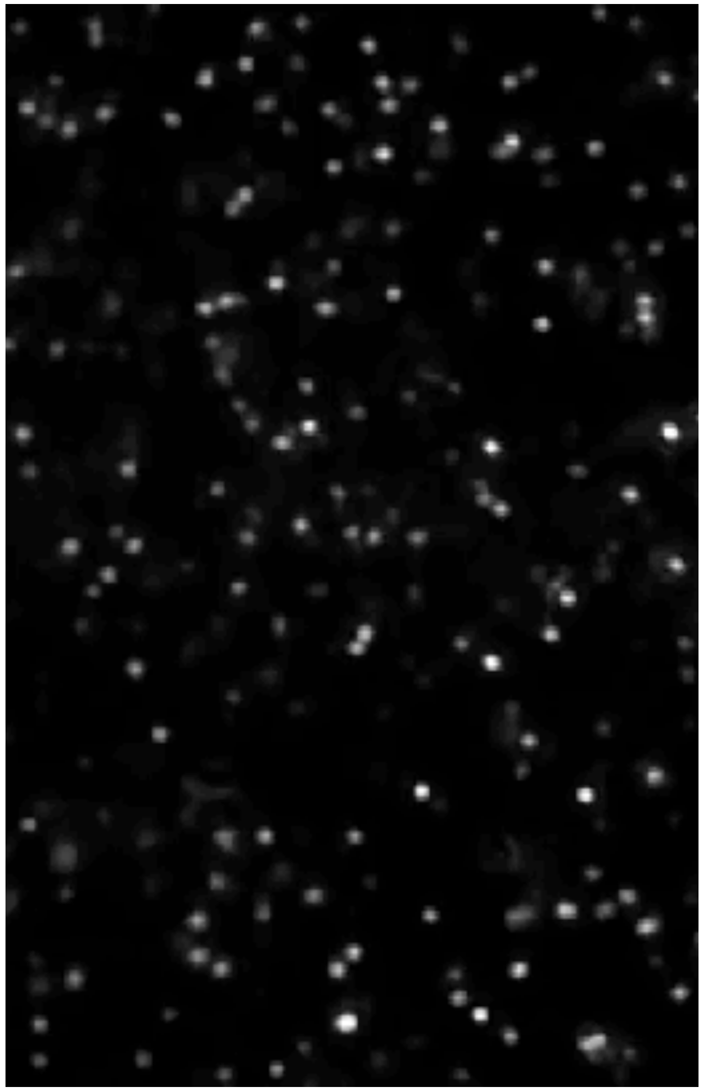
Observation 2: Microscopic Diffusion is Time Reversible



© Rutger Saly

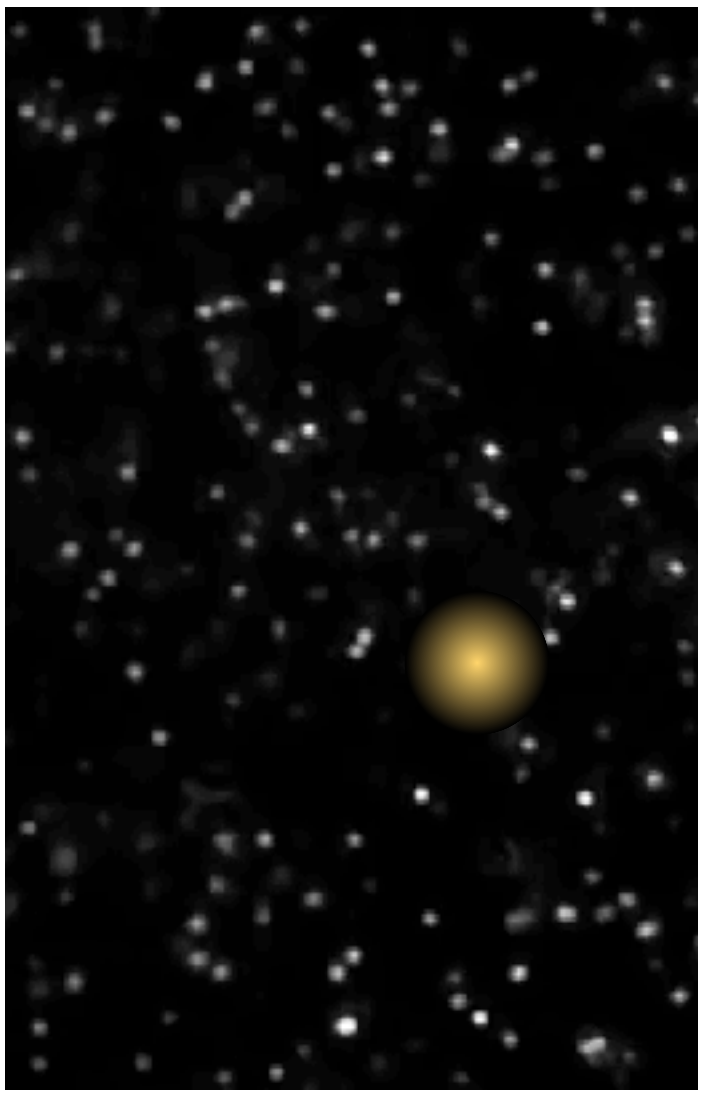
- Microscopic view
- Brownian motion
- Position updates are small Gaussians

Observation 2: Microscopic Diffusion is Time Reversible



- Microscopic view
- Brownian motion
- Position updates are small Gaussians

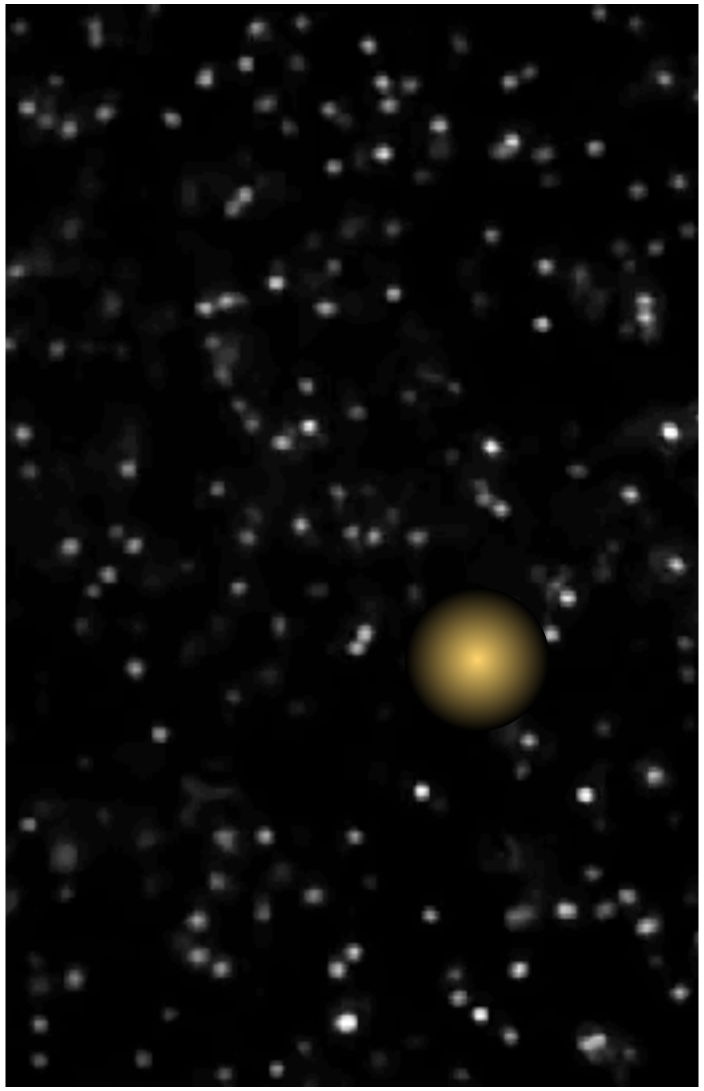
Observation 2: Microscopic Diffusion is Time Reversible



© Rutger Saly

- Microscopic view
- Brownian motion
- Position updates are small Gaussians

Observation 2: Microscopic Diffusion is Time Reversible



© Rutger Saly

- Microscopic view
- Brownian motion
- Position updates are small Gaussians
    - Both forwards and backwards in time

Overview of Diffusion-based Probabilistic Models

Overview of Diffusion-based Probabilistic Models

- Destroy all structure in data distribution using diffusion process

Overview of Diffusion-based Probabilistic Models

- Destroy all structure in data distribution using diffusion process
- Learn reversal of diffusion process
    - Estimate function for mean and covariance of each step in the reverse diffusion process (binomial rate for binary data)

Overview of Diffusion-based Probabilistic Models

- Destroy all structure in data distribution using diffusion process
- Learn reversal of diffusion process
    - Estimate function for mean and covariance of each step in the reverse diffusion process (binomial rate for binary data)
- Reverse diffusion process is the model of the data

Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- **Diffusion probabilistic model: Derivation and experimental results**
    - Algorithm
    - Deep convolutional network: Universal function approximator
    - Multiplying distributions: Inputation, denoising, computing posteriors
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
  - **Algorithm**
  - Deep convolutional network: Universal function approximator
  - Multiplying distributions: Inputation, denoising, computing posteriors
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Destroy All Structure in Data using Diffusion Process

Data
distribution

$$q\left(\mathbf{x}^{(0)}\right)$$

Destroy All Structure in Data using Diffusion Process

Data distribution

Forward diffusion

$$q\left(\mathbf{x}^{(0)}\right)$$

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = \mathcal{N}\left(\mathbf{x}^{(t)}; \mathbf{x}^{(t-1)}\sqrt{1-\beta_t}, \mathbf{I}\beta_t\right)$$

Destroy All Structure in Data using Diffusion Process

Data distribution

Forward diffusion

$$q\left(\mathbf{x}^{(0)}\right) \longrightarrow$$

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = \mathcal{N}\left(\mathbf{x}^{(t)}; \underline{\mathbf{x}^{(t-1)}\sqrt{1-\beta_t}}, \mathbf{I}\beta_t\right)$$

Decay towards origin

Destroy All Structure in Data using Diffusion Process

Data distribution

Forward diffusion

$$q\left(\mathbf{x}^{(0)}\right)$$

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = \mathcal{N}\left(\mathbf{x}^{(t)}; \mathbf{x}^{(t-1)}\sqrt{1-\beta_t}, \mathbf{I}\beta_t\right)$$

Decay towards origin

Add small noise

Destroy All Structure in Data using Diffusion Process

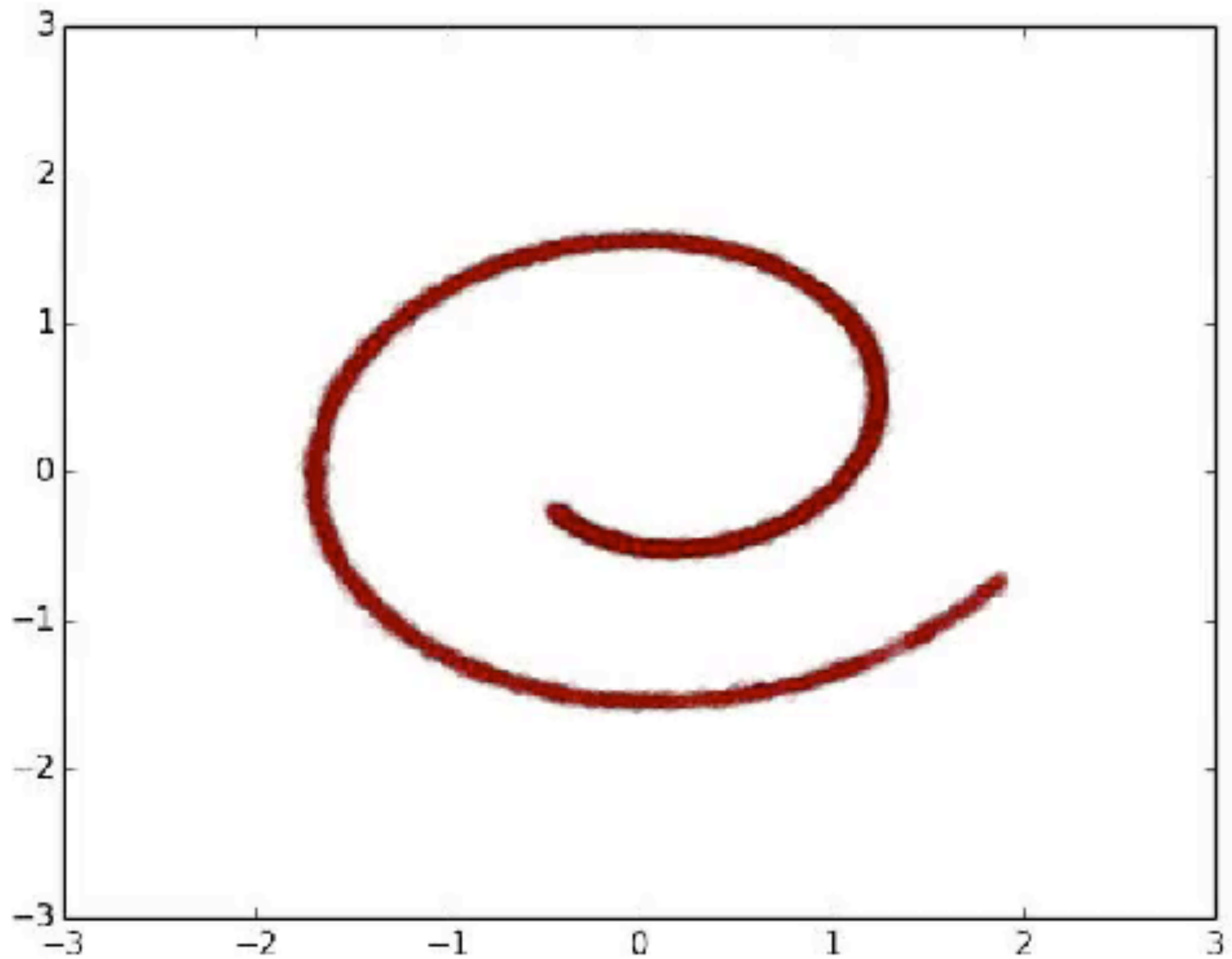
Data distribution	Forward diffusion	Noise distribution
$q\left(\mathbf{x}^{(0)}\right)$	→	$q\left(\mathbf{x}^{(T)}\right) \approx \mathcal{N}\left(\mathbf{x}^{(T)}; 0, \mathbf{I}\right)$

$$q\left(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}\right) = \mathcal{N}\left(\mathbf{x}^{(t)}; \underline{\mathbf{x}^{(t-1)}\sqrt{1-\beta_t}}, \underline{\mathbf{I}\beta_t}\right)$$

Decay towards origin — Add small noise

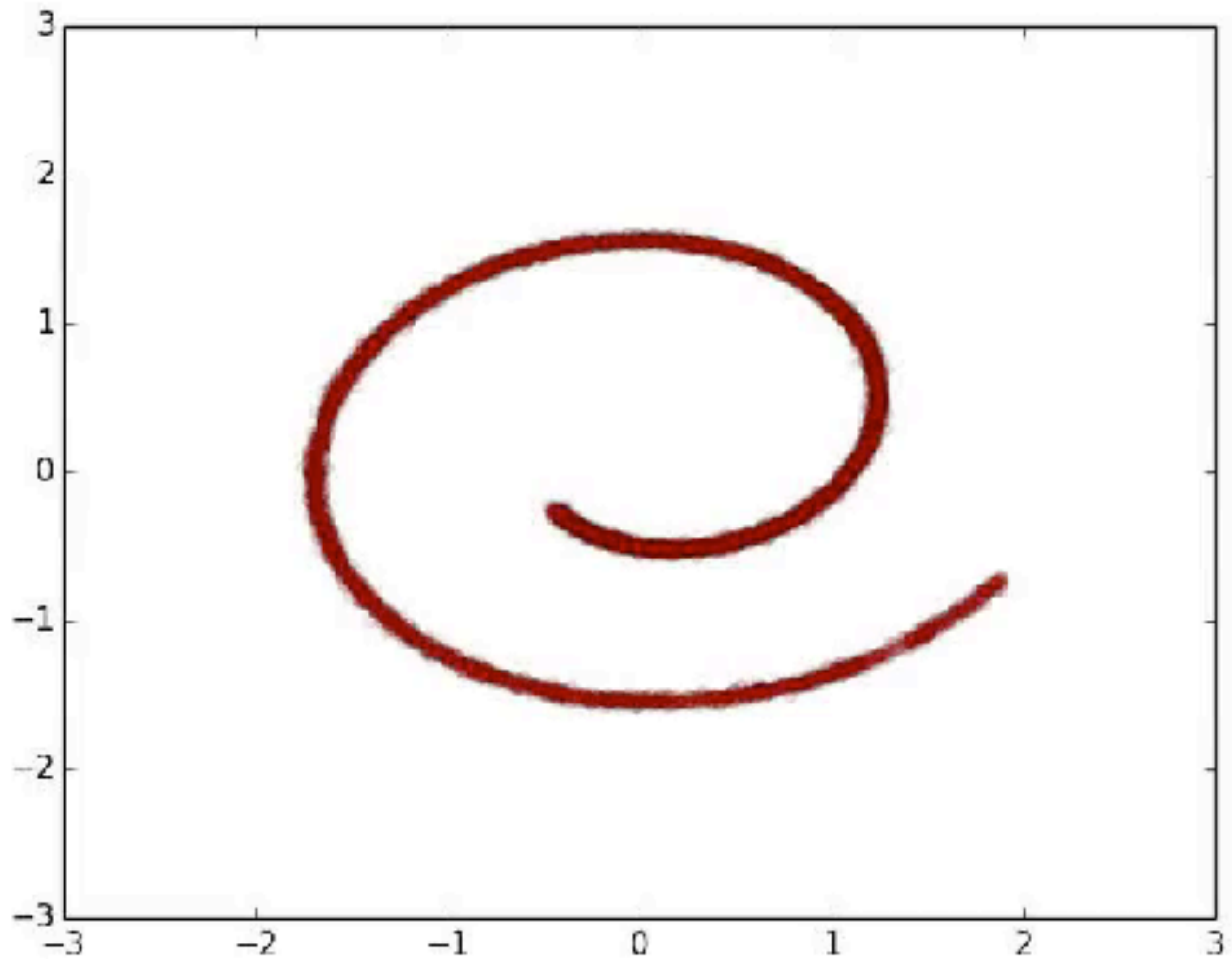
Forward Diffusion Process on Swiss Roll

Data → Gaussian diffusion in quadratic well → Isotropic Gaussian



Forward Diffusion Process on Swiss Roll

Data → Gaussian diffusion in quadratic well → Isotropic Gaussian



Recover Structure in Data using Reversal of Diffusion Process

Noise distribution

$$p\left(\mathbf{x}^{(T)}\right) = \mathcal{N}\left(\mathbf{x}^{(T)}; 0, \mathbf{I}\right)$$

Recover Structure in Data using Reversal of Diffusion Process

Reverse diffusion

Noise distribution

$$p\left(\mathbf{x}^{(T)}\right) = \mathcal{N}\left(\mathbf{x}^{(T)}; 0, \mathbf{I}\right)$$

$$p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)\right)$$

Recover Structure in Data using Reversal of Diffusion Process

Reverse diffusion

Noise distribution

$$p\left(\mathbf{x}^{(T)}\right) = \mathcal{N}\left(\mathbf{x}^{(T)}; 0, \mathbf{I}\right)$$

$$p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_\mu\left(\mathbf{x}^{(t)}, t\right), f_\Sigma\left(\mathbf{x}^{(t)}, t\right)\right)$$

Learned drift and covariance functions

Recover Structure in Data using Reversal of Diffusion Process

Data distribution

Reverse diffusion

Noise distribution

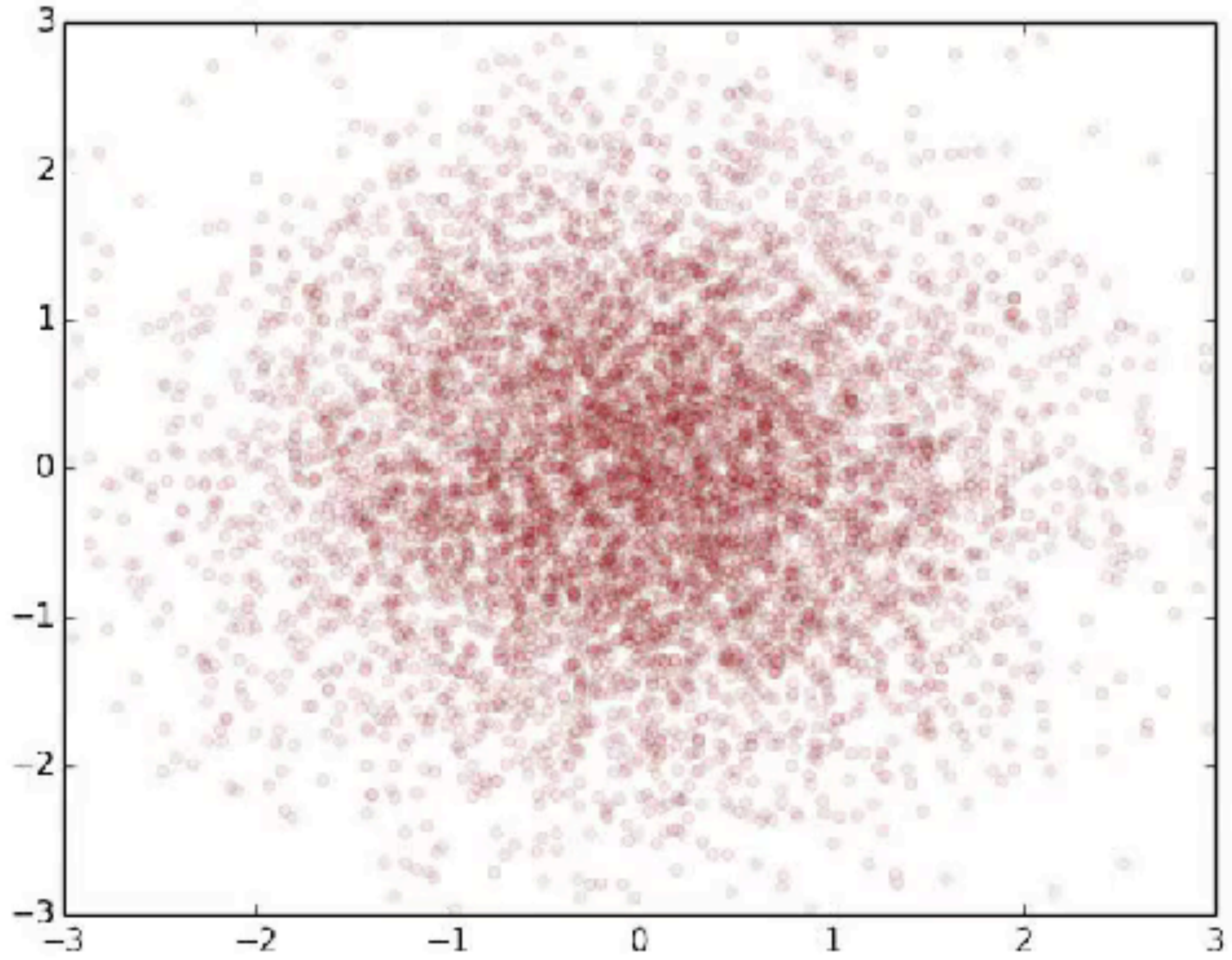
$$p\left(\mathbf{x}^{(0)}\right) \approx q\left(\mathbf{x}^{(0)}\right) \longleftarrow p\left(\mathbf{x}^{(T)}\right) = \mathcal{N}\left(\mathbf{x}^{(T)}; 0, \mathbf{I}\right)$$

$$p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_\mu\left(\mathbf{x}^{(t)}, t\right), f_\Sigma\left(\mathbf{x}^{(t)}, t\right)\right)$$

Learned drift and covariance functions

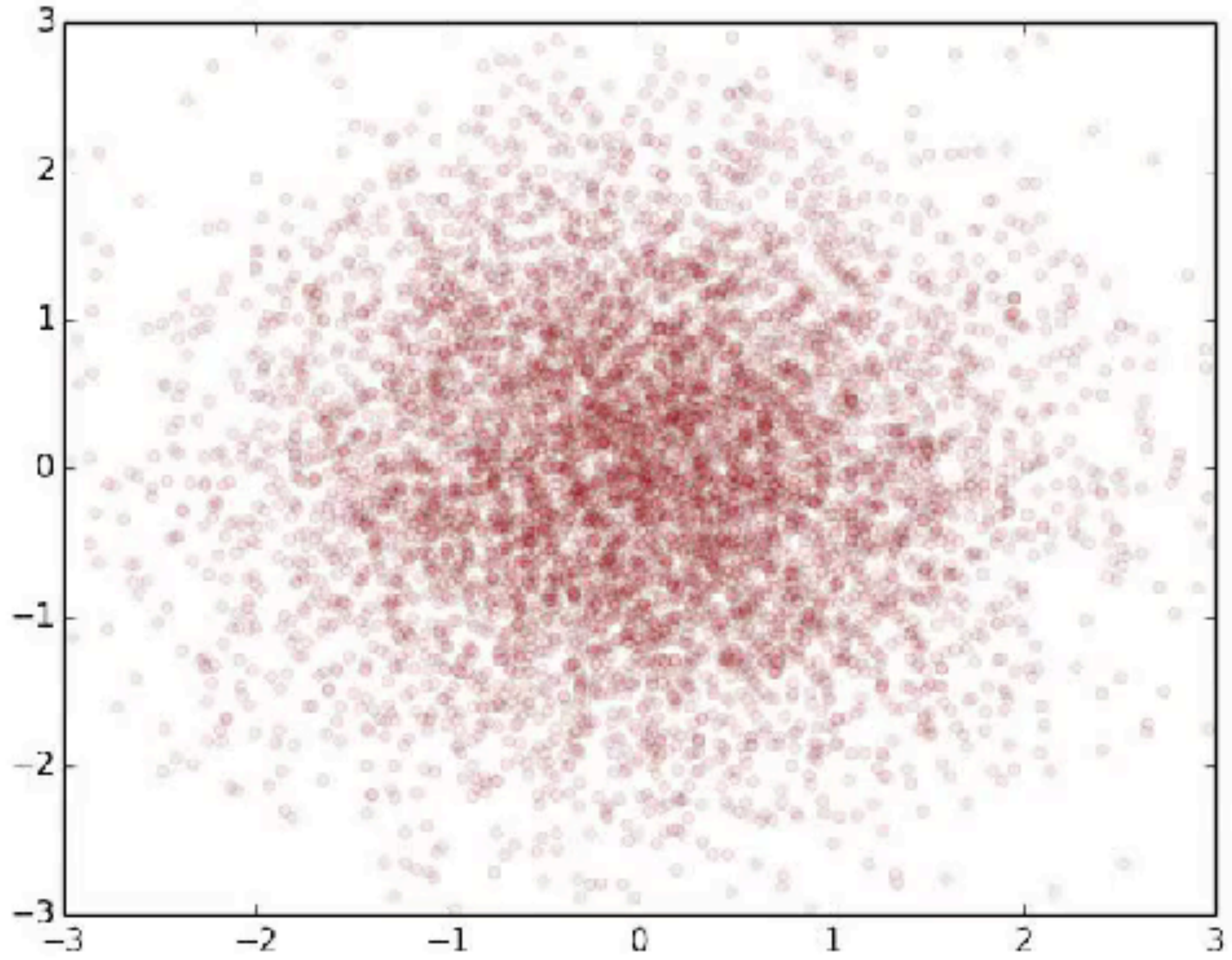
Learned Reverse Diffusion Process on Swiss Roll

Data dist. ← Gaussian diffusion w/ learned kernel ← Isotropic Gaussian

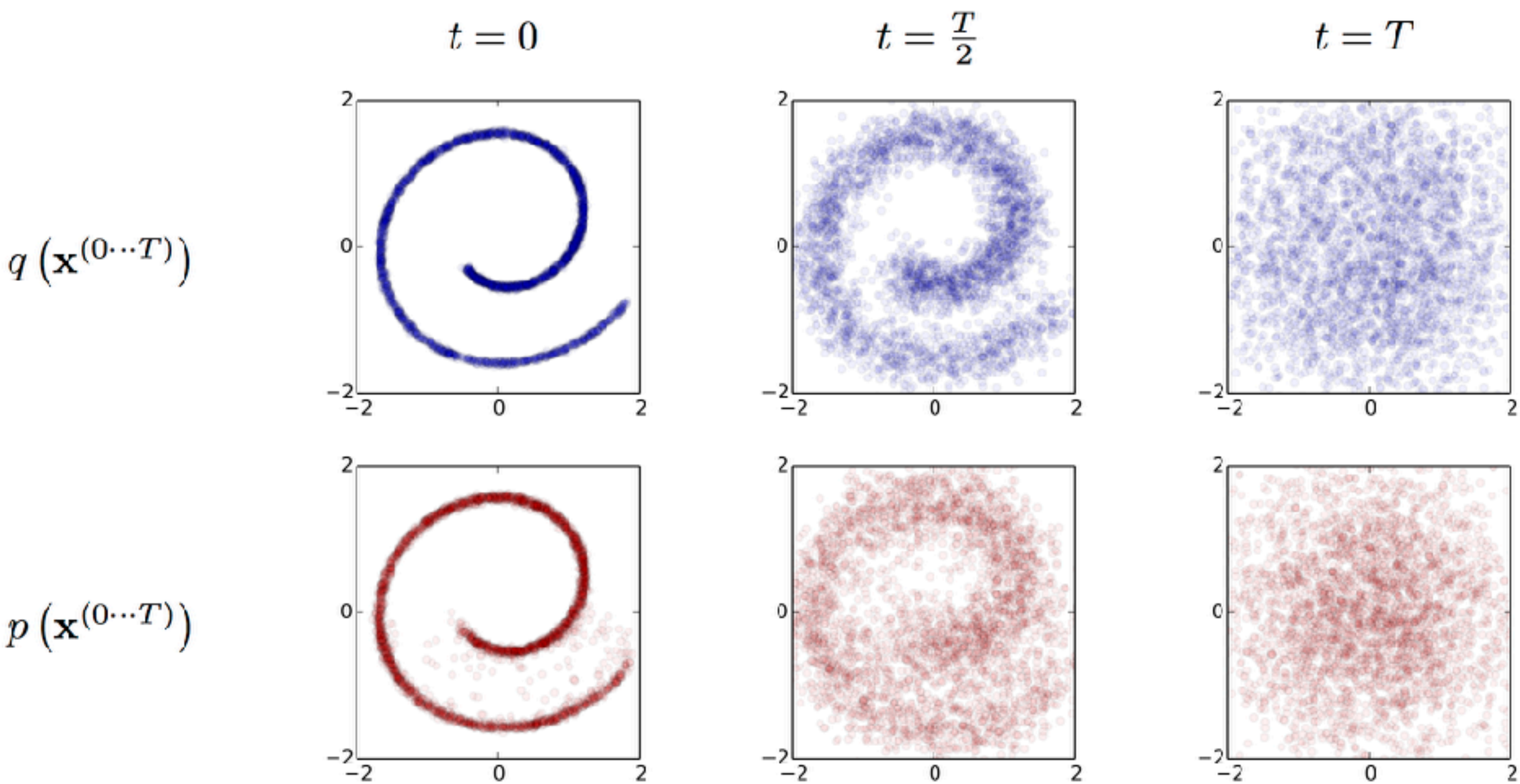


Learned Reverse Diffusion Process on Swiss Roll

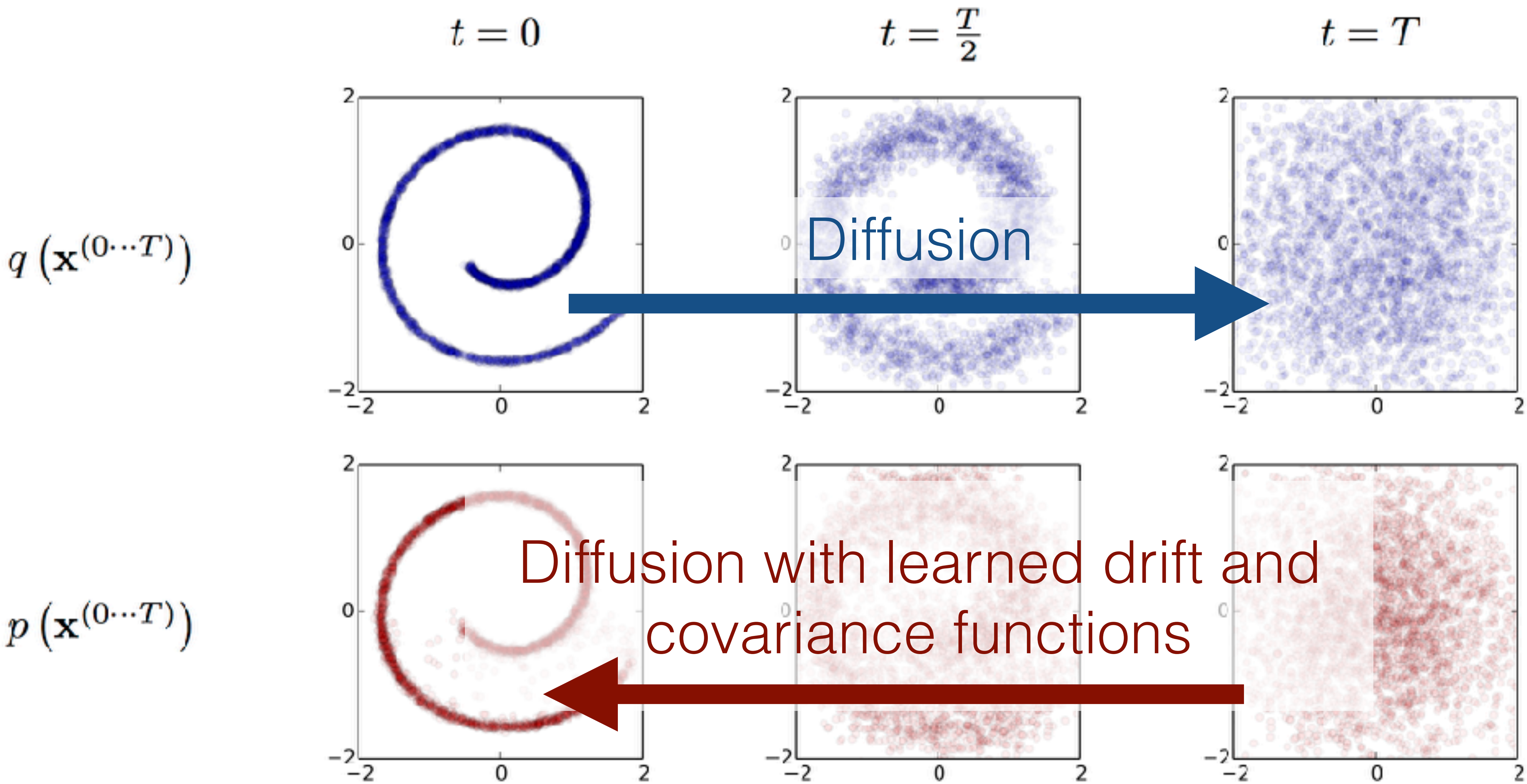
Data dist. ← Gaussian diffusion w/ learned kernel ← Isotropic Gaussian



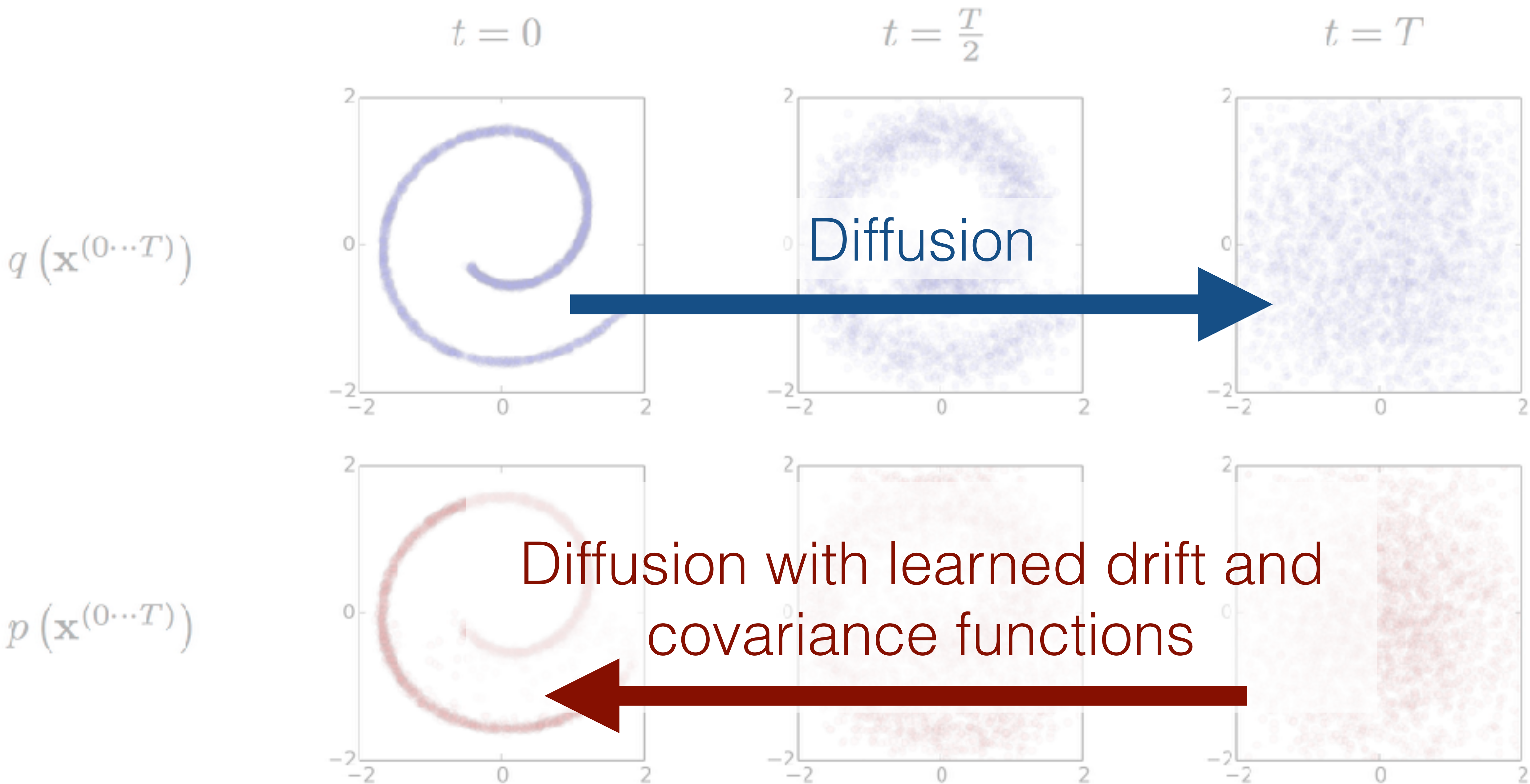
Summary of Forward and Reverse Diffusion on Swiss Roll



Summary of Forward and Reverse Diffusion on Swiss Roll



Summary of Forward and Reverse Diffusion on Swiss Roll



Training the Reverse Diffusion Process

Model probability

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} p\left(\mathbf{x}^{(0\cdots T)}\right)$$

Training the Reverse Diffusion Process

Model probability

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} p\left(\mathbf{x}^{(0\cdots T)}\right)$$

Annealed importance sampling / Jarzynski equality

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}$$

Training the Reverse Diffusion Process

Model probability

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} p\left(\mathbf{x}^{(0\cdots T)}\right)$$

Annealed importance sampling / Jarzynski equality

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}$$

Log Likelihood

$$L = \int d\mathbf{x}^{(0)} q\left(\mathbf{x}^{(0)}\right) \log\left[\int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}\right)}\right]$$

Model probability

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} p\left(\mathbf{x}^{(0\cdots T)}\right)$$

Annealed importance sampling / Jarzynski equality

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}$$

Log Likelihood

$$L = \int d\mathbf{x}^{(0)} q\left(\mathbf{x}^{(0)}\right) \log\left[\int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}\right)}\right]$$

Jensen's inequality

$$L \geq \int d\mathbf{x}^{(0\cdots T)} q\left(\mathbf{x}^{(0\cdots T)}\right) \log\left[\frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}\right]$$

$$p\left(\mathbf{x}^{(0)}\right) = \int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}$$

Log Likelihood

$$L = \int d\mathbf{x}^{(0)} q\left(\mathbf{x}^{(0)}\right) \log\left[\int d\mathbf{x}^{(1\cdots T)} q\left(\mathbf{x}^{(1\cdots T)}\right) \frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}\right)}\right]$$

Jensen's inequality

$$L \geq \int d\mathbf{x}^{(0\cdots T)} q\left(\mathbf{x}^{(0\cdots T)}\right) \log\left[\frac{p\left(\mathbf{x}^{(0\cdots T)}\right)}{q\left(\mathbf{x}^{(1\cdots T)}|\mathbf{x}^{(0)}\right)}\right]$$

… algebra …

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

$$L \geq - \sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

Gaussian

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

Gaussian

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

Gaussian

$$p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)\right)$$

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)$$

$$+ \text{const}$$

Gaussian

$$p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)\right)$$

Training

$$\underset{f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)}{\operatorname{argmin}} \mathbb{E}\left[D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)\right]$$

$$L \geq -\sum_{t=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(t)} q\left(\mathbf{x}^{(0)}, \mathbf{x}^{(t)}\right) D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right) + \text{const}$$

Gaussian

$$p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)\right)$$

Training

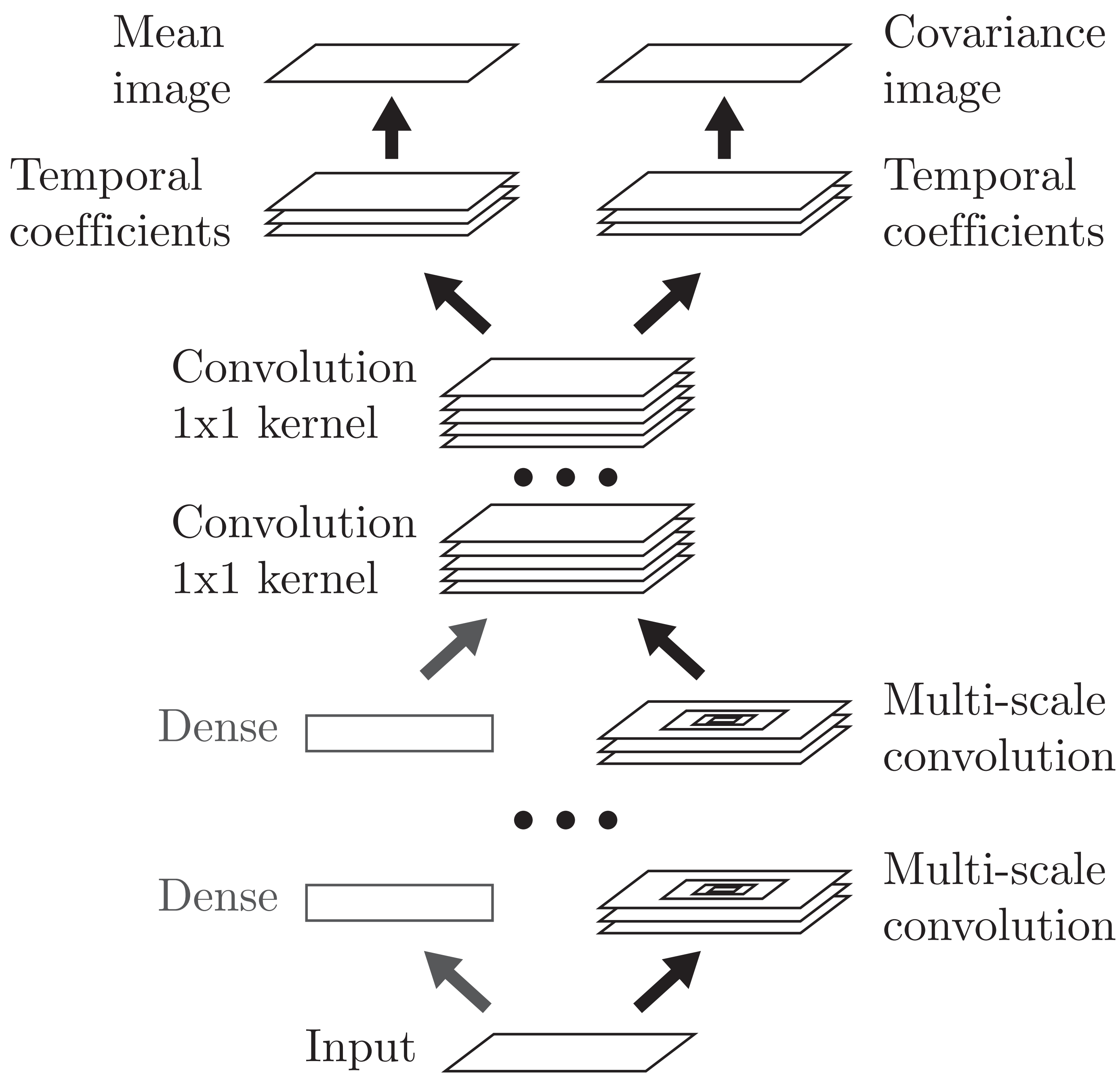
$$\underset{f_{\mu}\left(\mathbf{x}^{(t)}, t\right), f_{\Sigma}\left(\mathbf{x}^{(t)}, t\right)}{\operatorname{argmin}} \mathbb{E}\left[D_{KL}\left(q\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}, \mathbf{x}^{(0)}\right) \middle|\middle| p\left(\mathbf{x}^{(t-1)} | \mathbf{x}^{(t)}\right)\right)\right]$$

Unsupervised learning → Regression

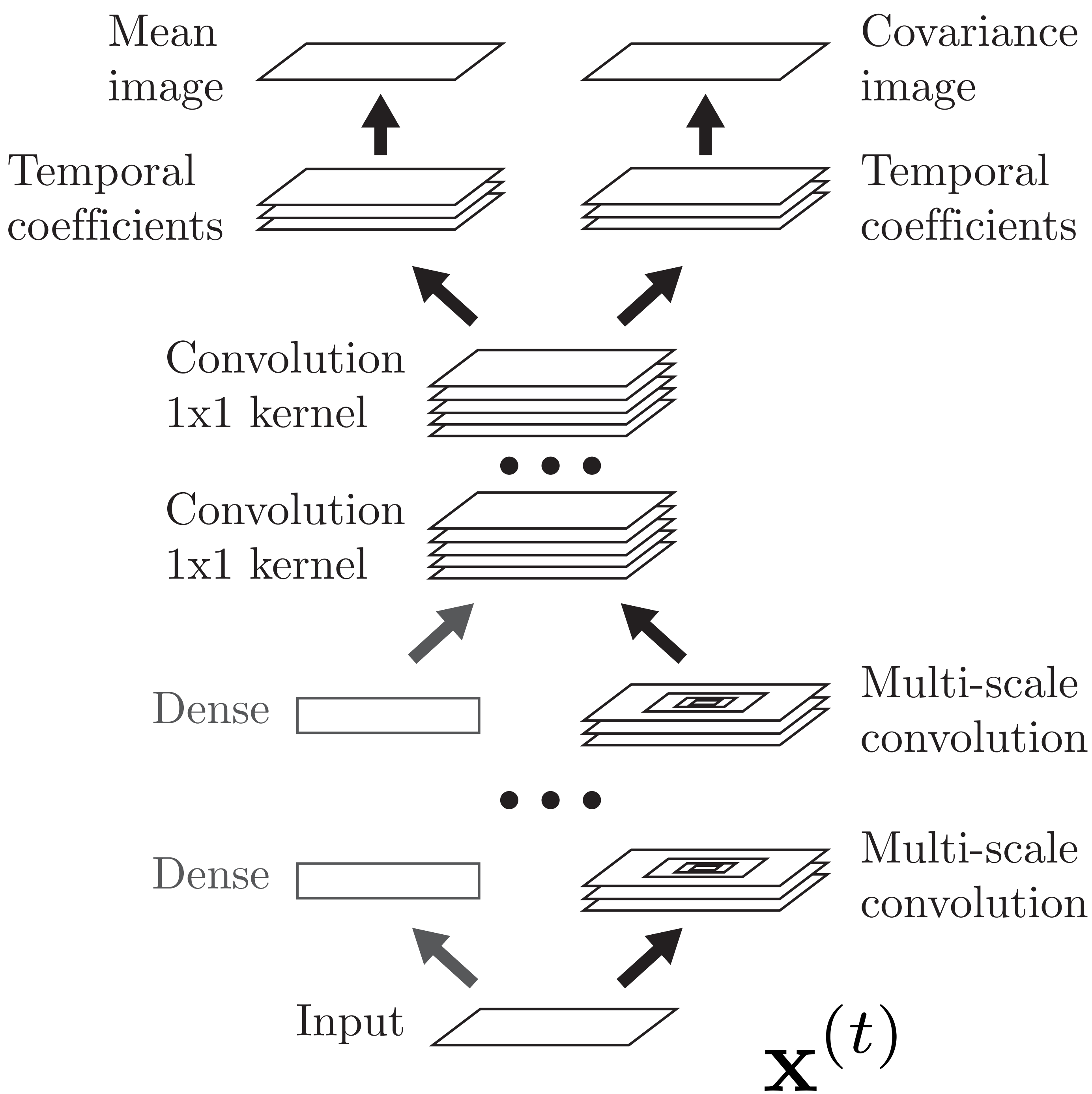
Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
    - Algorithm
    - **Deep convolutional network: Universal function approximator**
    - Multiplying distributions: Inputation, denoising, computing posteriors
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Use Deep Network as Function Approximator for Images

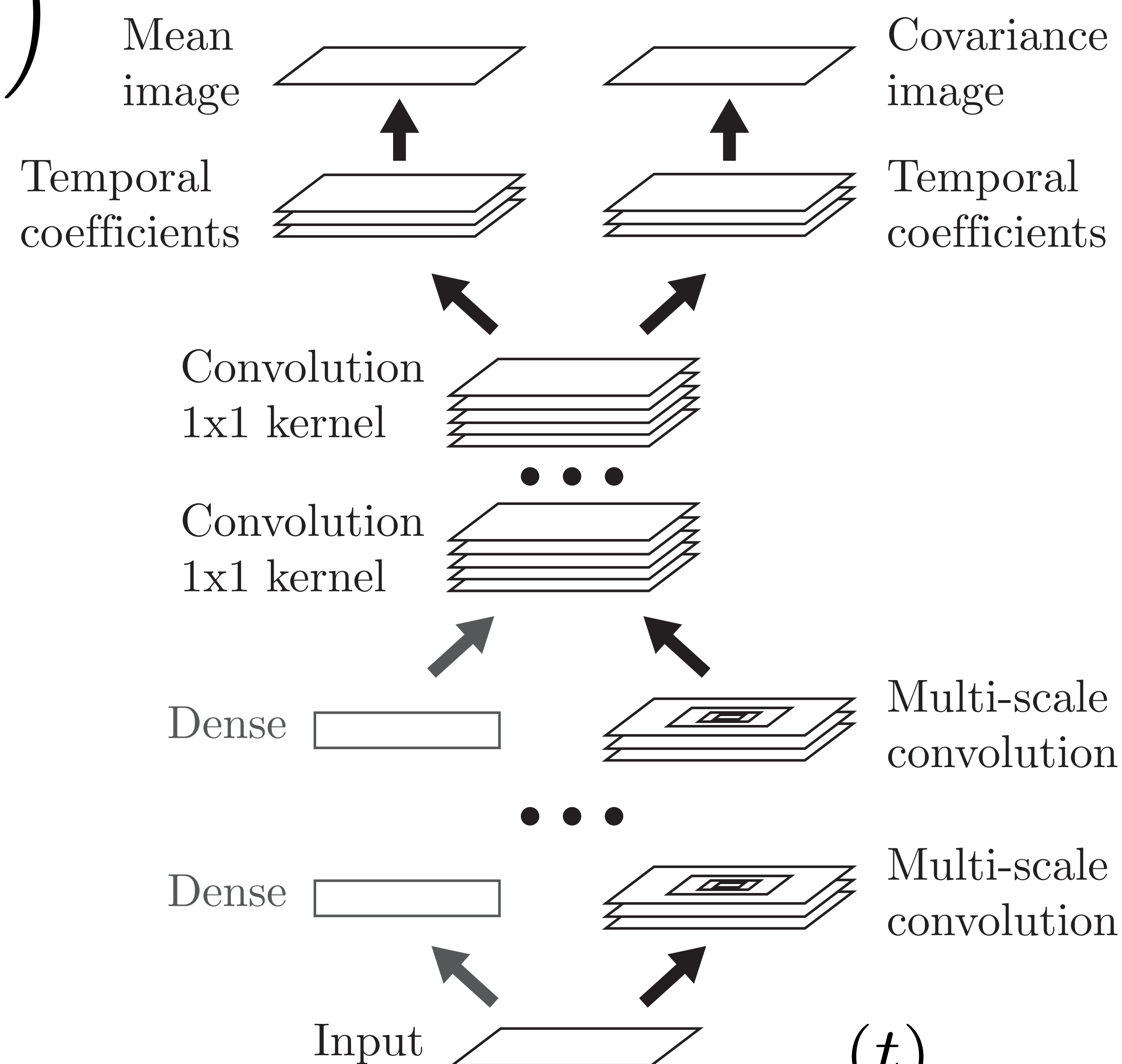


Use Deep Network as Function Approximator for Images



Use Deep Network as Function Approximator for Images

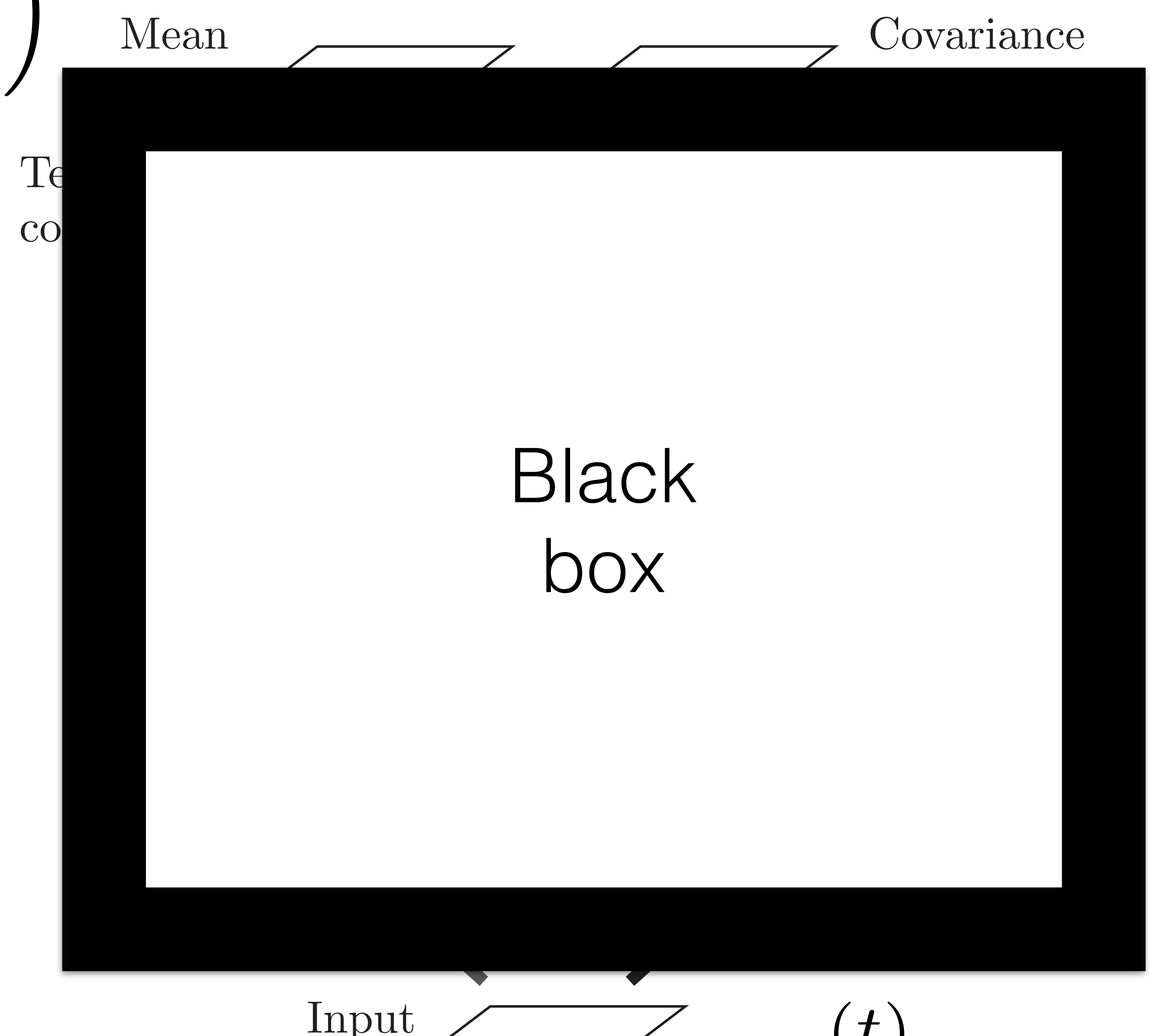
$$f_\mu\left(\mathbf{x}^{(t)}, t\right) \qquad f_\Sigma\left(\mathbf{x}^{(t)}, t\right)$$



$$\mathbf{x}^{(t)}$$

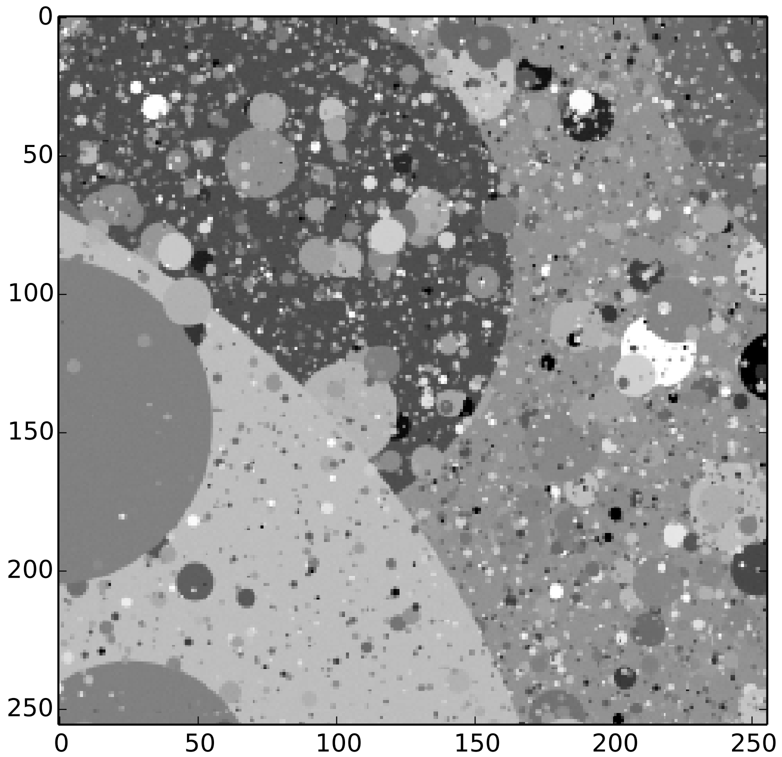
Use Deep Network as Function Approximator for Images

$$f_\mu\left(\mathbf{x}^{(t)}, t\right) \qquad f_\Sigma\left(\mathbf{x}^{(t)}, t\right)$$



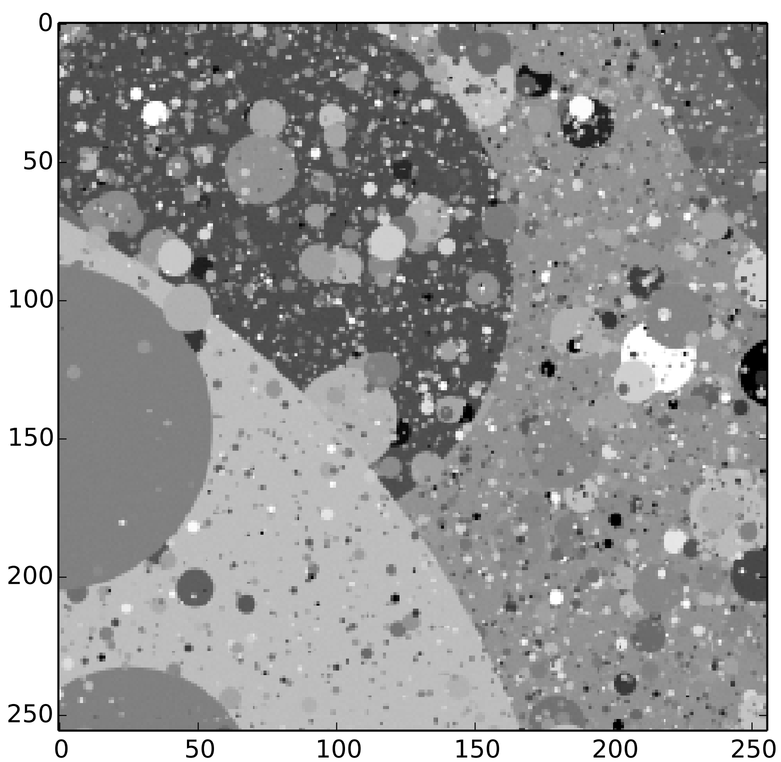
$$\mathbf{x}^{(t)}$$

Diffusion Probabilistic Model Applied to Dead Leaves

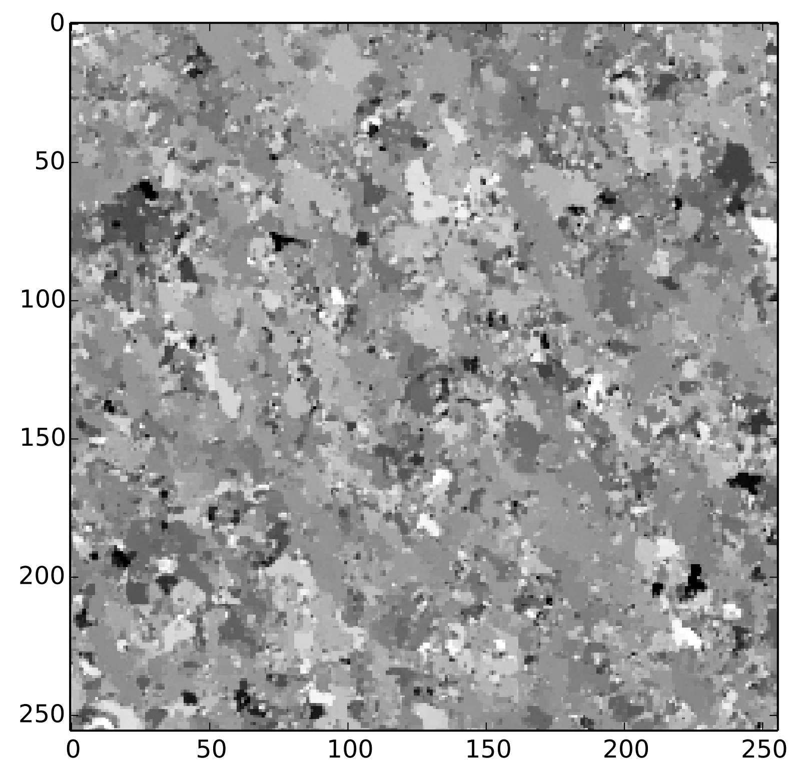


Training Data

Diffusion Probabilistic Model Applied to Dead Leaves

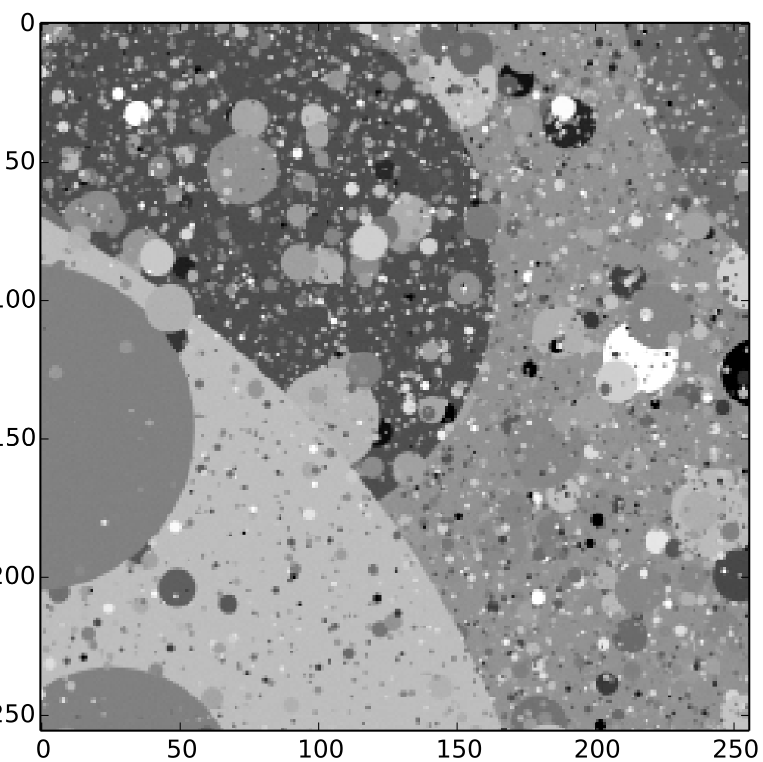


Training Data

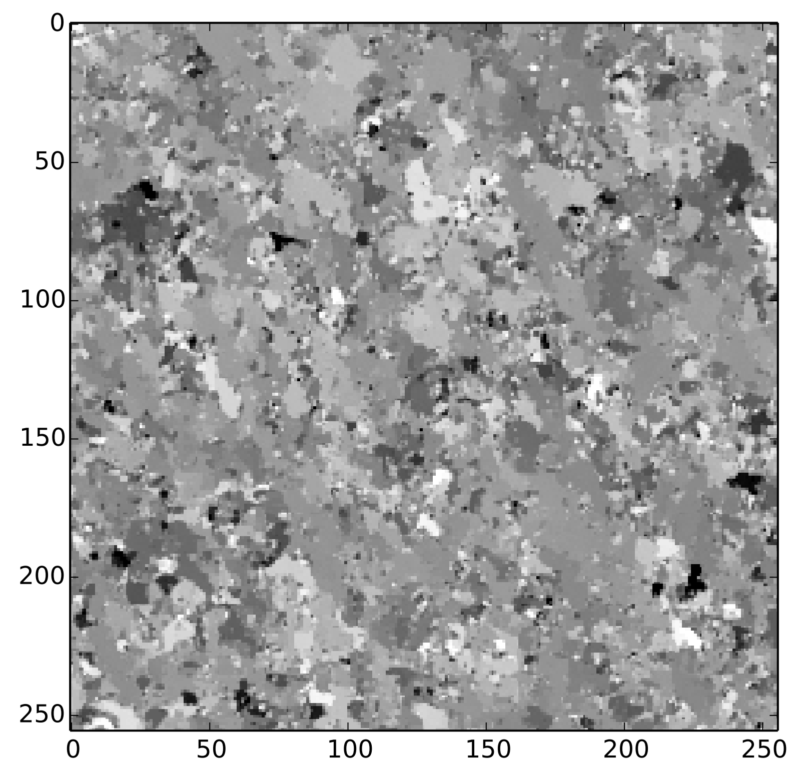


Sample from
[Theis *et al*, 2012]

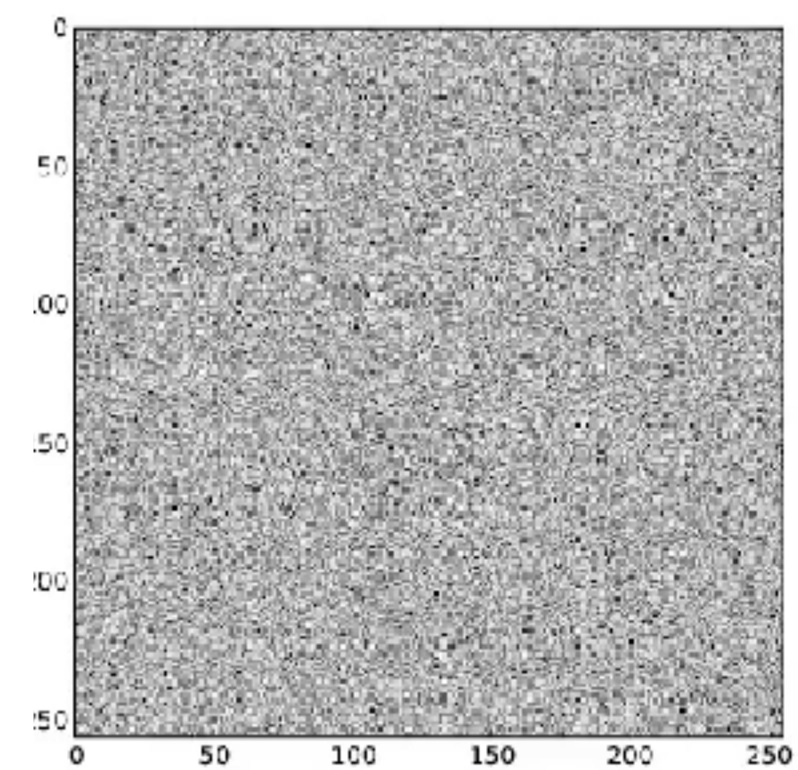
Diffusion Probabilistic Model Applied to Dead Leaves



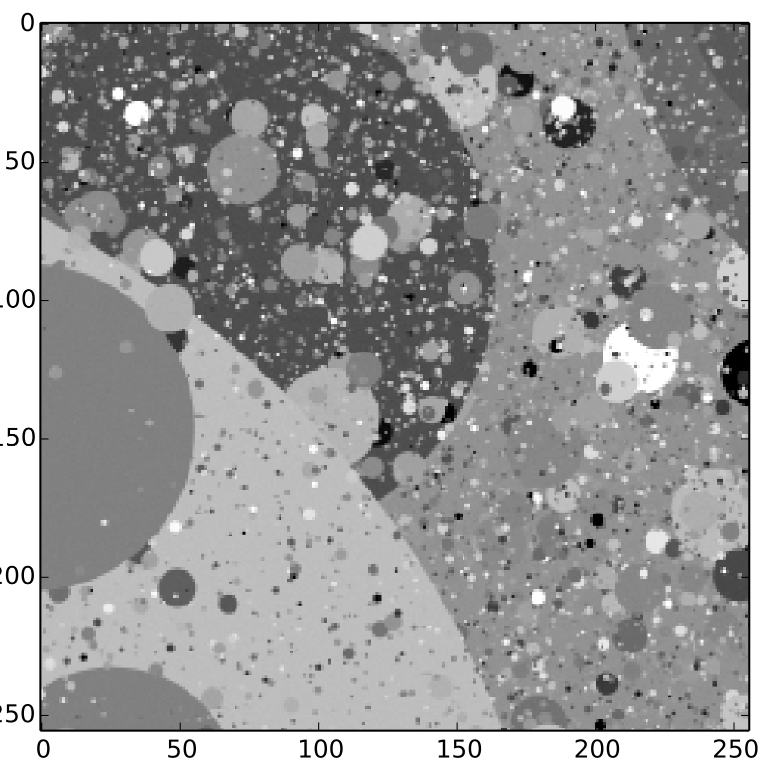
Training Data



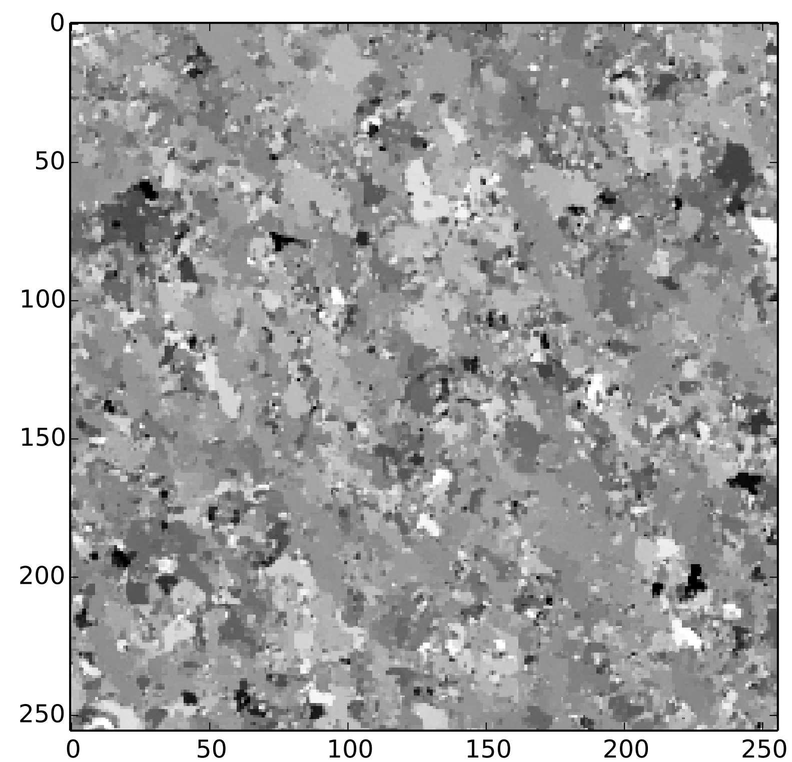
Sample from
[Theis *et al*, 2012]



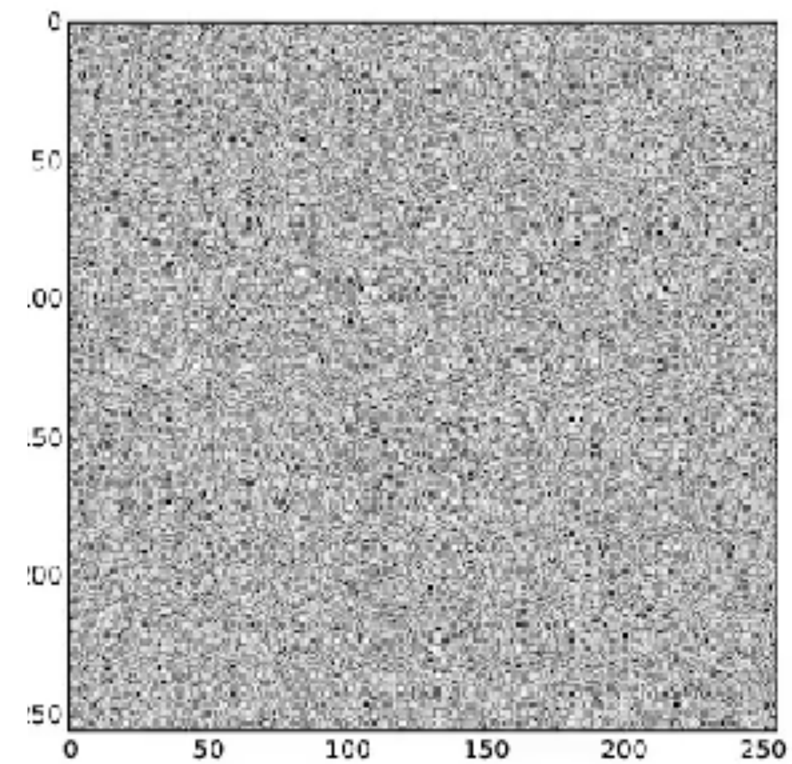
Diffusion Probabilistic Model Applied to Dead Leaves



Training Data

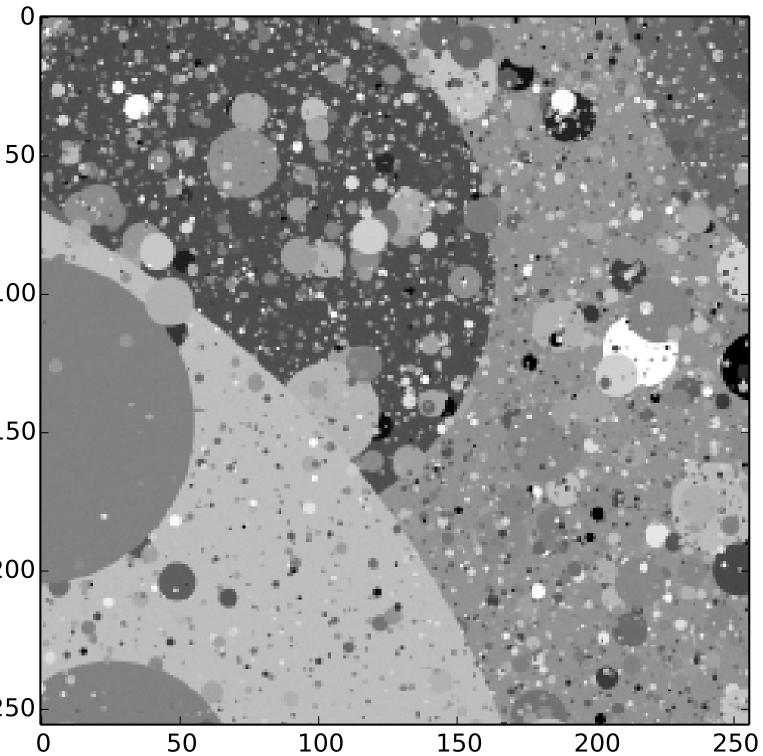


Sample from
[Theis *et al*, 2012]



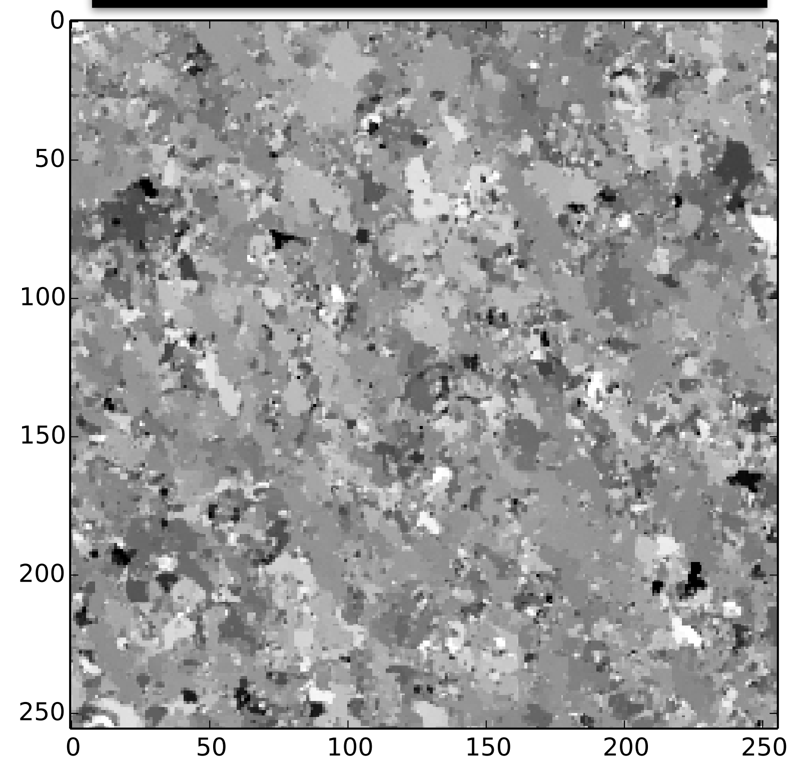
Sample from
diffusion model

Diffusion Probabilistic Model Applied to Dead Leaves



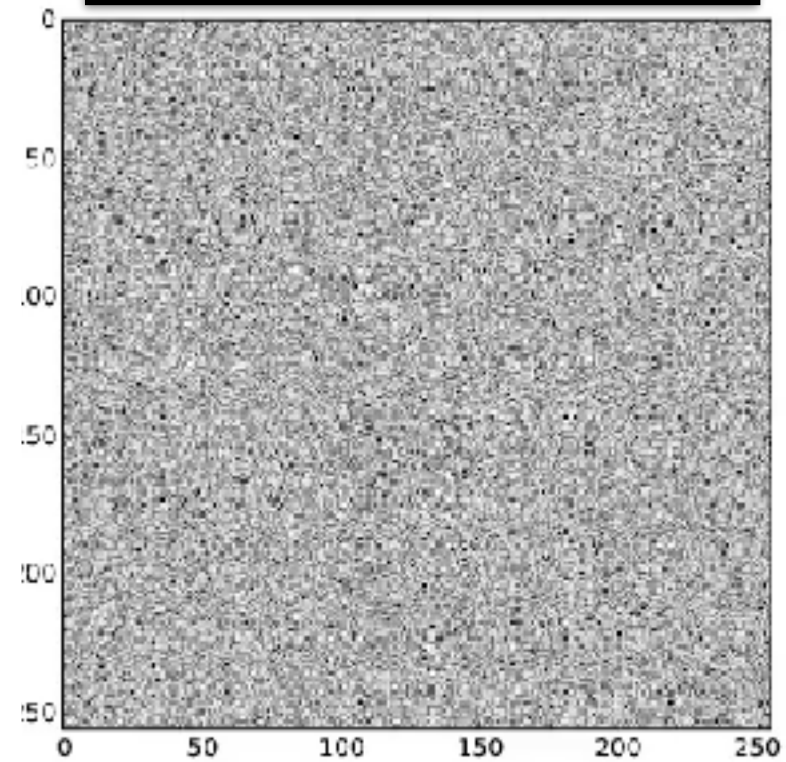
Training Data

Log likelihood
1.24 bits/pixel



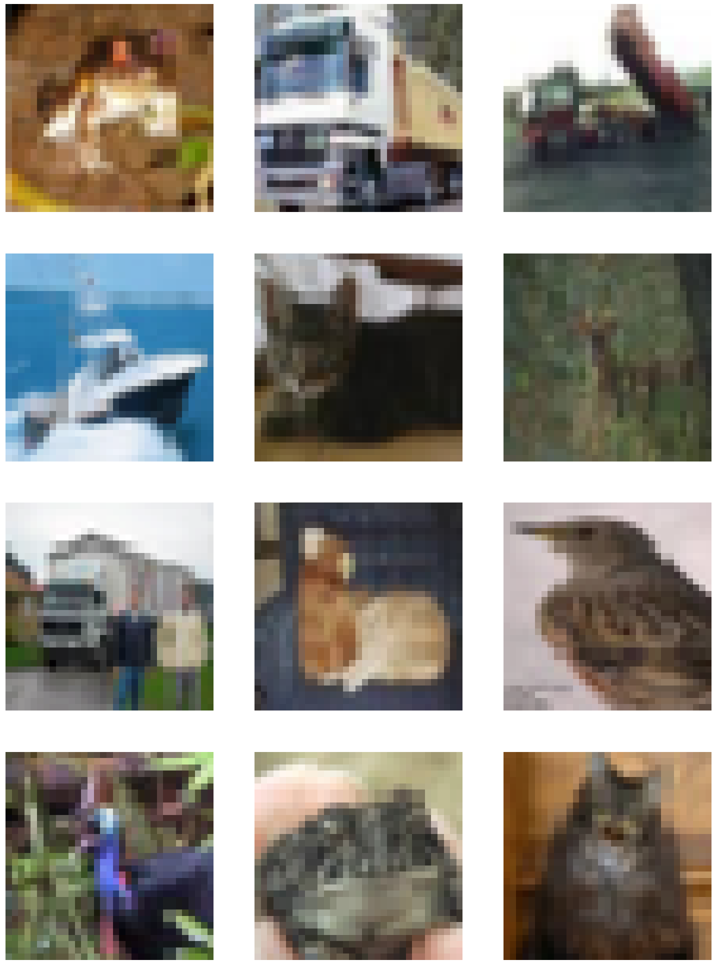
Sample from
[Theis *et al*, 2012]

Log likelihood
1.49 bits/pixel



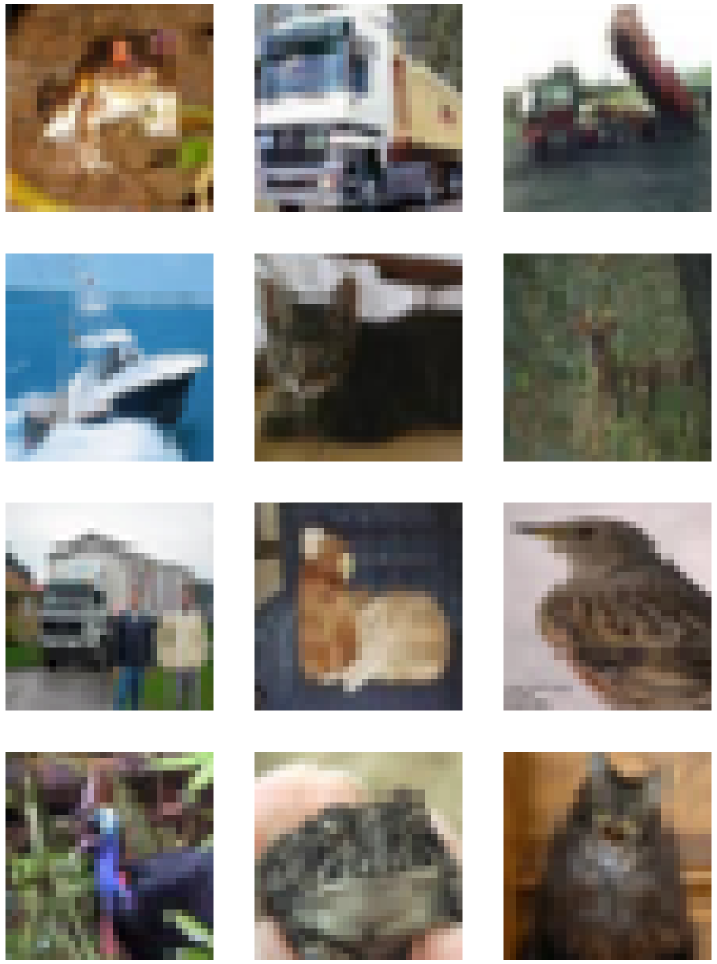
Sample from
diffusion model

Diffusion Probabilistic Model Applied to CIFAR-10

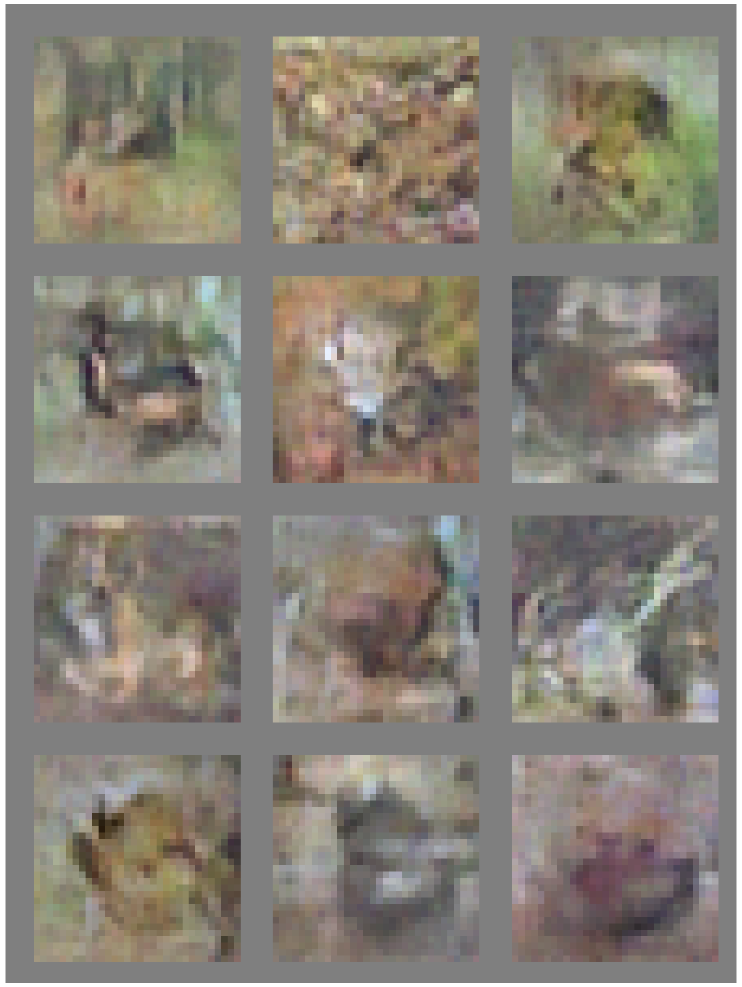


Training Data

Diffusion Probabilistic Model Applied to CIFAR-10

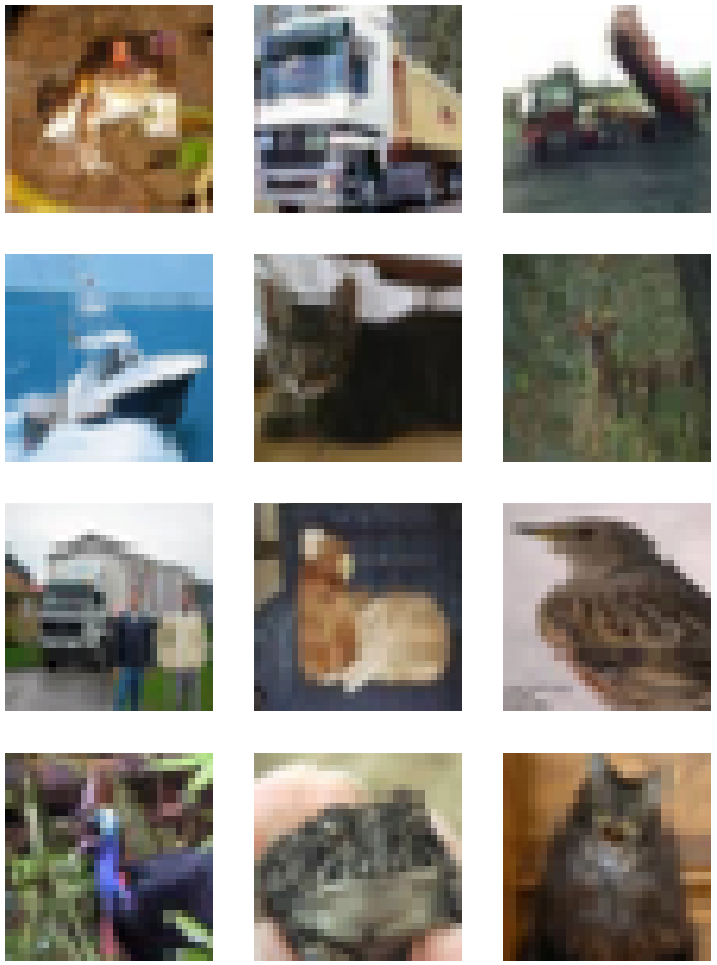


Training Data

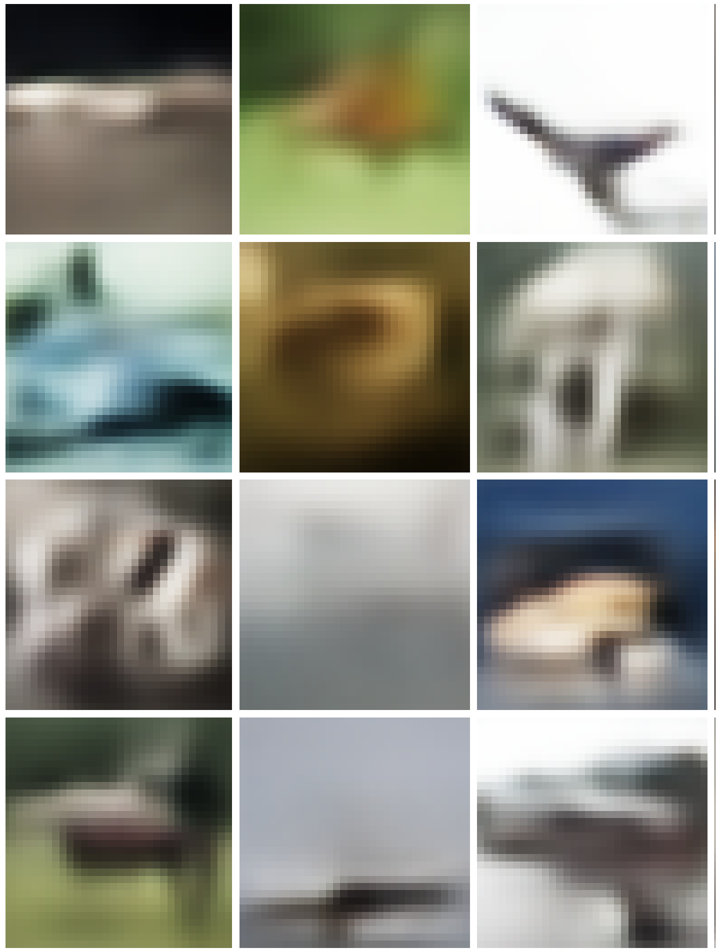


Samples from
Generative Adverserial
[Goodfellow *et al*, 2014]

Diffusion Probabilistic Model Applied to CIFAR-10

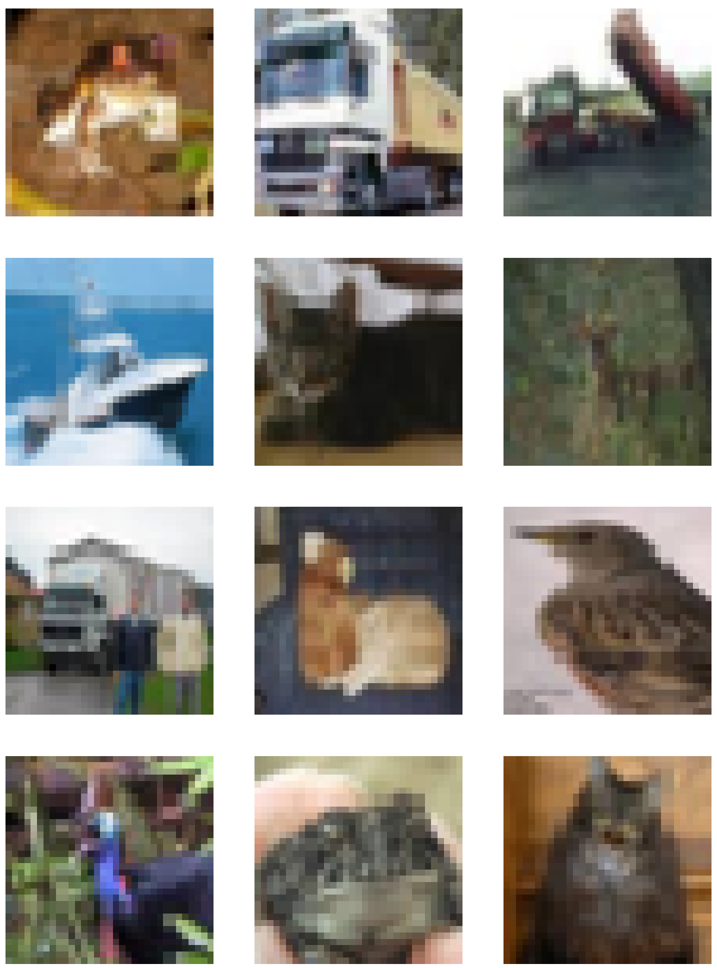


Training Data

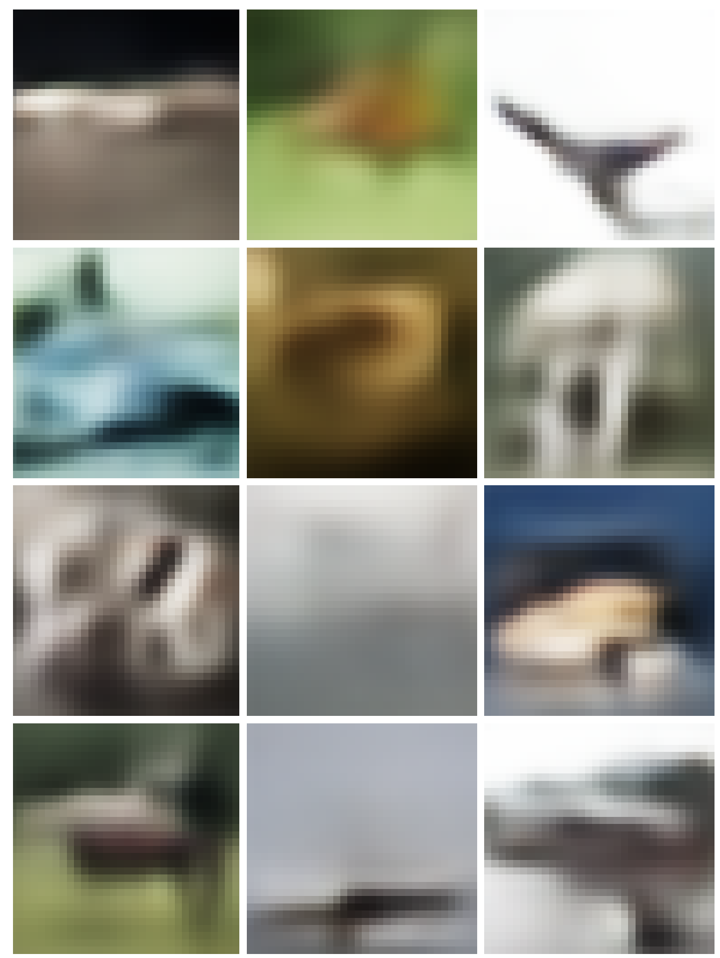


Samples from
DRAW
[Gregor *et al*, 2015]

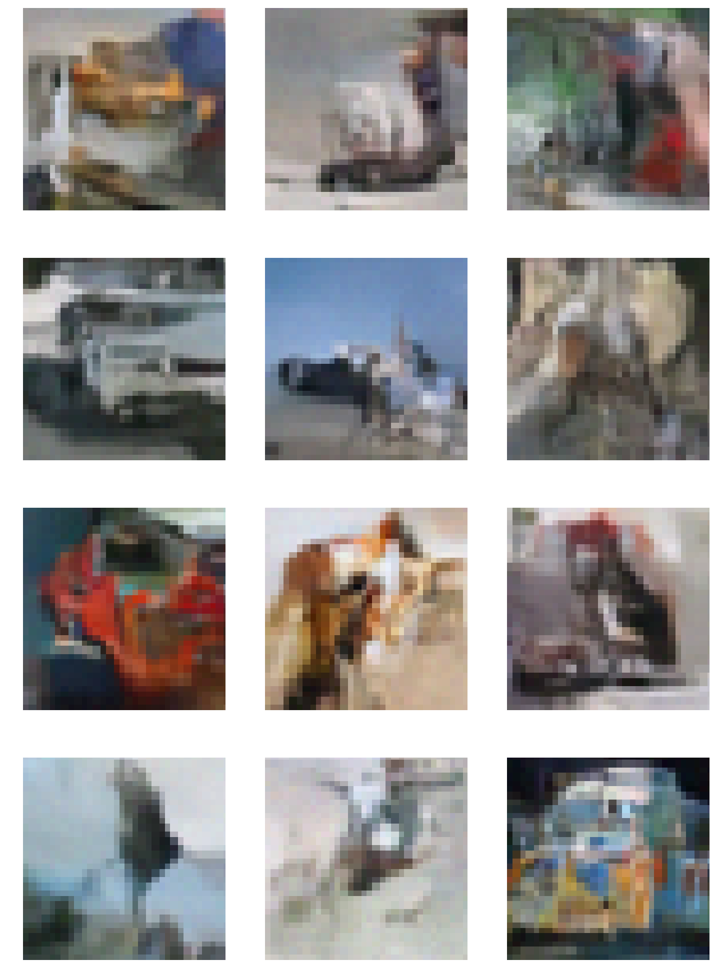
Diffusion Probabilistic Model Applied to CIFAR-10



Training Data



Samples from DRAW [Gregor *et al*, 2015]



Samples from diffusion model

Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
  - Algorithm
  - Deep convolutional network: Universal function approximator
  - **Multiplying distributions: Inputation, denoising, computing posteriors**
- Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks

Multiplying Distributions is Straightforward

Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

- Required to compute posterior distributions
  - Missing data (inpainting)
  - Corrupted data (denoising)

Multiplying Distributions is Straightforward

Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

- Required to compute posterior distributions
  - Missing data (inpainting)
  - Corrupted data (denoising)
- Difficult and expensive using competing techniques
  - e.g. VAEs, GSNs, NADEs, GANs, RNVP, most graphical models

Multiplying Distributions is Straightforward

Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

Multiplying Distributions is Straightforward

Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

Multiplying Distributions is Straightforward

Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

Acts as small perturbation to diffusion process

Multiplying Distributions is Straightforward

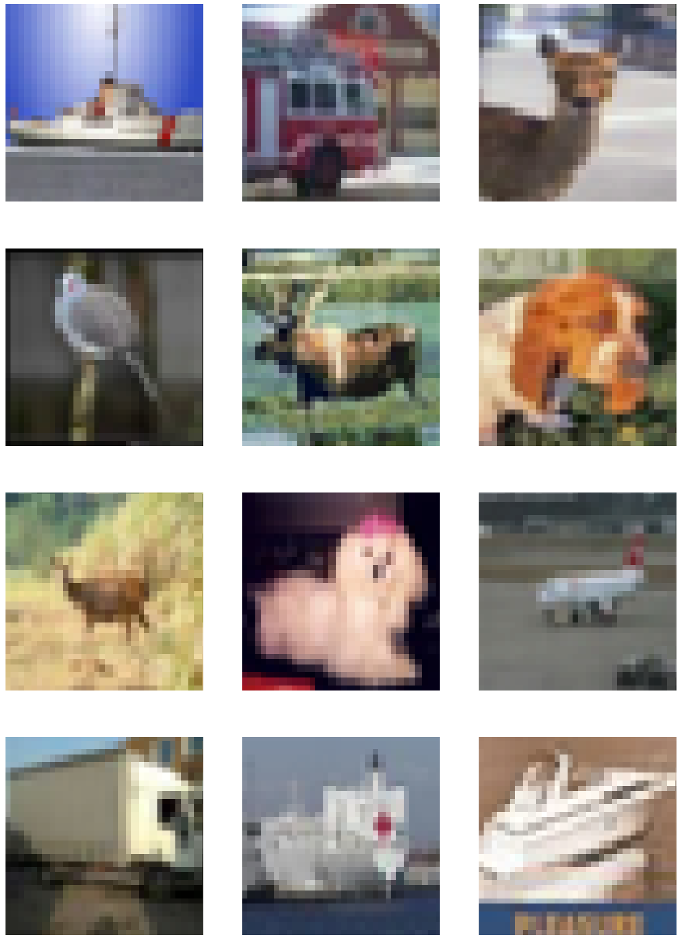
Interested in $\tilde{p}\left(\mathbf{x}^{(0)}\right) \propto p\left(\mathbf{x}^{(0)}\right) r\left(\mathbf{x}^{(0)}\right)$

Acts as small perturbation to diffusion process

$$p\left(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}\right) = \mathcal{N}\left(\mathbf{x}^{(t-1)}; f_\mu\left(\mathbf{x}^{(t)}, t\right), f_\Sigma\left(\mathbf{x}^{(t)}, t\right)\right)$$

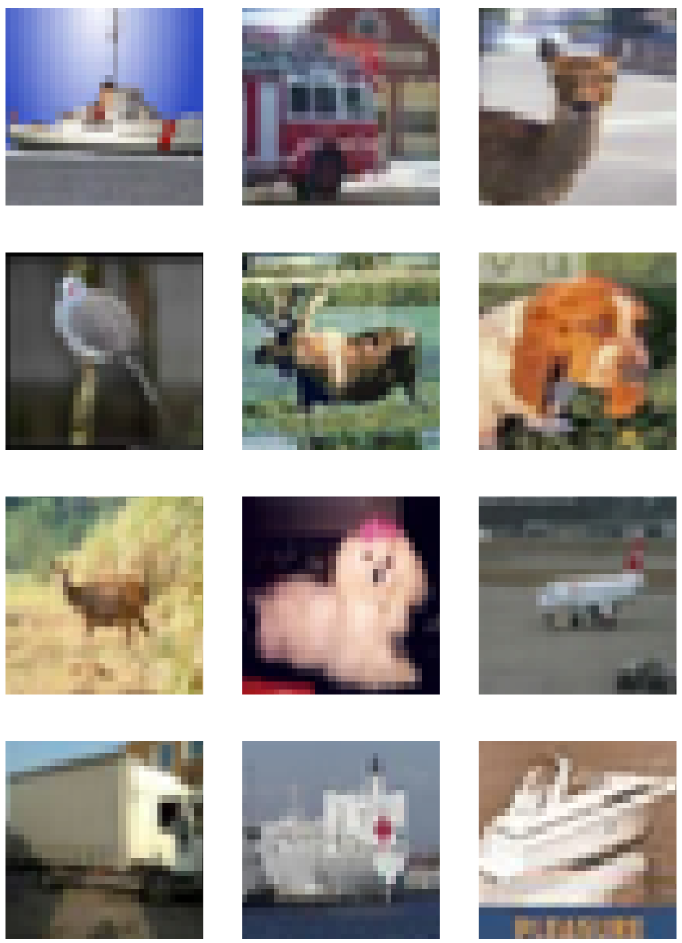
$$\tilde{p}\left(\mathbf{x}^{(t-1)} \mid \mathbf{x}^{(t)}\right) \approx \mathcal{N}\left(x^{(t-1)}; \mathbf{f}_\mu\left(\mathbf{x}^{(t)}, t\right) + \mathbf{f}_\Sigma\left(\mathbf{x}^{(t)}, t\right) \frac{\partial \log r\left(\mathbf{x}^{(t-1)'}\right)}{\partial \mathbf{x}^{(t-1)'}}\Bigg|_{\mathbf{x}^{(t-1)'}=f_\mu\left(\mathbf{x}^{(t)}, t\right)}, \mathbf{f}_\Sigma\left(\mathbf{x}^{(t)}, t\right)\right)$$

Image Denoising by Sampling from Posterior

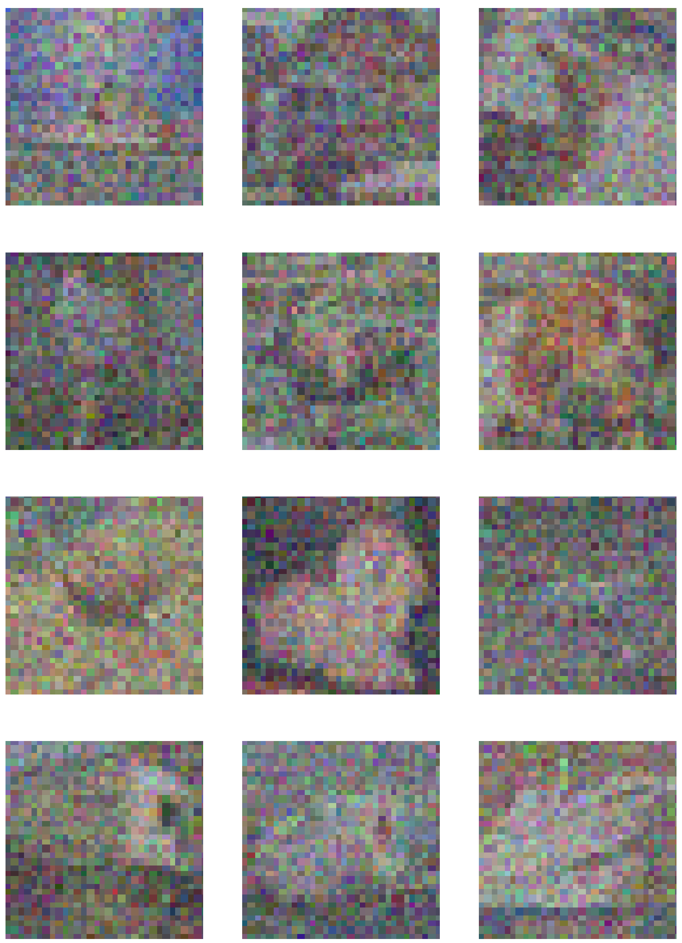


Holdout Data

Image Denoising by Sampling from Posterior

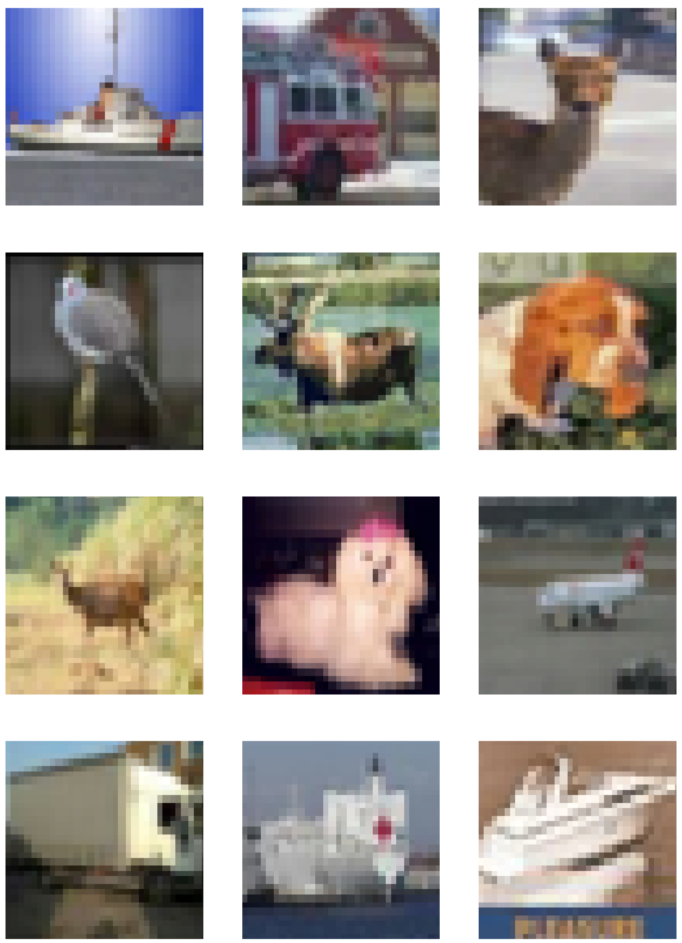


Holdout Data

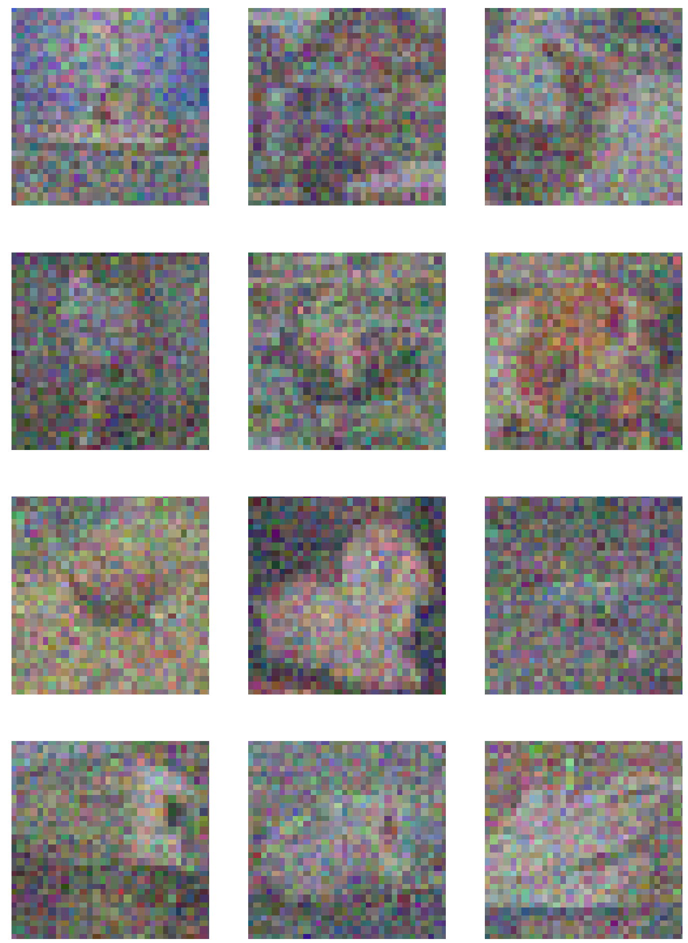


Corrupted
(SNR = 1)

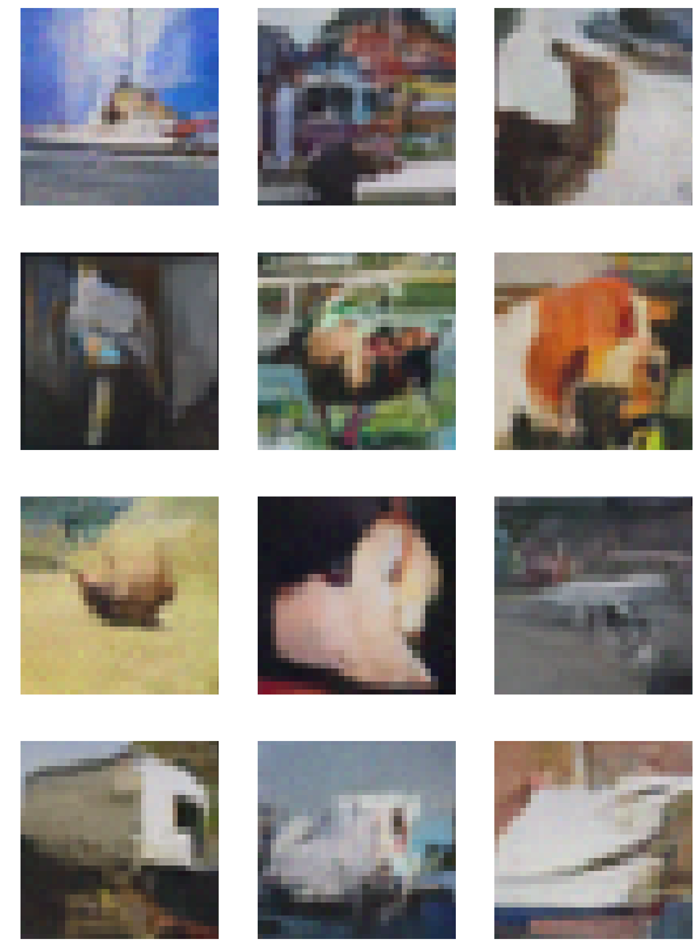
Image Denoising by Sampling from Posterior



Holdout Data

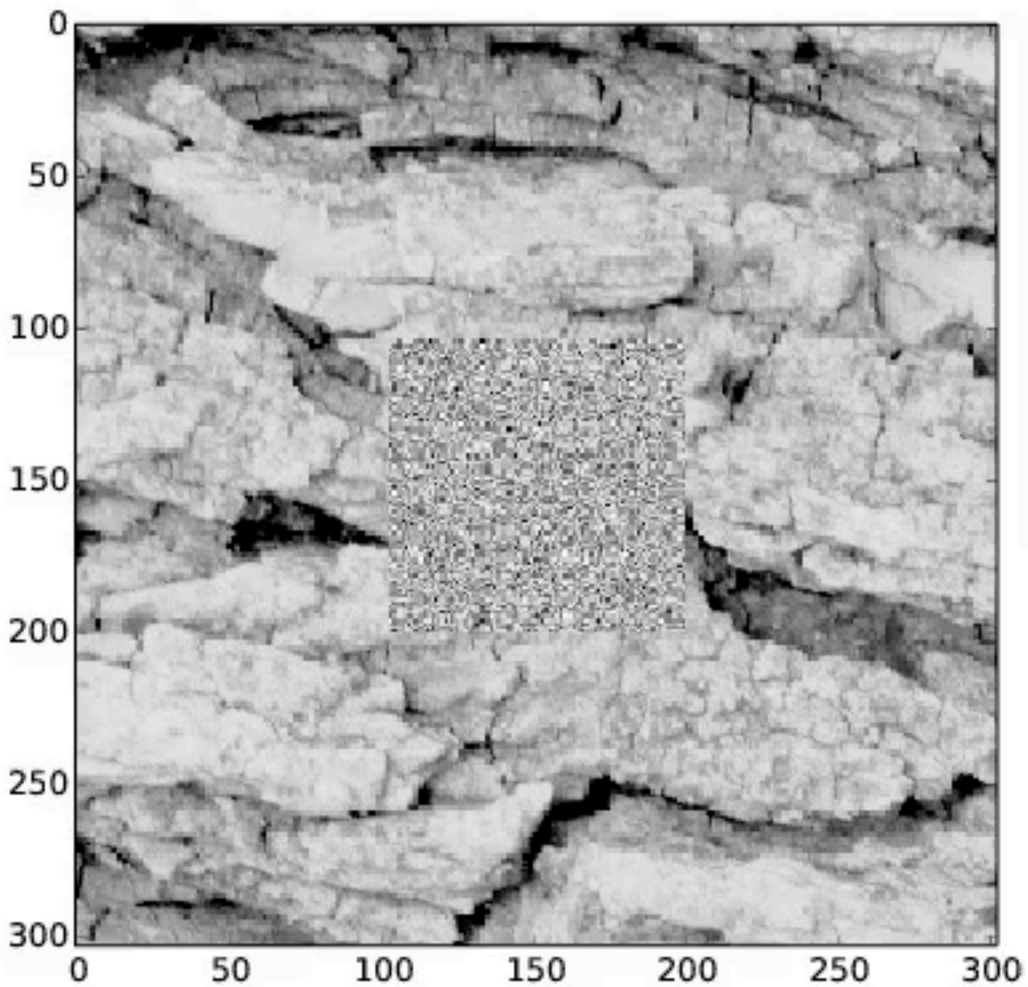


Corrupted
(SNR = 1)

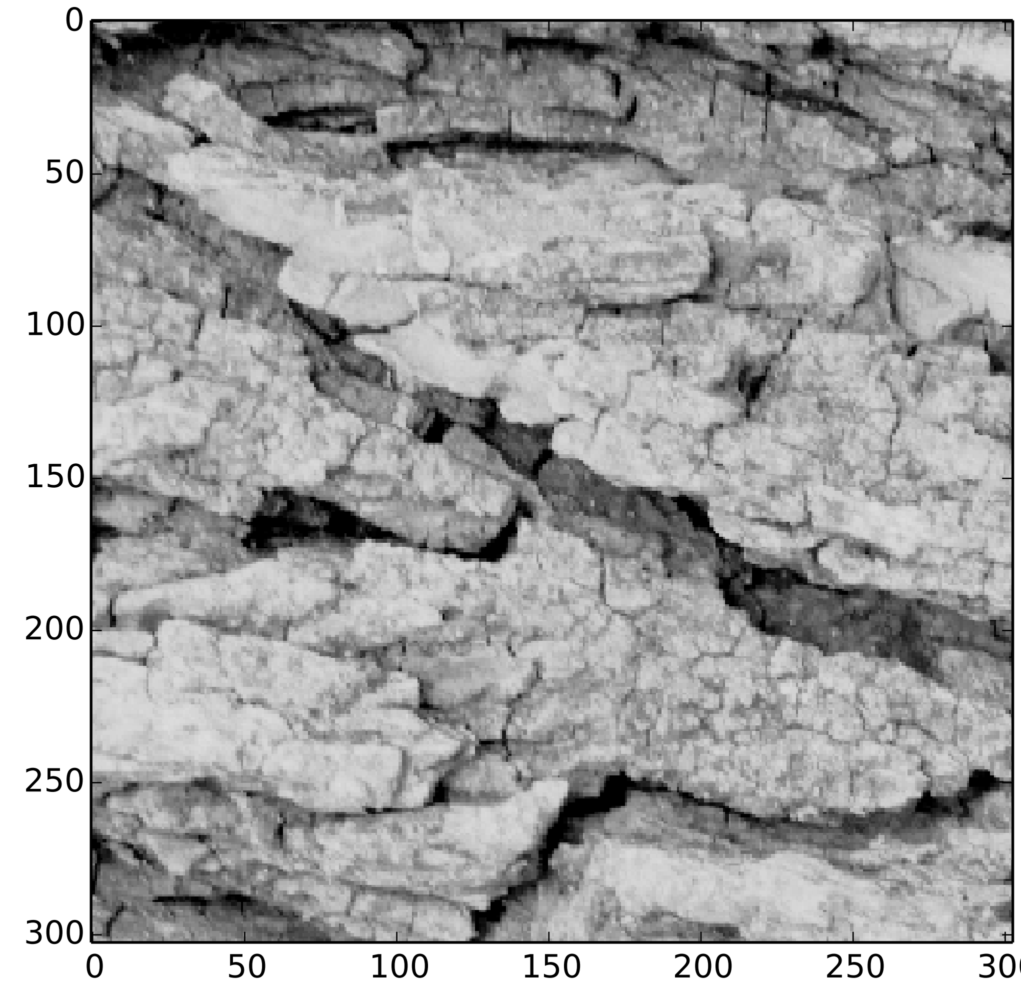


Denoised

Image Inpainting by Sampling from Posterior

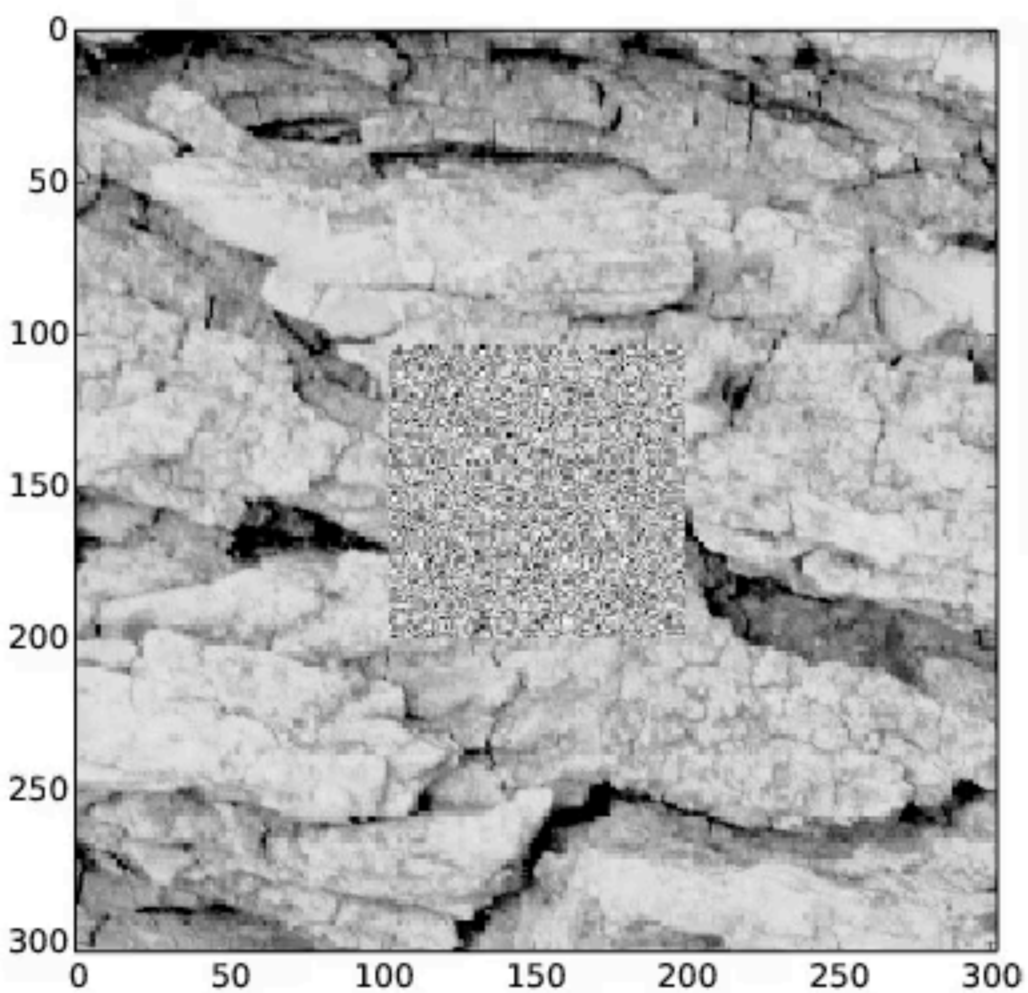


Inpainted image

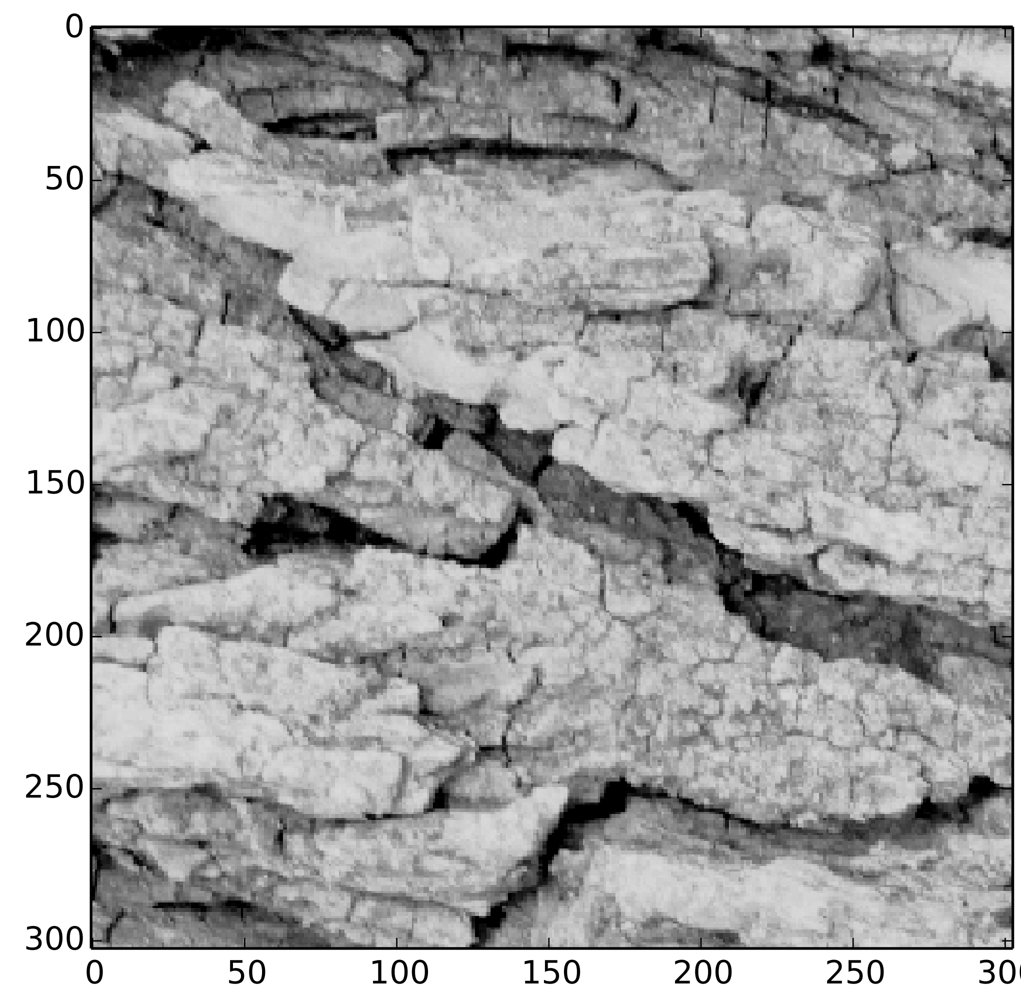


True image

Image Inpainting by Sampling from Posterior



Inpainted image



True image

Flexible and tractable method for deep unsupervised learning

Flexible and tractable method for deep unsupervised learning

- Flexible: Diffusion process for any (smooth) distribution

Flexible and tractable method for deep unsupervised learning

- Flexible: Diffusion process for any (smooth) distribution
  - Binary or continuous state space

Flexible and tractable method for deep unsupervised learning

- Flexible: Diffusion process for any (smooth) distribution
  - Binary or continuous state space
- Tractable: Training, exact sampling, inference, evaluation

Flexible and tractable method for deep unsupervised learning

- Flexible: Diffusion process for any (smooth) distribution
    - Binary or continuous state space
- Tractable: Training, exact sampling, inference, evaluation
- Deep networks with thousands of layers (/ time steps)

Flexible and tractable method for deep unsupervised learning

- Flexible: Diffusion process for any (smooth) distribution
  - Binary or continuous state space
- Tractable: Training, exact sampling, inference, evaluation
- Deep networks with thousands of layers (/ time steps)
- Easy to multiply distributions (e.g. for posterior)

Flexible and tractable method for deep unsupervised learning

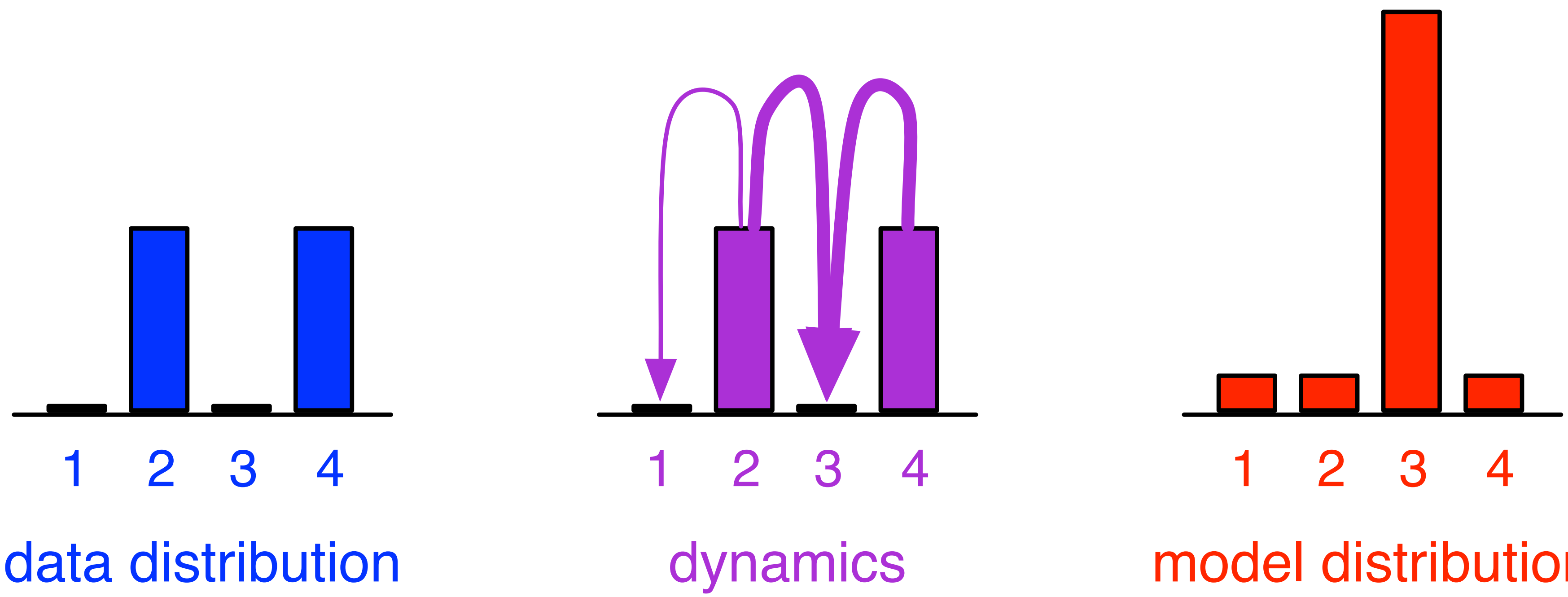
- Flexible: Diffusion process for any (smooth) distribution
  - Binary or continuous state space
- Tractable: Training, exact sampling, inference, evaluation
- Deep networks with thousands of layers (/ time steps)
- Easy to multiply distributions (e.g. for posterior)
- Bounds on entropy production

Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
- **Other projects: Inverse Ising, non-equilibrium Monte Carlo, stat. mech. of neural networks**

Minimum Probability Flow Learning

- Estimate parameters in energy based models, by minimizing probability flow under master equation from stat. mech.



$$p_i^{(0)} = \text{fraction data in state } i$$

$$p_i^{(\infty)}(\theta) = \frac{e^{-E_i(\theta)}}{Z(\theta)}$$

$$\dot{p}_i^{(t)} = \sum_{j \neq i} \Gamma_{ij}(\theta)\, p_j^{(0)} - \sum_{j \neq i} \Gamma_{ji}(\theta)\, p_i^{(0)}$$

$$\dot{p}_i^{(t)} = \sum_{j \neq i} \Gamma_{ij}(\theta)\, p_j^{(t)} - \sum_{j \neq i} \Gamma_{ji}(\theta)\, p_i^{(t)}$$

$$\dot{p}_i^{(\infty)} = 0$$

[PRL, 2011]
[ICML, 2011]

Minimum Probability Flow Learning

- Estimate parameters in energy based models, by minimizing probability flow under master equation from stat. mech.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \, D_{KL}\left(\mathbf{p}^{(0)} \middle\| \mathbf{p}^{(\epsilon)}(\theta)\right)$$

data distribution | dynamics | model distribution

$$p_i^{(0)} = \text{fraction data in state } i$$

$$p_i^{(\infty)}(\theta) = \frac{e^{-E_i(\theta)}}{Z(\theta)}$$

$$\dot{p}_i^{(t)} = \sum_{j \neq i} \Gamma_{ij}(\theta)\, p_j^{(0)} - \sum_{j \neq i} \Gamma_{ji}(\theta)\, p_i^{(0)}$$

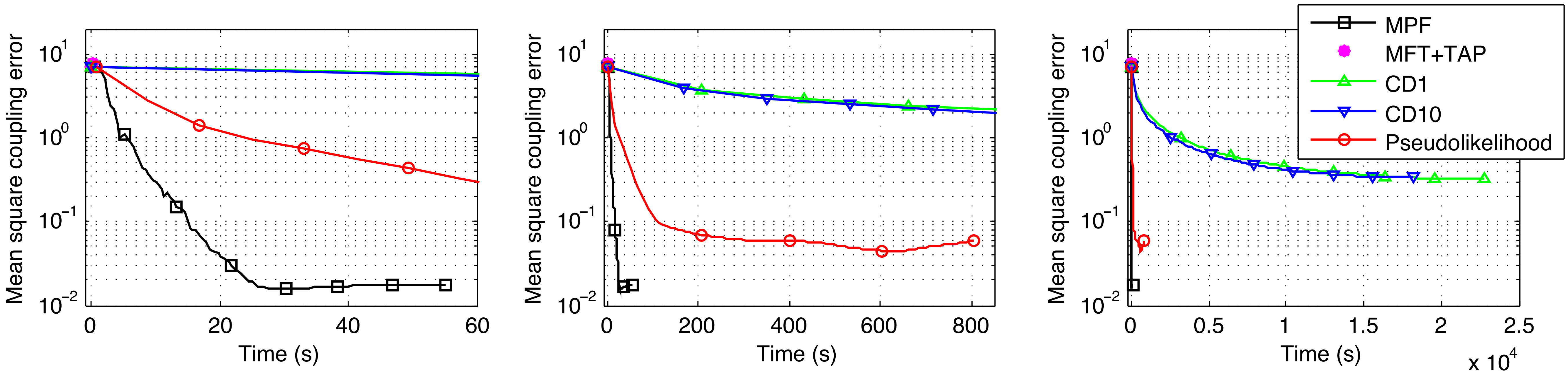
$$\dot{p}_i^{(t)} = \sum_{j \neq i} \Gamma_{ij}(\theta)\, p_j^{(t)} - \sum_{j \neq i} \Gamma_{ji}(\theta)\, p_i^{(t)}$$

$$\dot{p}_i^{(\infty)} = 0$$

[PRL, 2011]
[ICML, 2011]

Minimum Probability Flow Learning

- More rapidly solves inverse Ising problem (estimate Ising couplings from samples)



First 60 seconds | First 800 seconds | First 25,000 seconds

[PRL, 2011]
[ICML, 2011]

Hamiltonian (hybrid) Monte Carlo Without Detailed Balance

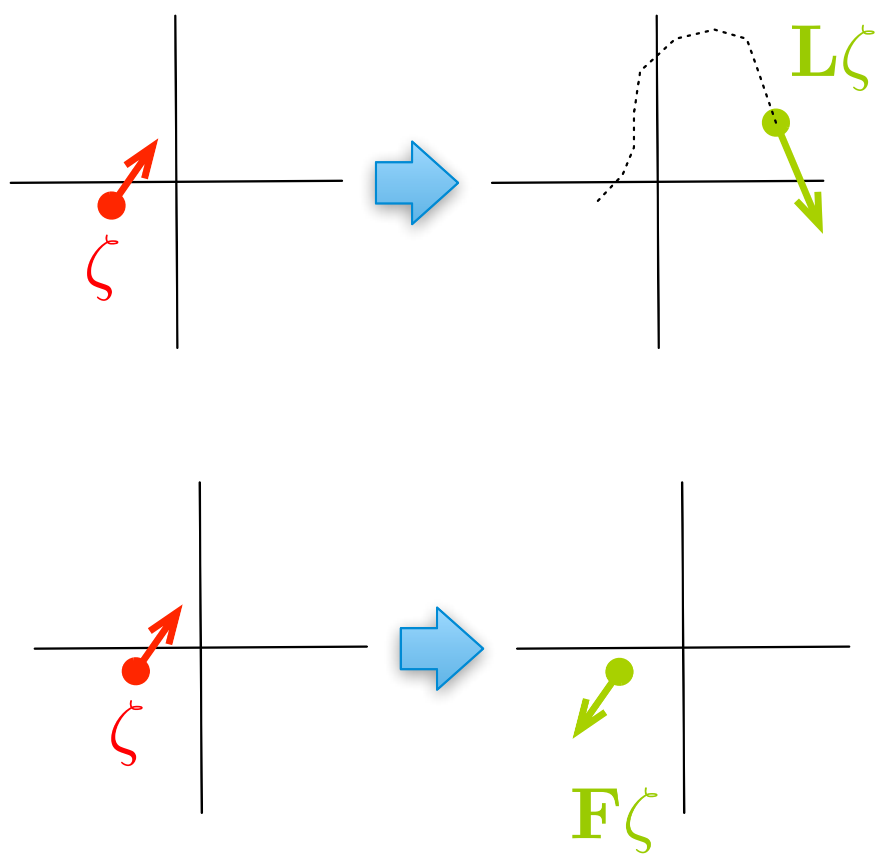
- HMC as operators on discrete state space

[ICML, 2014]

Hamiltonian (hybrid) Monte Carlo Without Detailed Balance

- HMC as operators on discrete state space

Operators:

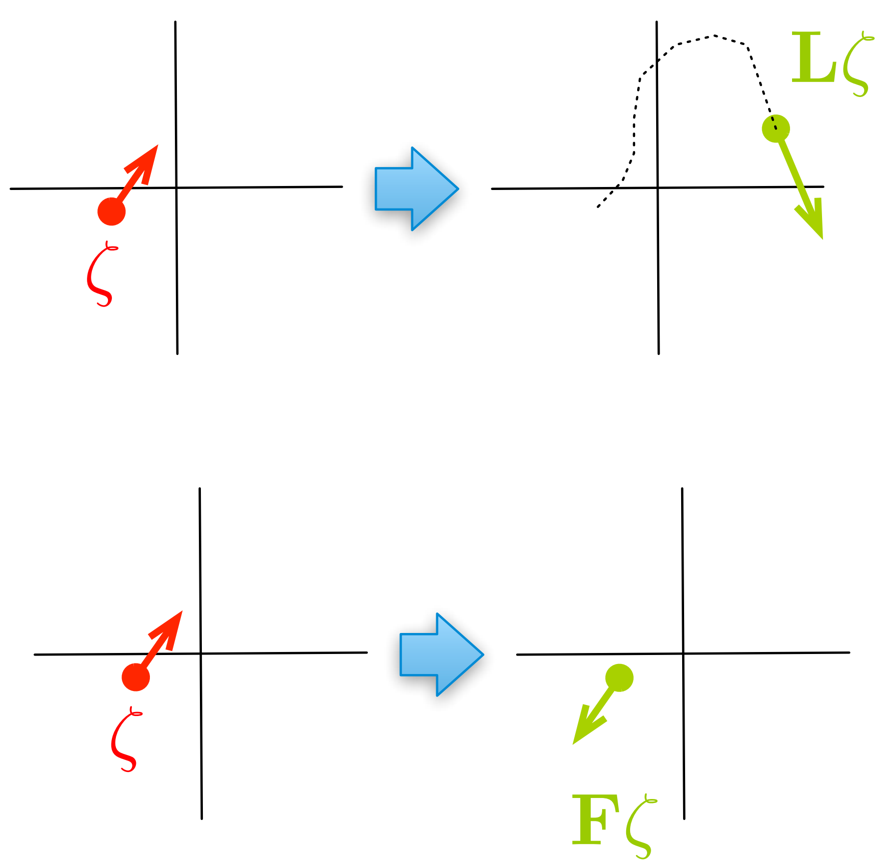


[ICML, 2014]

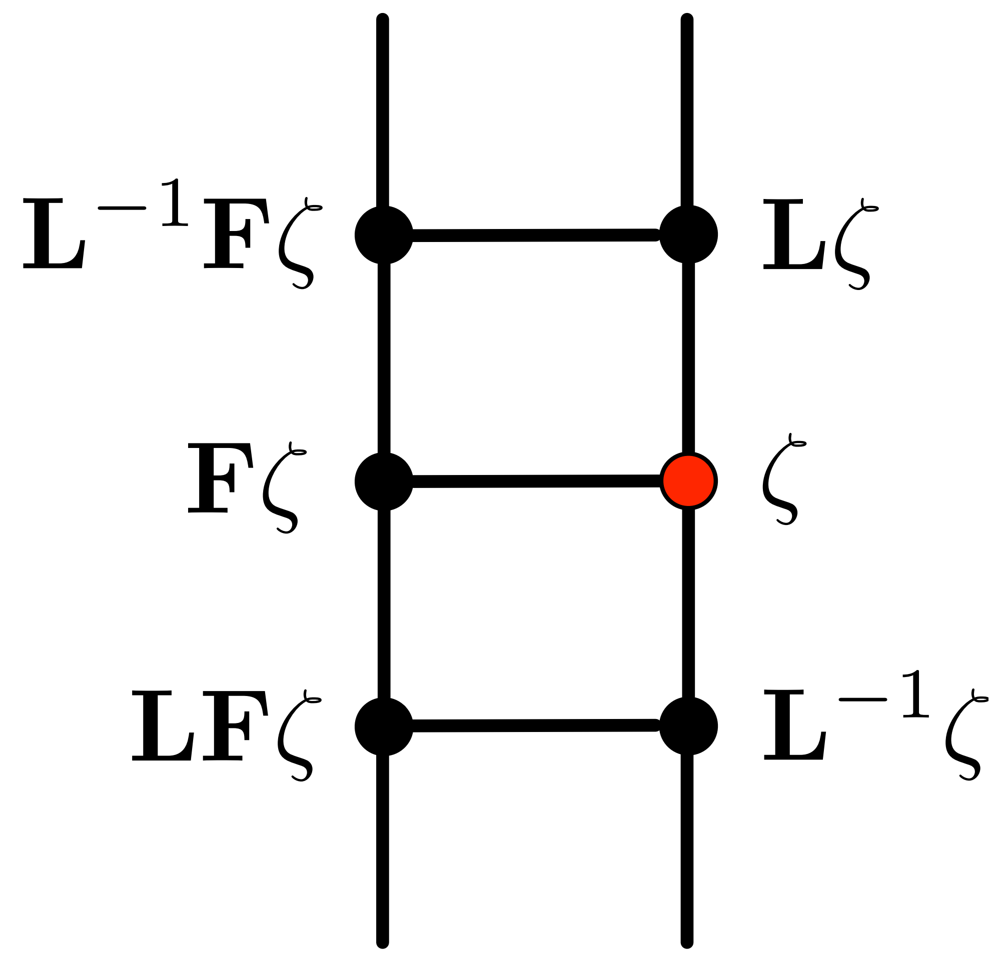
Hamiltonian (hybrid) Monte Carlo Without Detailed Balance

- HMC as operators on discrete state space

Operators:



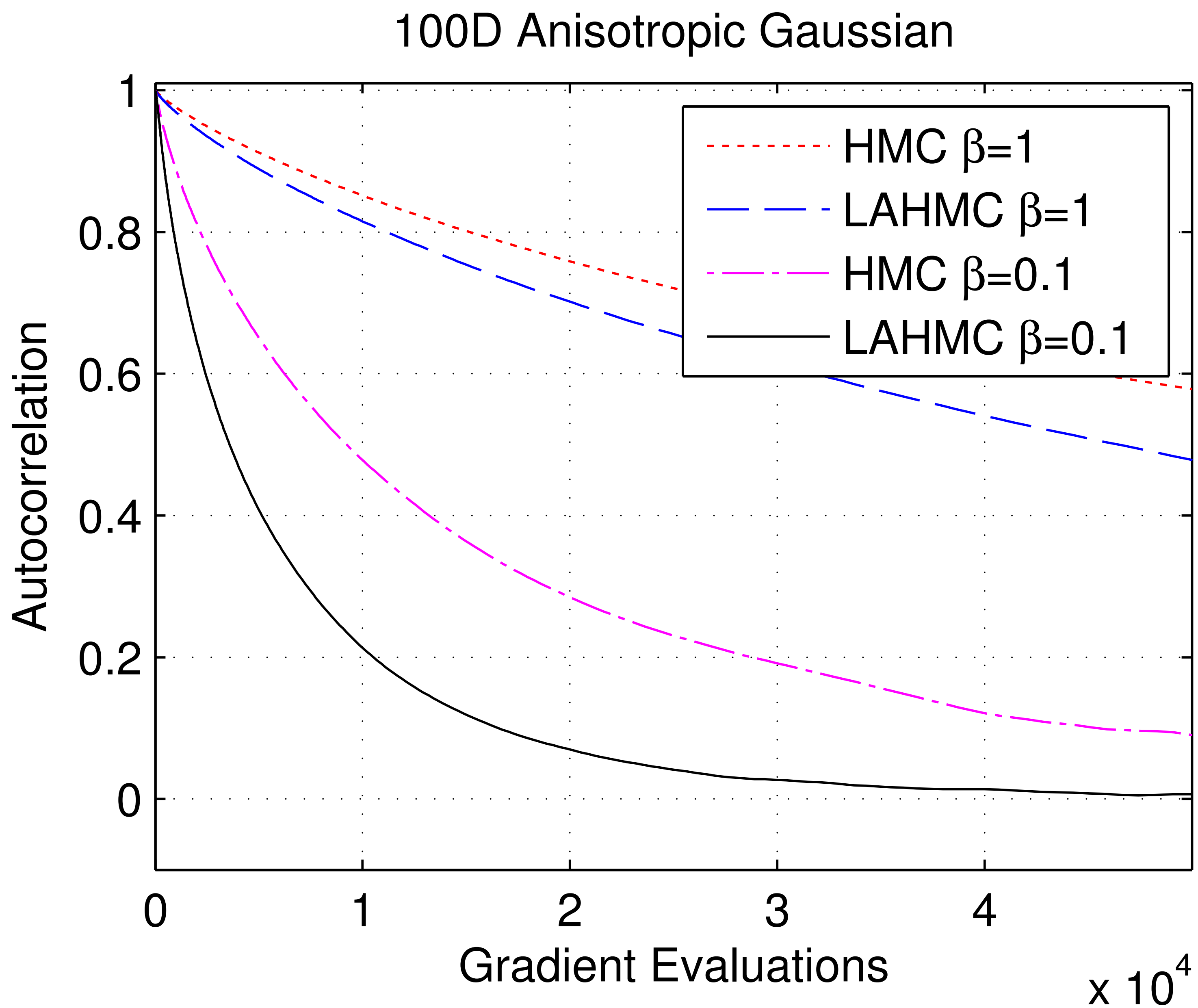
State space:



[ICML, 2014]

Hamiltonian (hybrid) Monte Carlo Without Detailed Balance

- Improved mixing by violating detailed balance



[ICML, 2014]

Statistical Physics of Deep Neural Networks

Neural network after random initialization:

$$z_i^l = \sum_j W_{ij}^l y_j^l + b^j \qquad y_i^{l+1} = \phi(z_i^l)$$

$$W_{ij}^l \sim \mathcal{N}(0, \sigma_w^2 / N_{l-1}) \qquad b_i^l \sim \mathcal{N}(0, \sigma_b^2)$$

[NIPS, 2016]
[ICLR, 2017]

Statistical Physics of Deep Neural Networks

Neural network after random initialization:

$$z_i^l = \sum_j W_{ij}^l y_j^l + b^j \qquad y_i^{l+1} = \phi(z_i^l)$$

$$W_{ij}^l \sim \mathcal{N}(0, \sigma_w^2 / N_{l-1}) \qquad b_i^l \sim \mathcal{N}(0, \sigma_b^2)$$

Central limit theorem ➡ recurrence relation:

$$z_i^l \sim \mathcal{N}(0, q^l)$$

$$q^l = \sigma_w^2 \frac{1}{\sqrt{2\pi}} \int dz e^{-\frac{1}{2}z^2} \phi^2(\sqrt{q^{l-1}} z) + \sigma_b^2$$

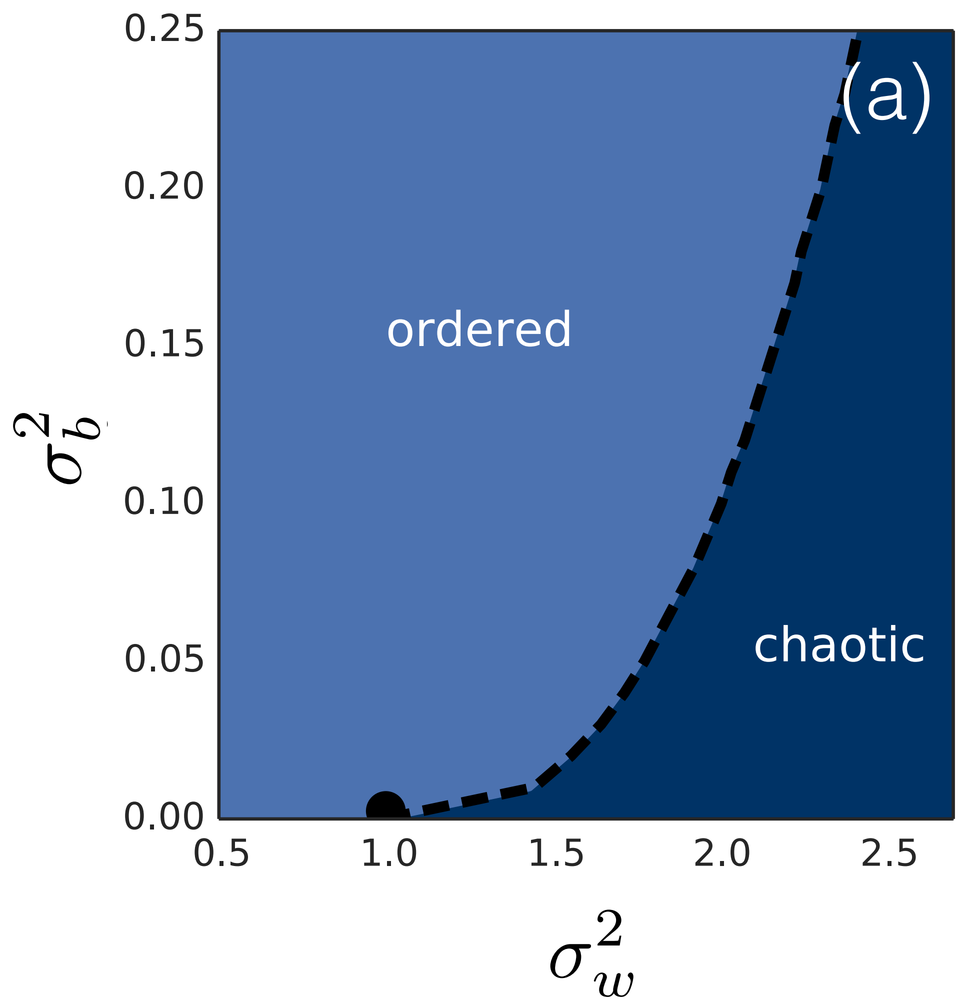
[NIPS, 2016]
[ICLR, 2017]

Statistical Physics of Deep Neural Networks

[NIPS, 2016]
[ICLR, 2017]

Statistical Physics of Deep Neural Networks

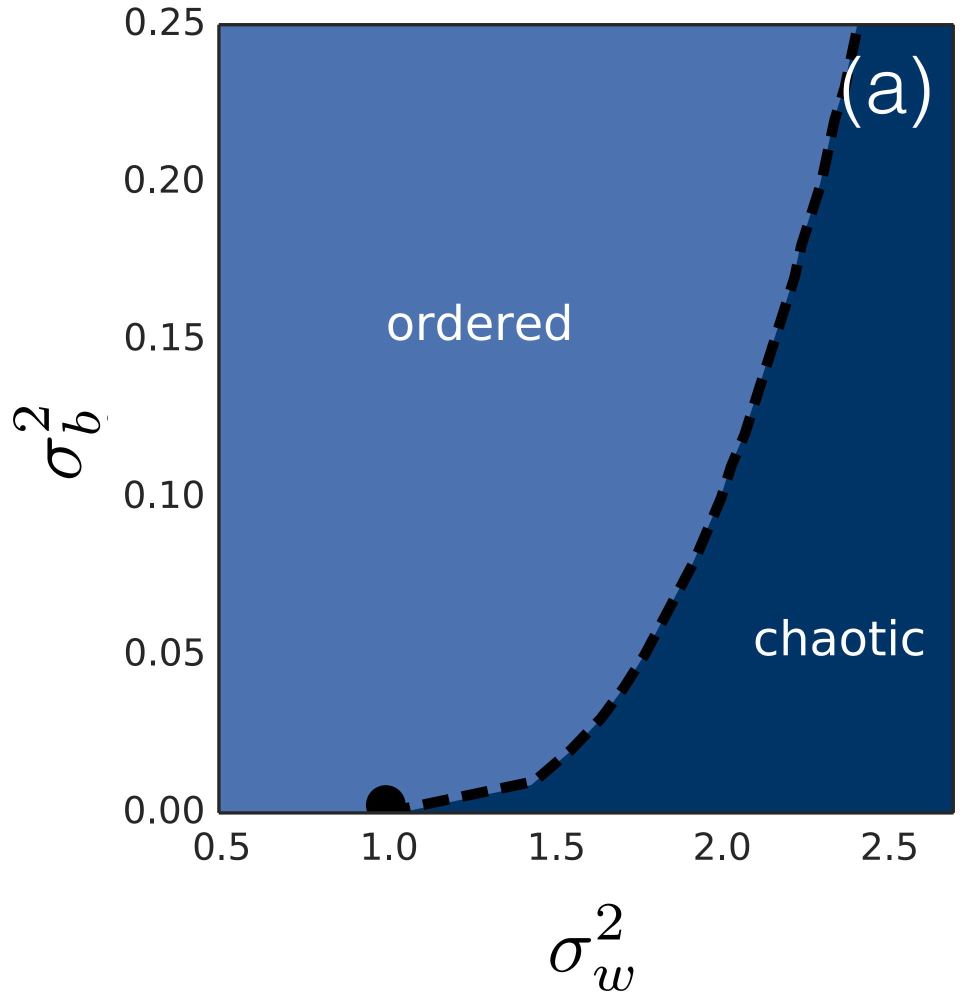
Phase diagram:



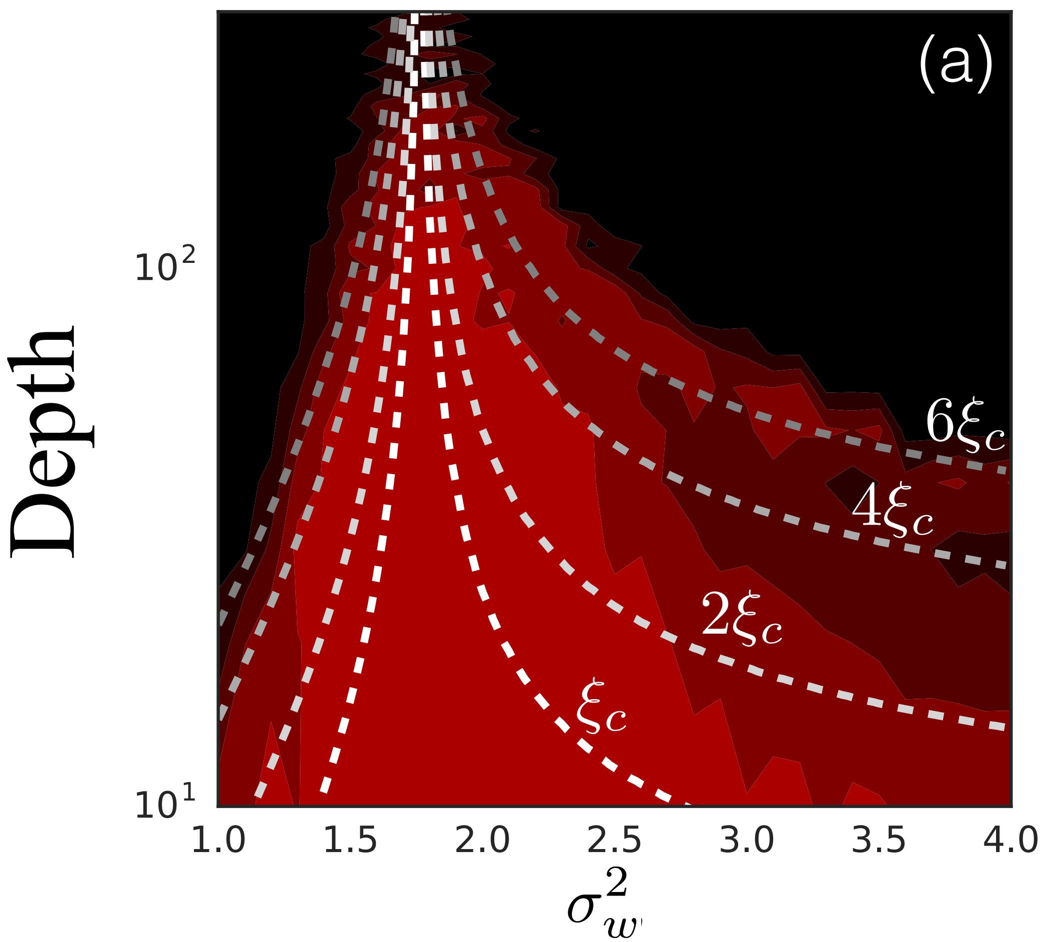
[NIPS, 2016]
[ICLR, 2017]

Statistical Physics of Deep Neural Networks

Phase diagram:



Predict trainable depth:



[NIPS, 2016]
[ICLR, 2017]

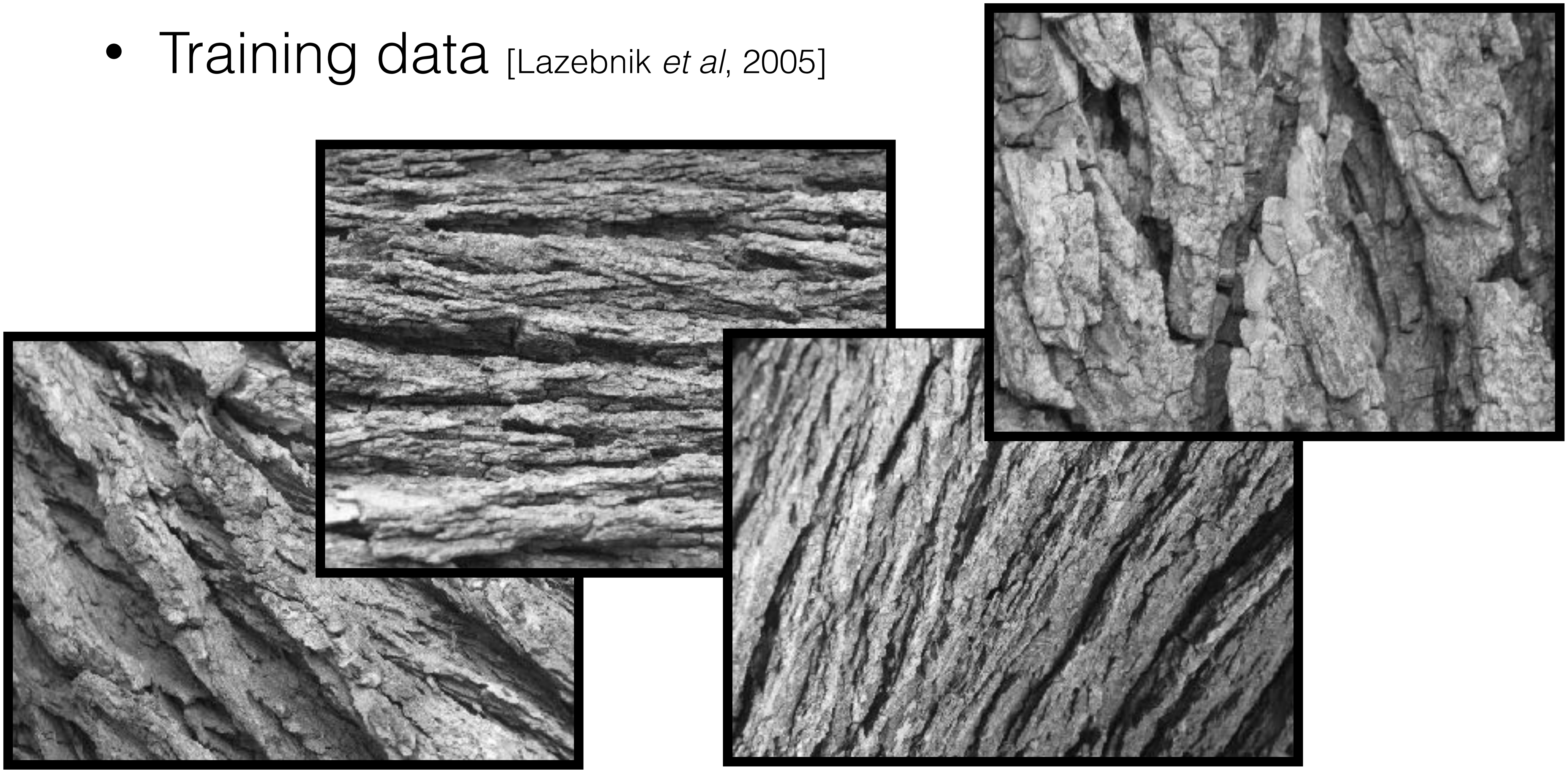
Thanks!

- Unsupervised Learning using Nonequilibrium Thermodynamics
  - Eric Weiss
  - Niru Maheswaranathan
  - Surya Ganguli
- Minimum Probability Flow
  - Peter Battaglino
  - Michael R. DeWeese
- Hamiltonian Monte Carlo without Detailed Balance
  - Mayur Mudigonda
  - Michael R. DeWeese
- Statistical Physics of Deep Networks
  - Sam Schoenholz
  - Ben Poole
  - Jeffrey Pennington
  - Justin Gilmer
  - Surya Ganguli
  - Maithra Raghu
  - Subhaneil Lahiri

SCRAP SLIDES FROM HERE ON

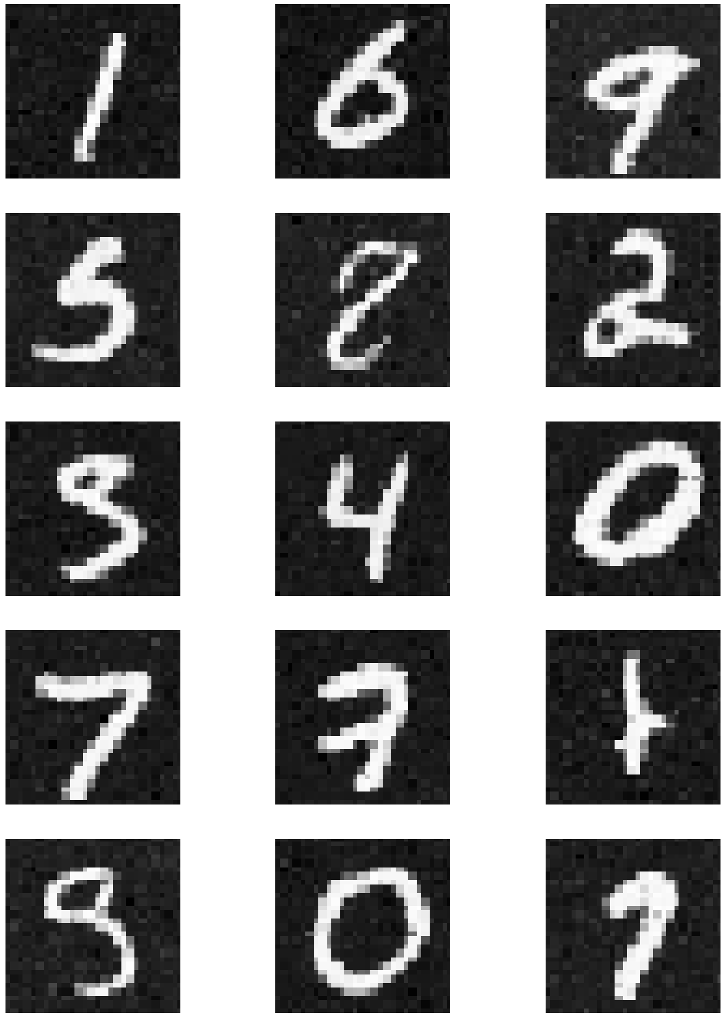
Image Inpainting by Sampling from Posterior

- Training data [Lazebnik *et al*, 2005]



Diffusion Probabilistic Model Applied to MNIST

Model	**Log likelihood estimate***
Stacked CAE	121 ± 1.6 bits
DBN	138 ± 2 bits
Deep GSN	214 ± 1.1 bits
Diffusion	**220 ± 1.9 bits**
Adversarial net	225 ± 2 bits



Samples from diffusion model

* via Parzen window code from [Goodfellow *et al*, 2014]

Future Work

Future Work

- Continuous time formulation

Future Work

- Continuous time formulation
- Perturbation around energy based model

Future Work

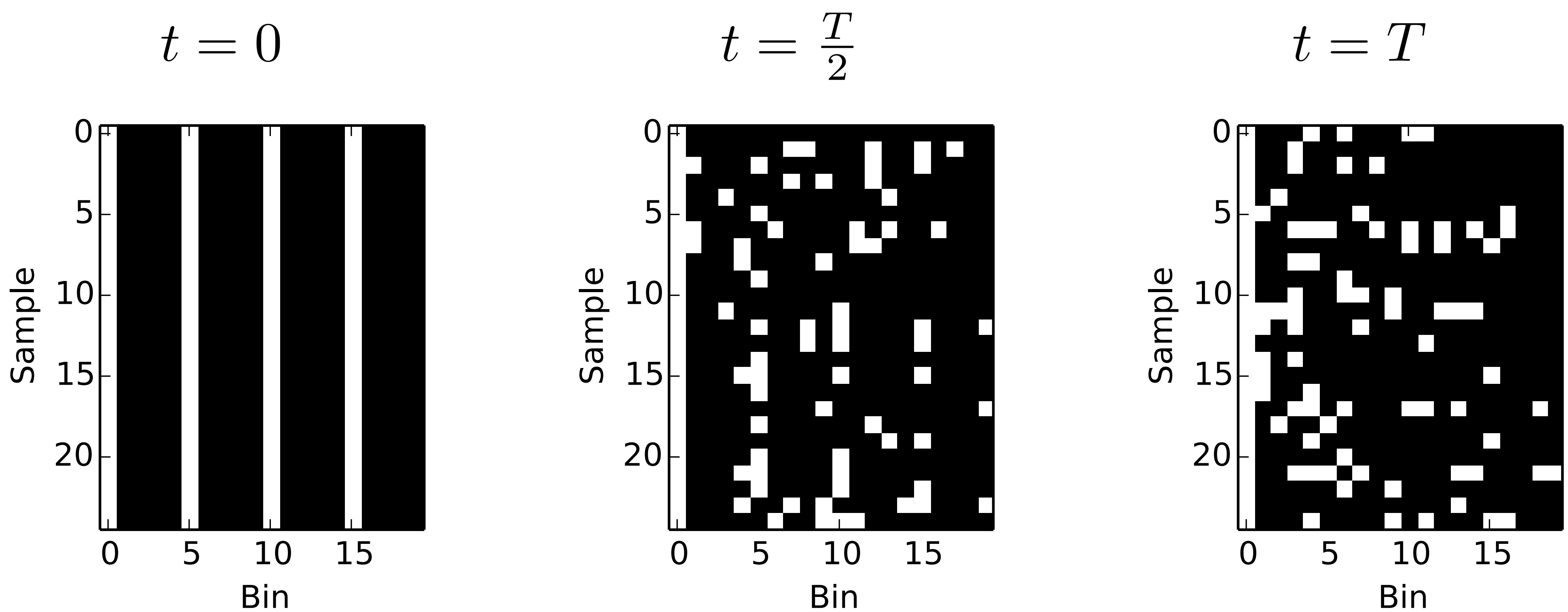
- Continuous time formulation
- Perturbation around energy based model
- Binary data (e.g. spike trains)

Outline

- Other projects: Training energy based models, Monte Carlo, deep learning theory
- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results

Toy Binary Sequence Learning

$$p\left(\mathbf{x}^{(0\cdots T)}\right)$$



Outline

- Motivation: The promise of deep unsupervised learning
- Physical intuition: Diffusion processes and time reversal
- Diffusion probabilistic model: Derivation and experimental results
  - Algorithm
  - **Deep convolutional network: Universal function approximator**
  - Multiplying distributions: Inputation, denoising, computing posteriors

Deep Networks

- Extremely flexible, parametric, function approximation

Deep Networks

- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity

Deep Networks

- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity

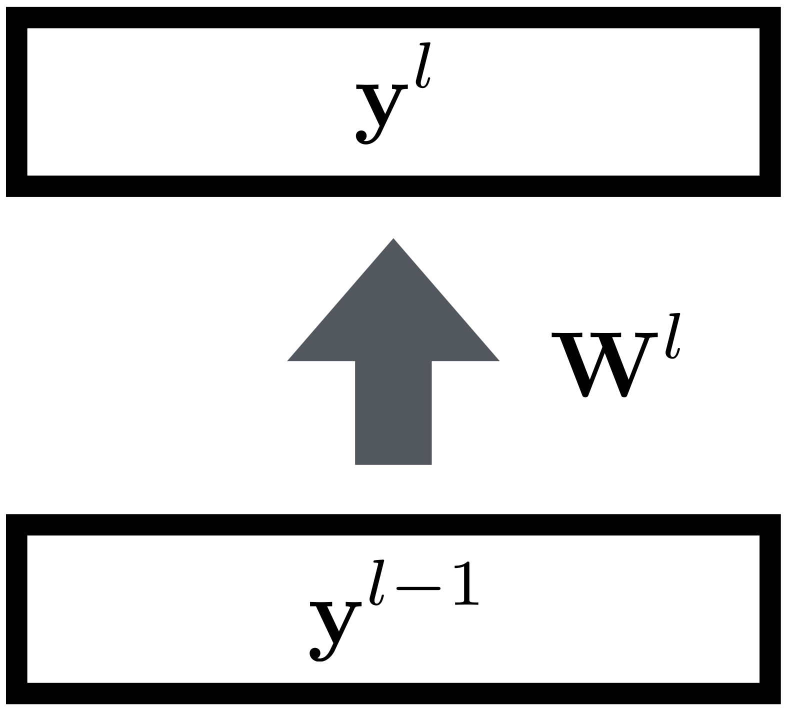
$$\mathbf{y}^l = \sigma\left(\mathbf{W}^l \mathbf{y}^{l-1}\right)$$

Deep Networks

- Extremely flexible, parametric, function approximation

- **Single layer:** linear transformation, pointwise nonlinearity

$$\mathbf{y}^l = \sigma\left(\mathbf{W}^l \mathbf{y}^{l-1}\right)$$



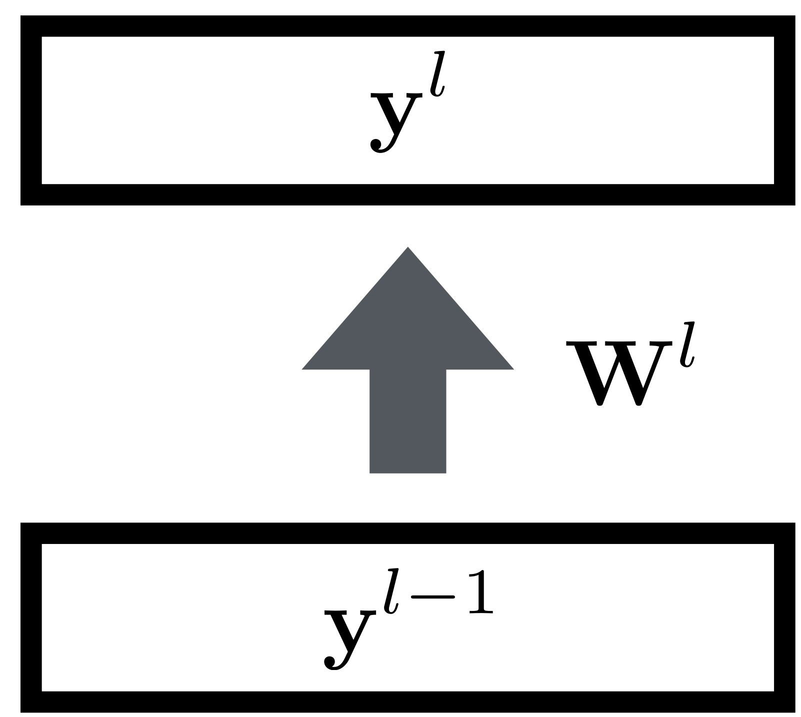
Deep Networks

- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity

$$\mathbf{y}^l = \sigma\left(\mathbf{W}^l \mathbf{y}^{l-1}\right)$$

$$\sigma(u) \equiv \text{leaky ReLU}$$

$$= \begin{cases} u & u \geq 0 \\ 0.05u & u < 0 \end{cases}$$



Deep Networks

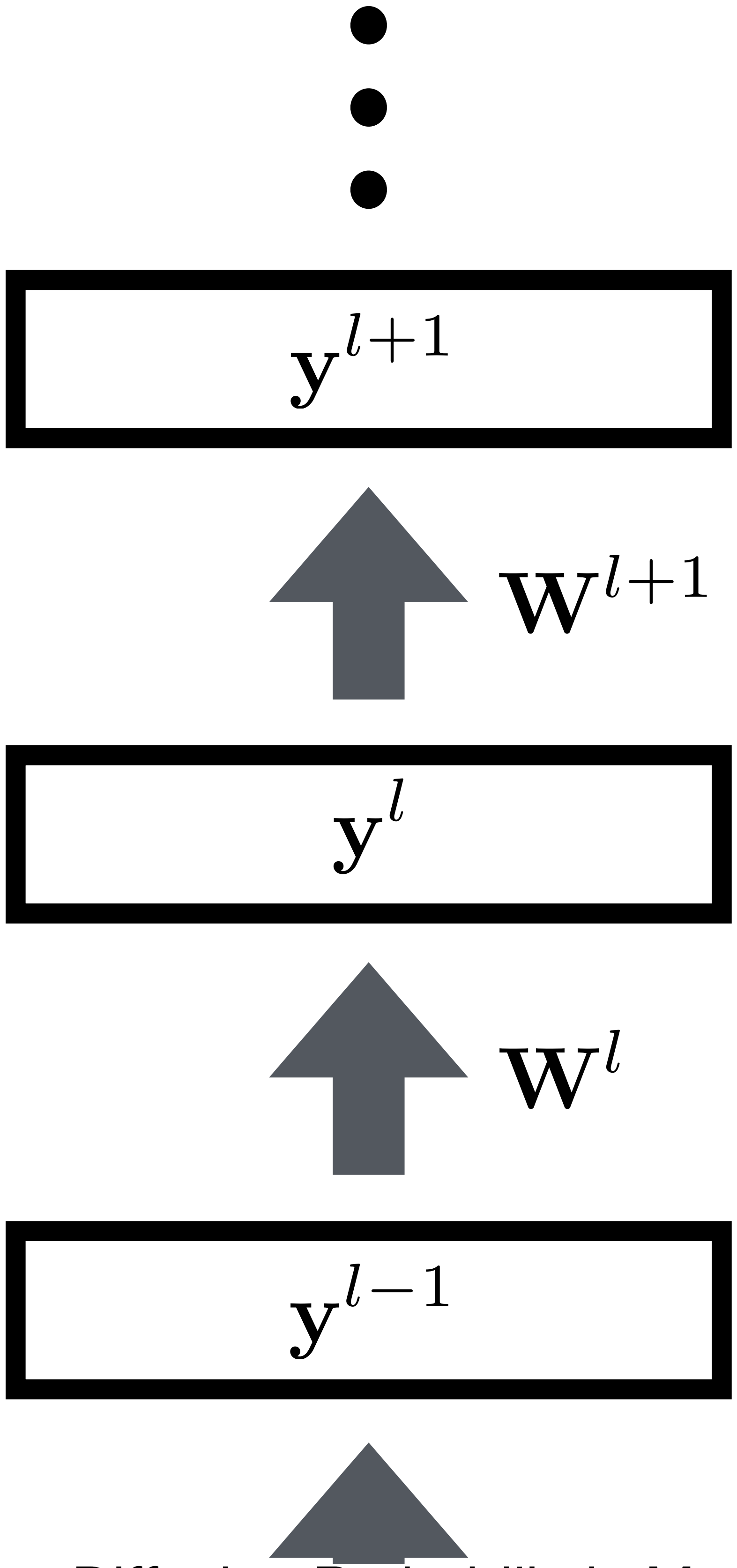
- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity

Deep Networks

- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity
- **Deep network:** stack single layers

Deep Networks

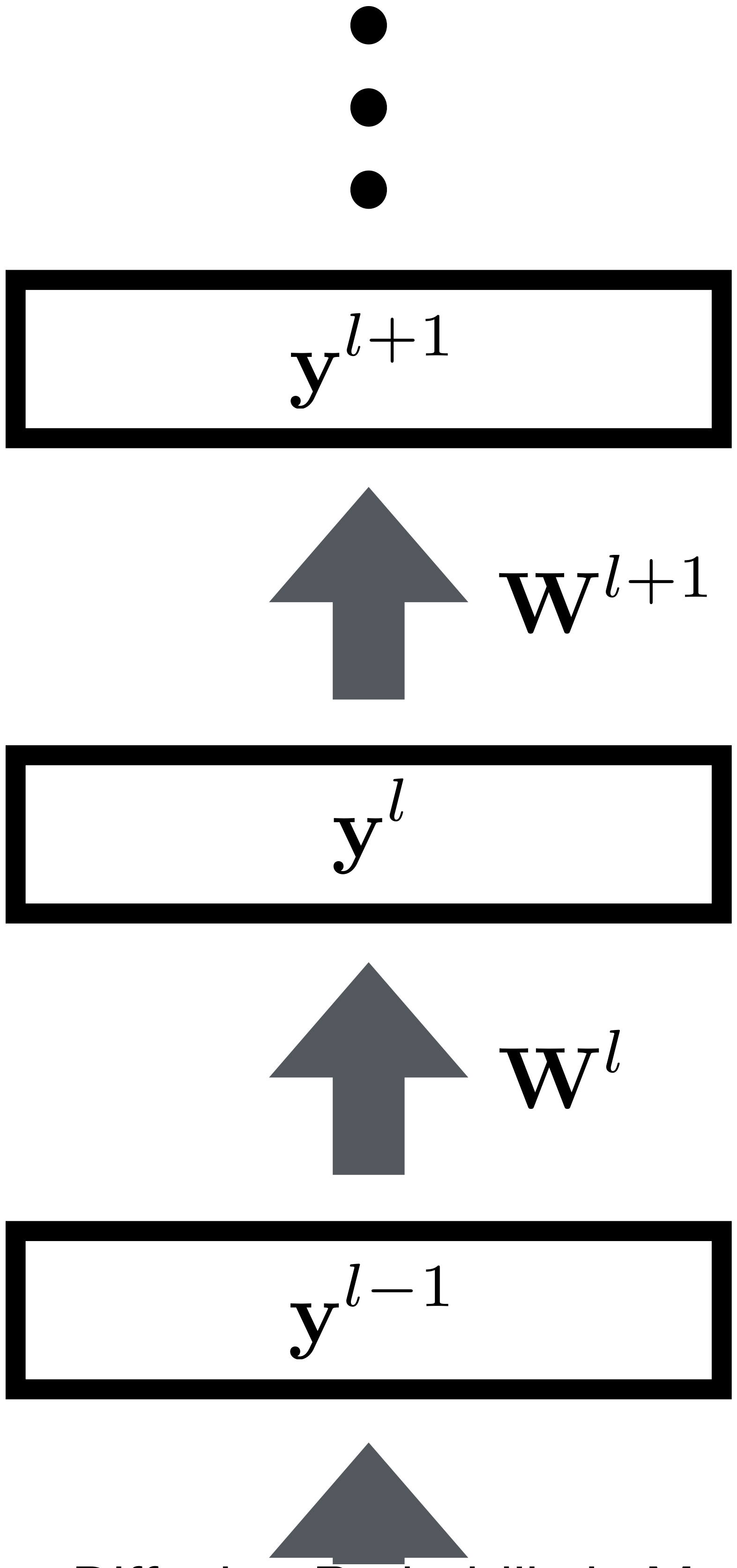
- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity
- **Deep network:** stack single layers



Deep Networks

- Extremely flexible, parametric, function approximation
- **Single layer:** linear transformation, pointwise nonlinearity
- **Deep network:** stack single layers

$$\mathbf{y}^L = \sigma\left(\mathbf{W}^L \sigma\left(\mathbf{W}^{L-1} \cdots \sigma\left(\mathbf{W}^1 \mathbf{y}^0\right)\right)\right)$$

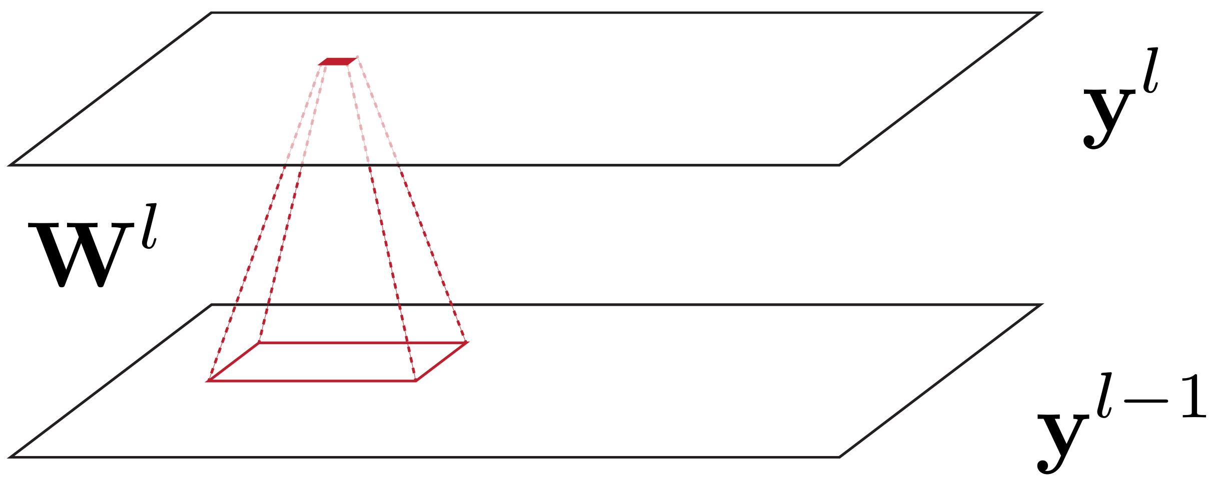


Convolutional Neural Network

- Single convolutional layer:
  - Same linear transform for every pixel
  - Pointwise nonlinearity

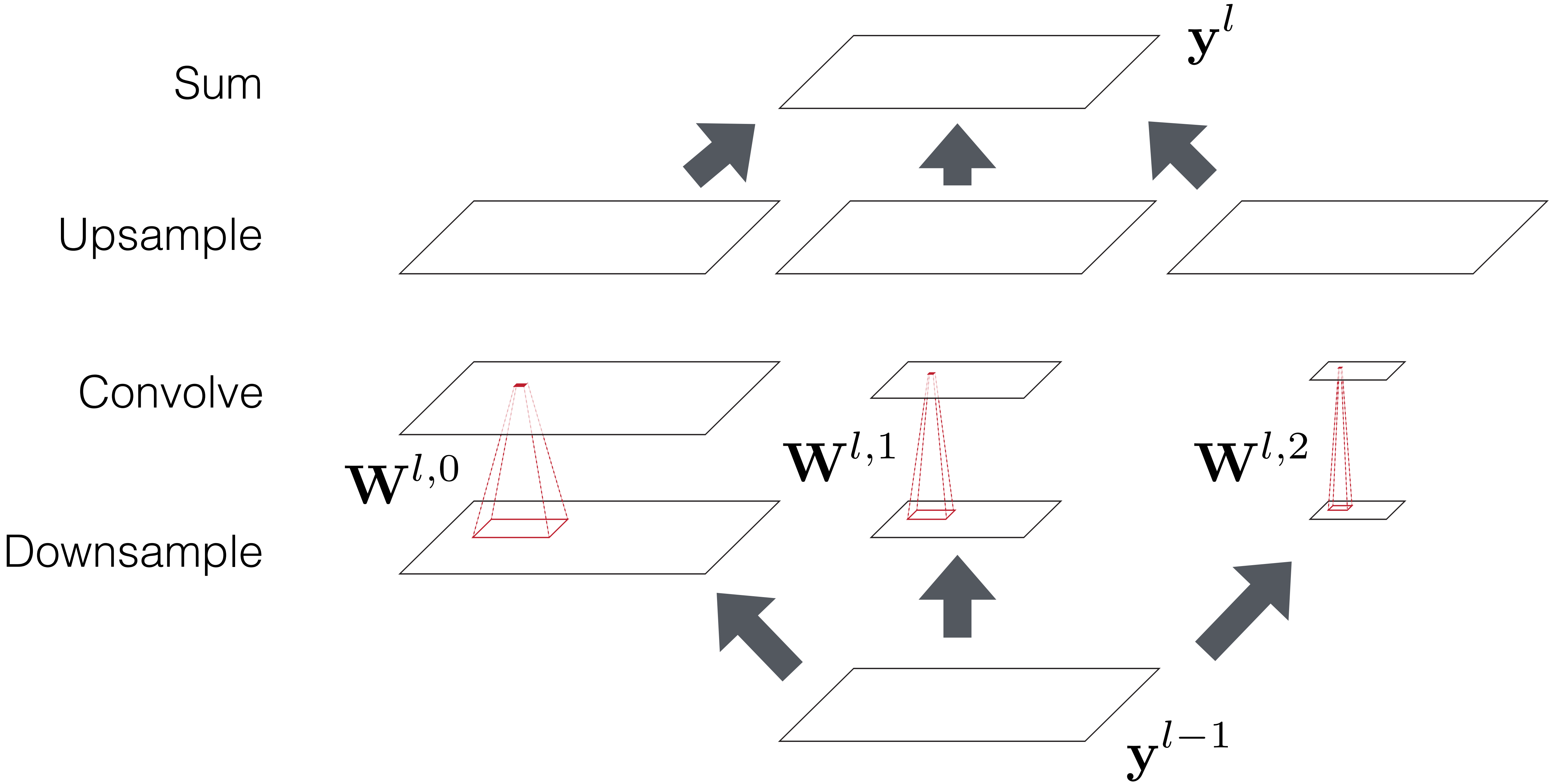
Convolutional Neural Network

- Single convolutional layer:
  - Same linear transform for every pixel
  - Pointwise nonlinearity



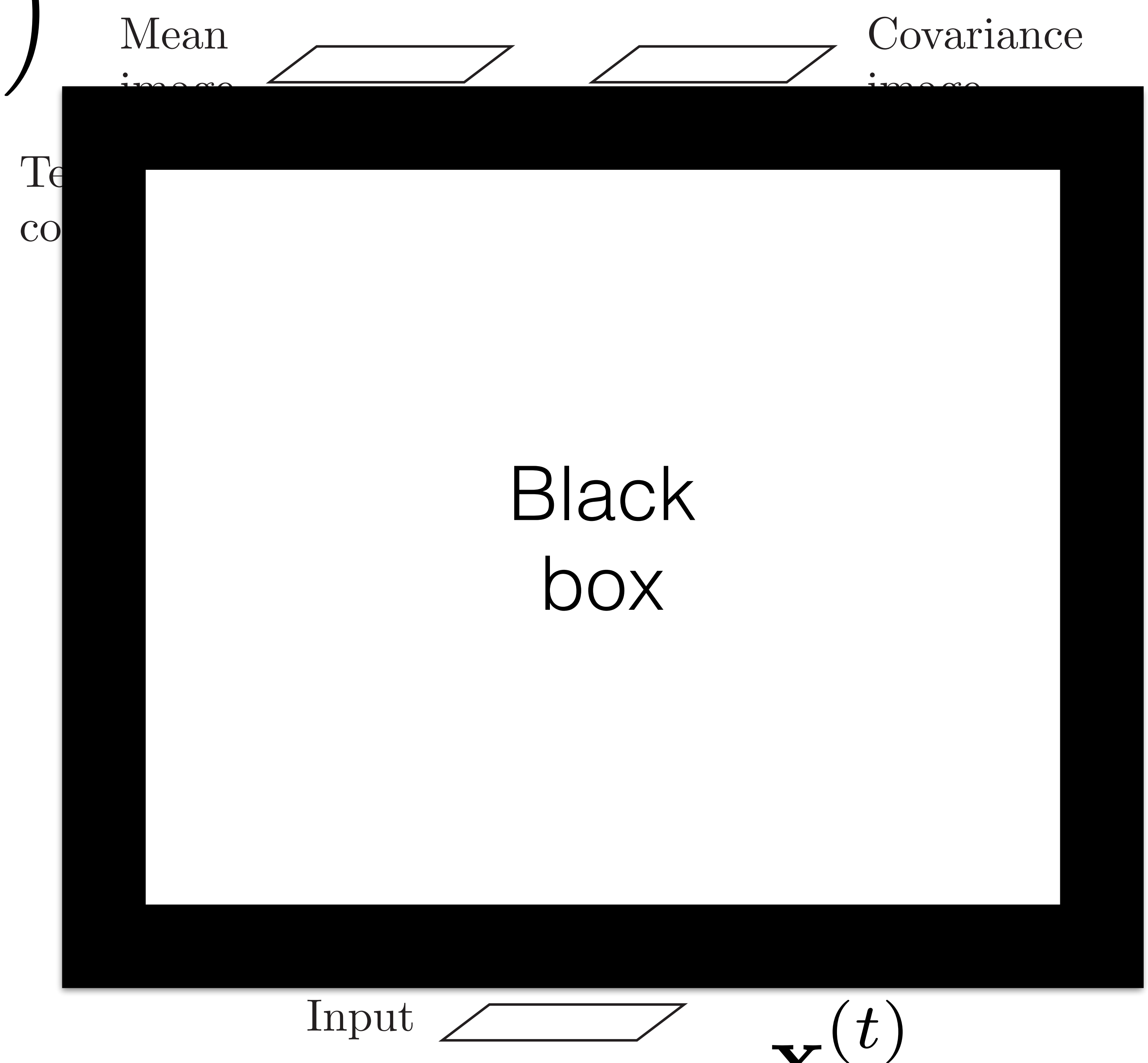
Multiscale Convolution

- Single multi-scale convolutional layer:



Deep Network Architecture for Diffusion

$f_\mu\left(\mathbf{x}^{(t)}, t\right)$ $f_\Sigma\left(\mathbf{x}^{(t)}, t\right)$

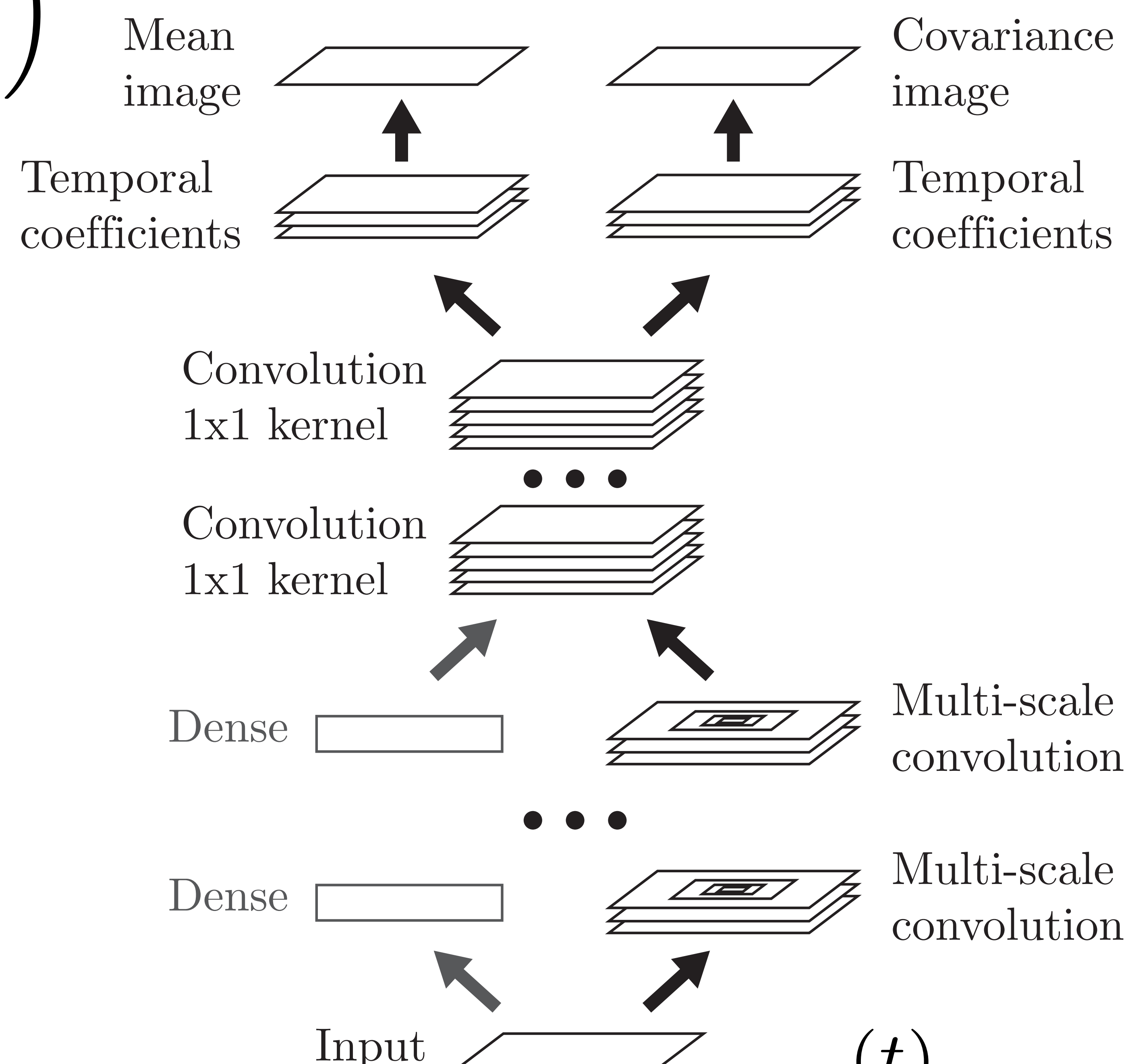


$\mathbf{x}^{(t)}$

Deep Network Architecture for Diffusion

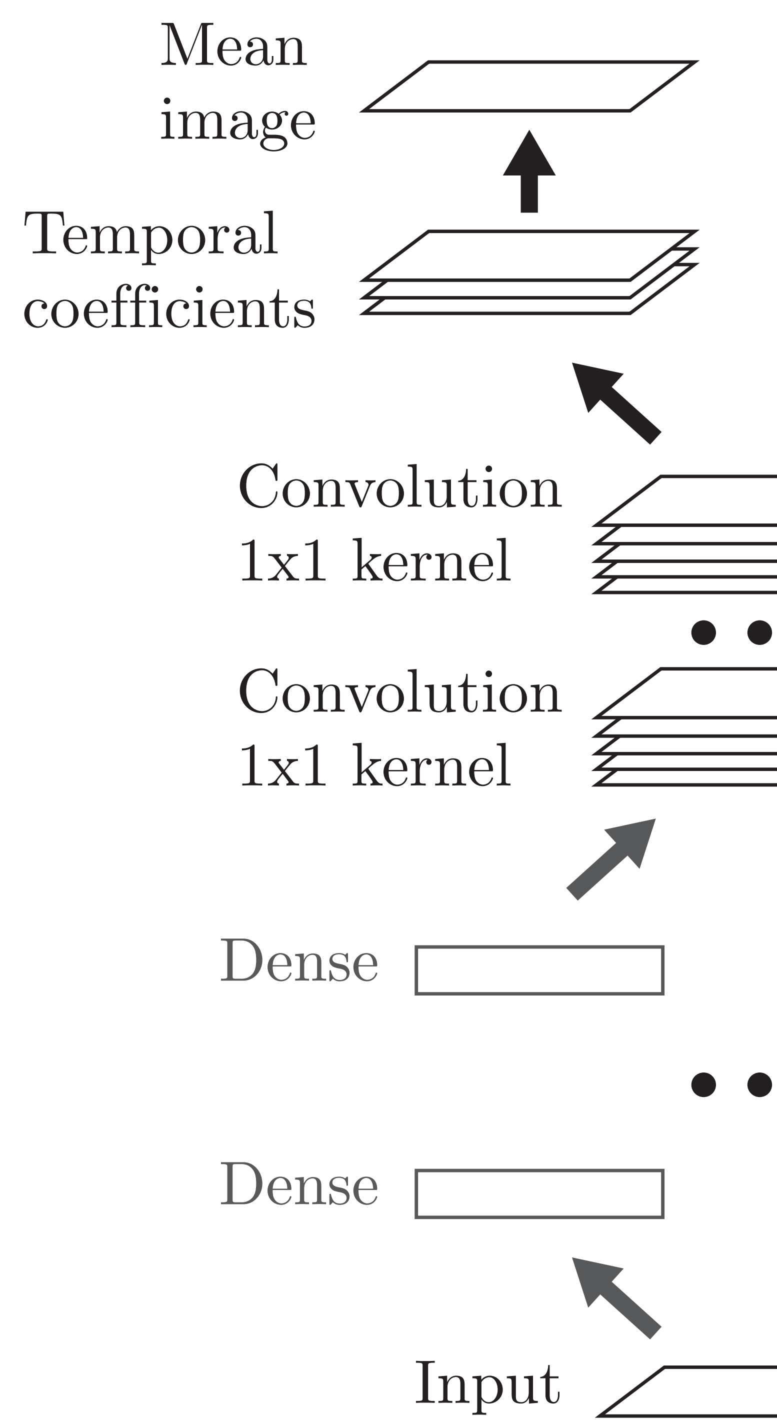
$f_\mu\left(\mathbf{x}^{(t)}, t\right)$

$f_\Sigma\left(\mathbf{x}^{(t)}, t\right)$

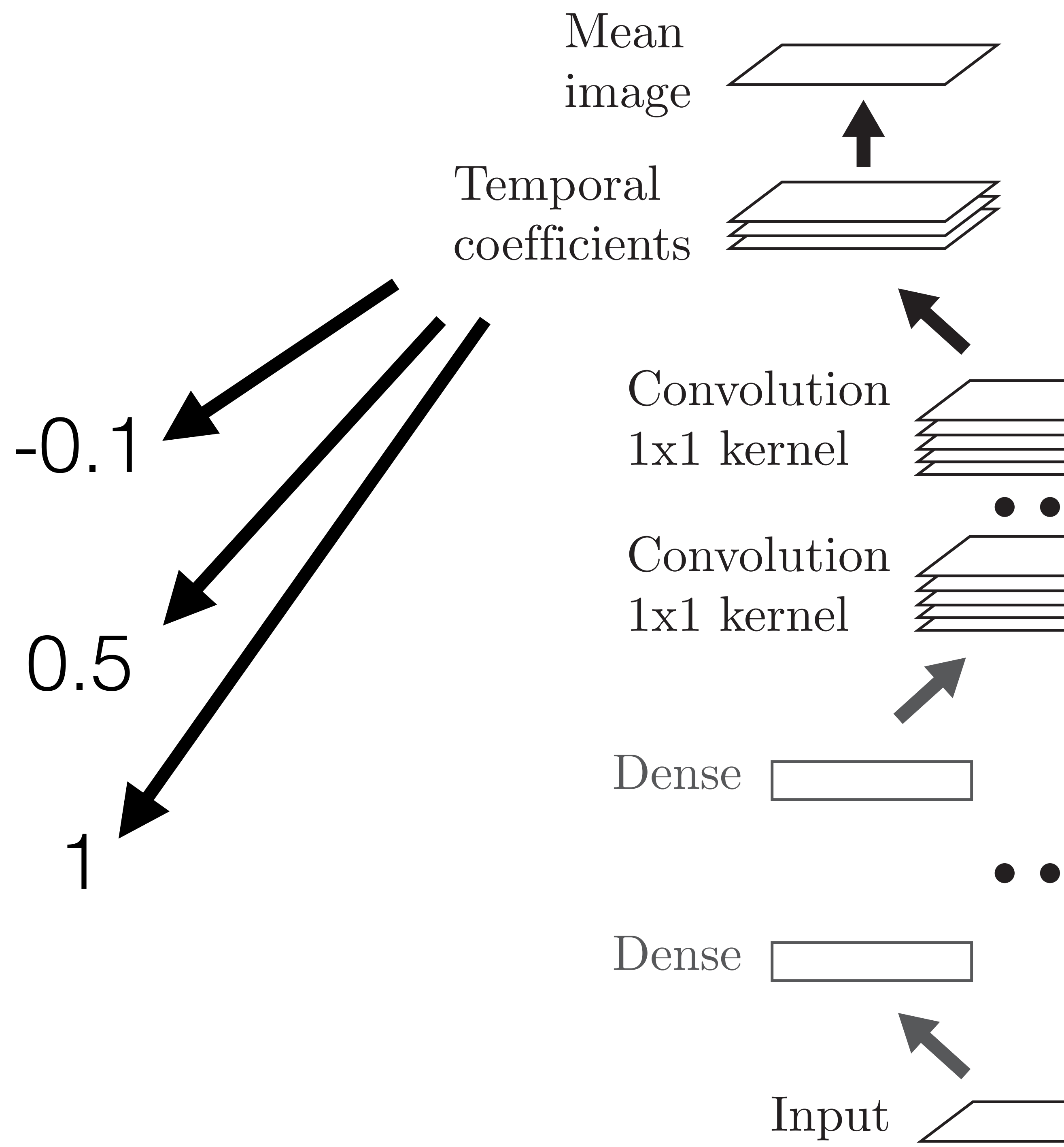


$\mathbf{x}^{(t)}$

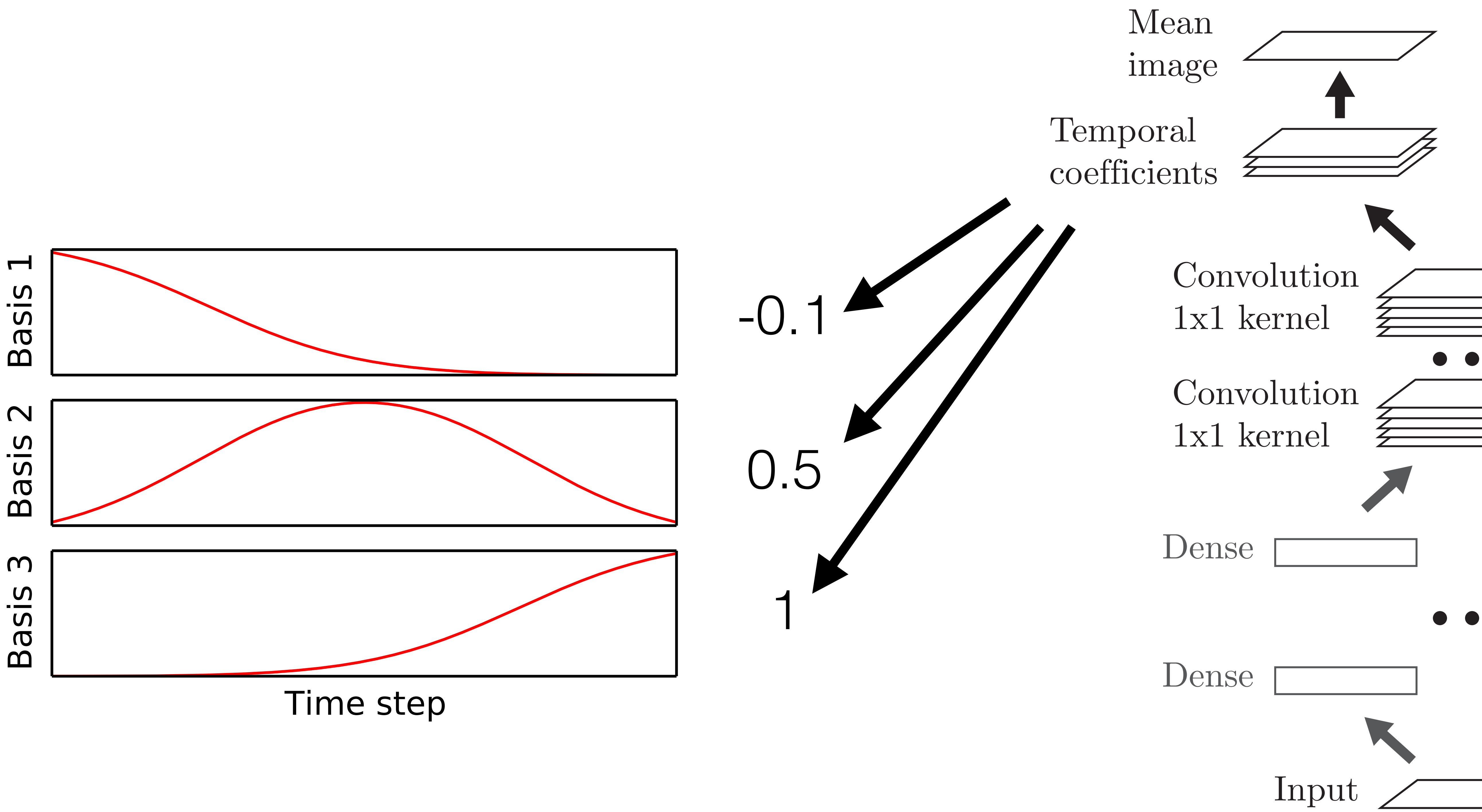
Time Dependence using Temporal Basis



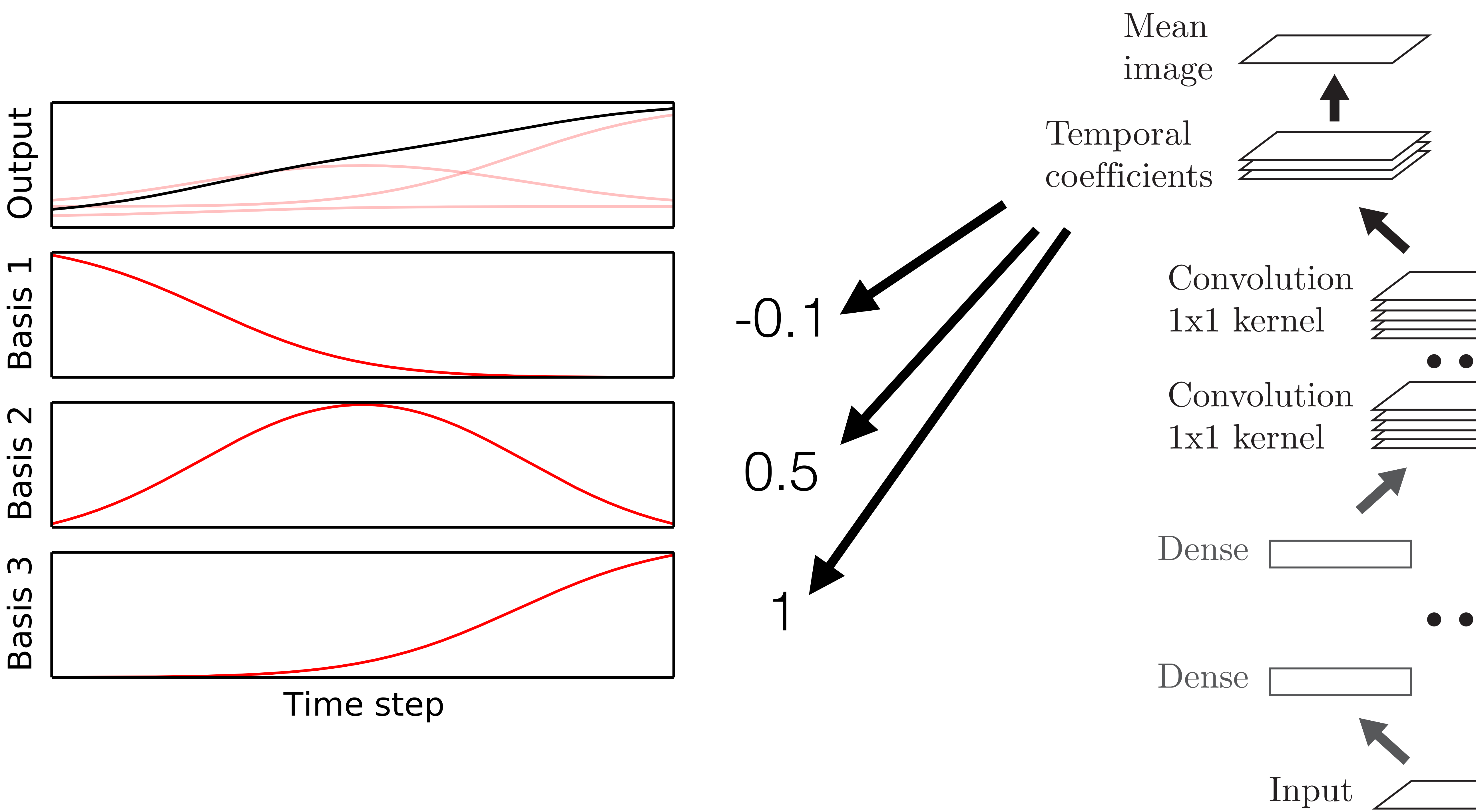
Time Dependence using Temporal Basis



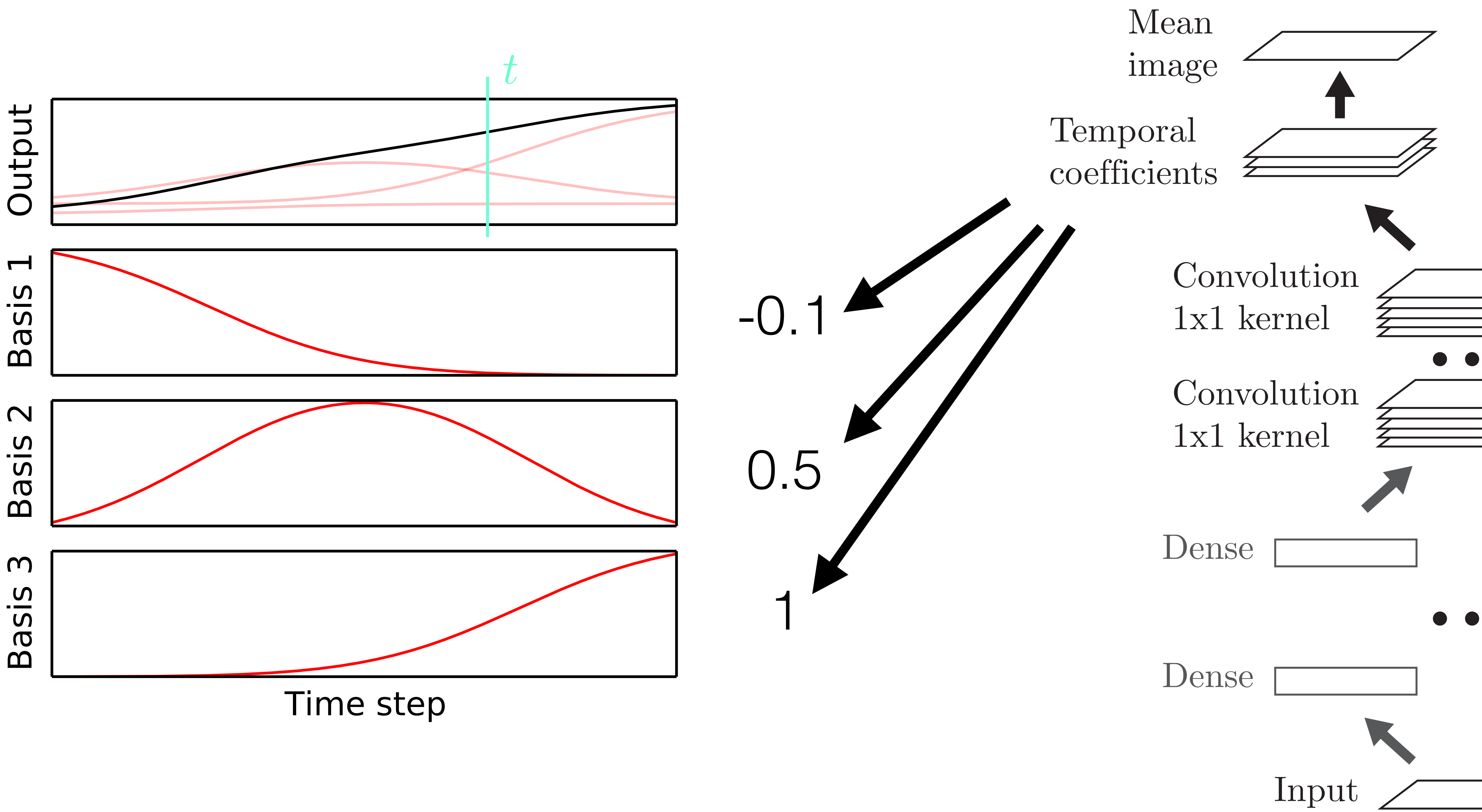
Time Dependence using Temporal Basis



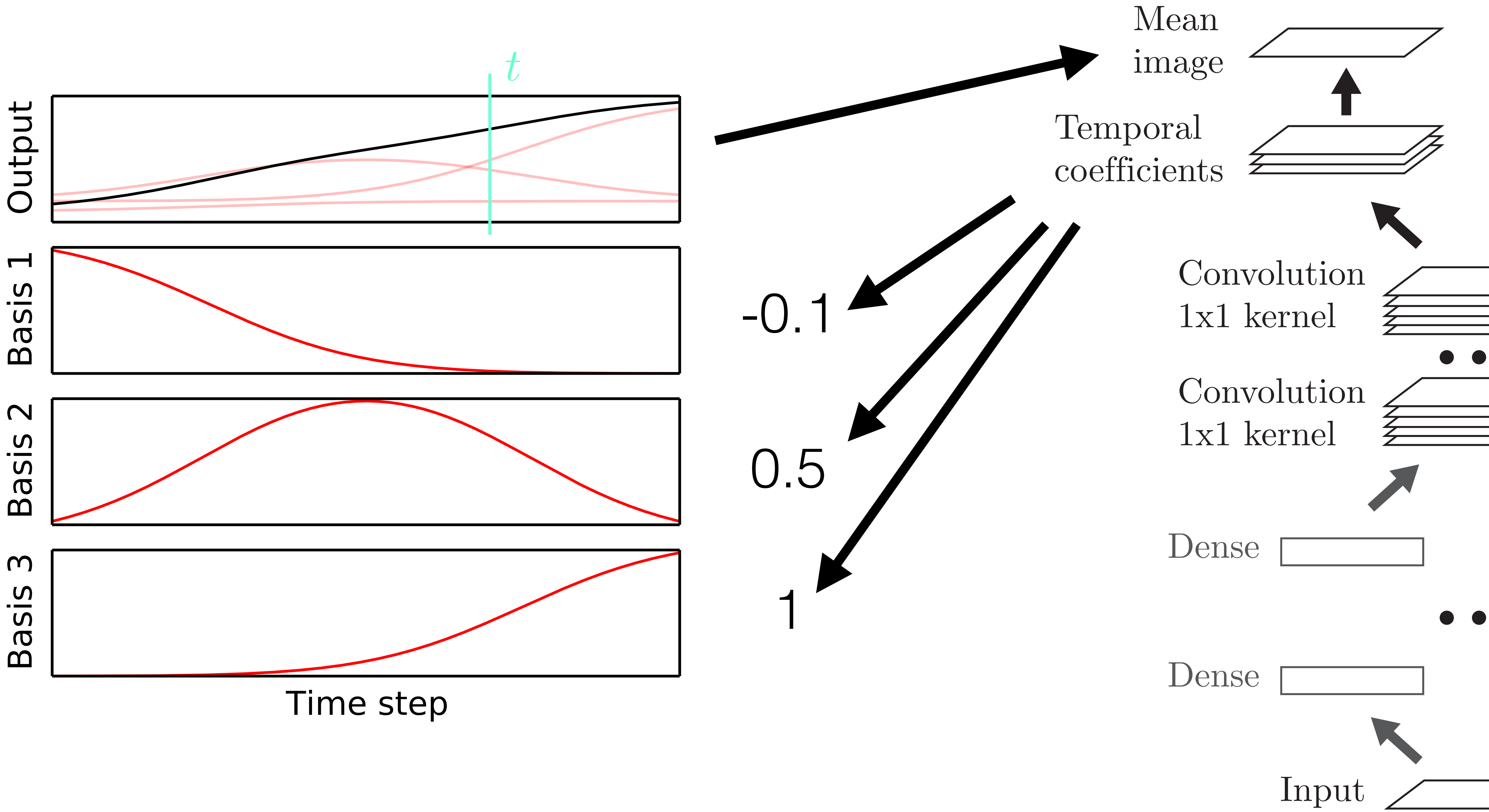
Time Dependence using Temporal Basis



Time Dependence using Temporal Basis



Time Dependence using Temporal Basis



Setting Diffusion Rate

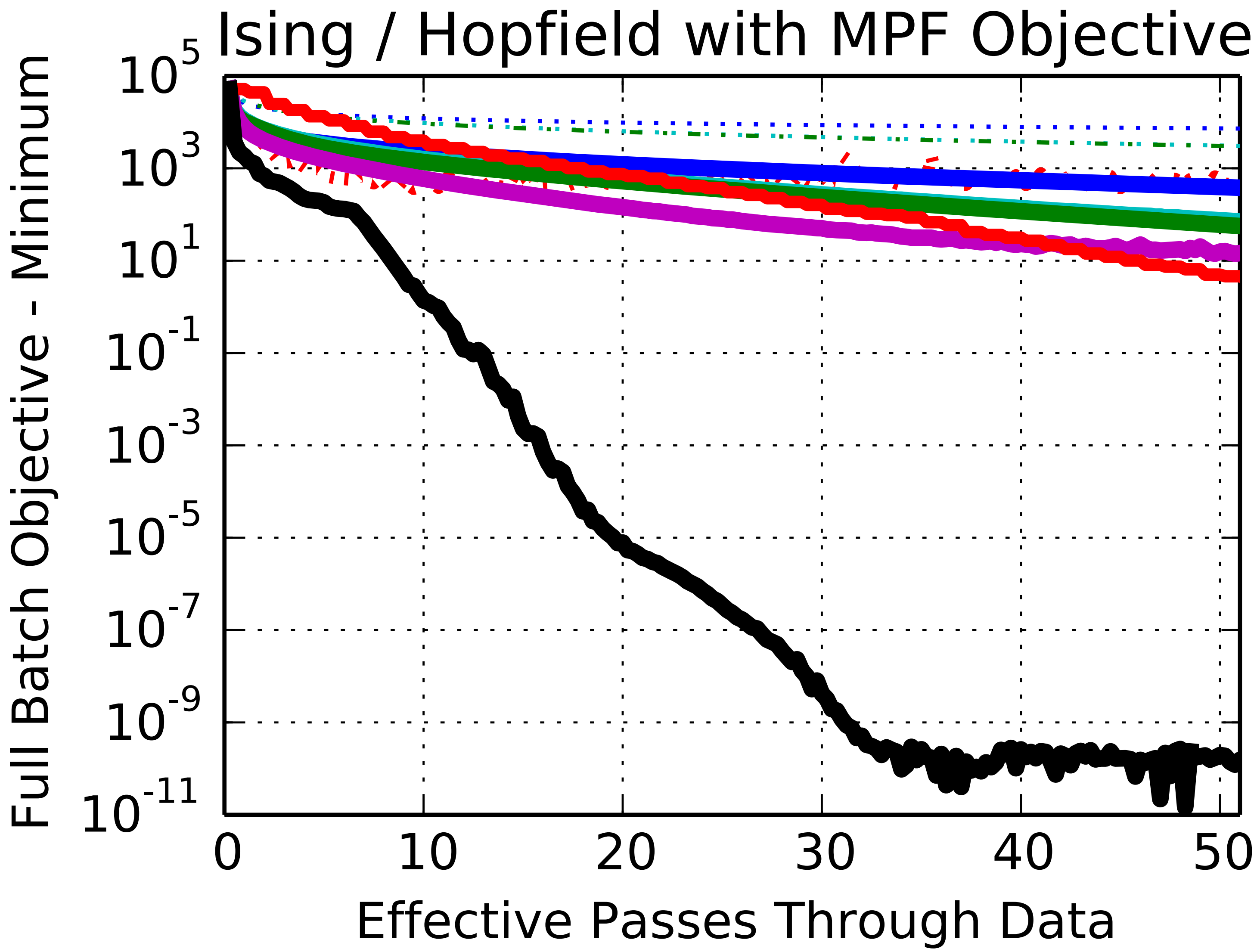
- Erase constant fraction of stimulus variance each step

$$\beta_t = \frac{1}{T - t + 1}$$

- Can also train β_t

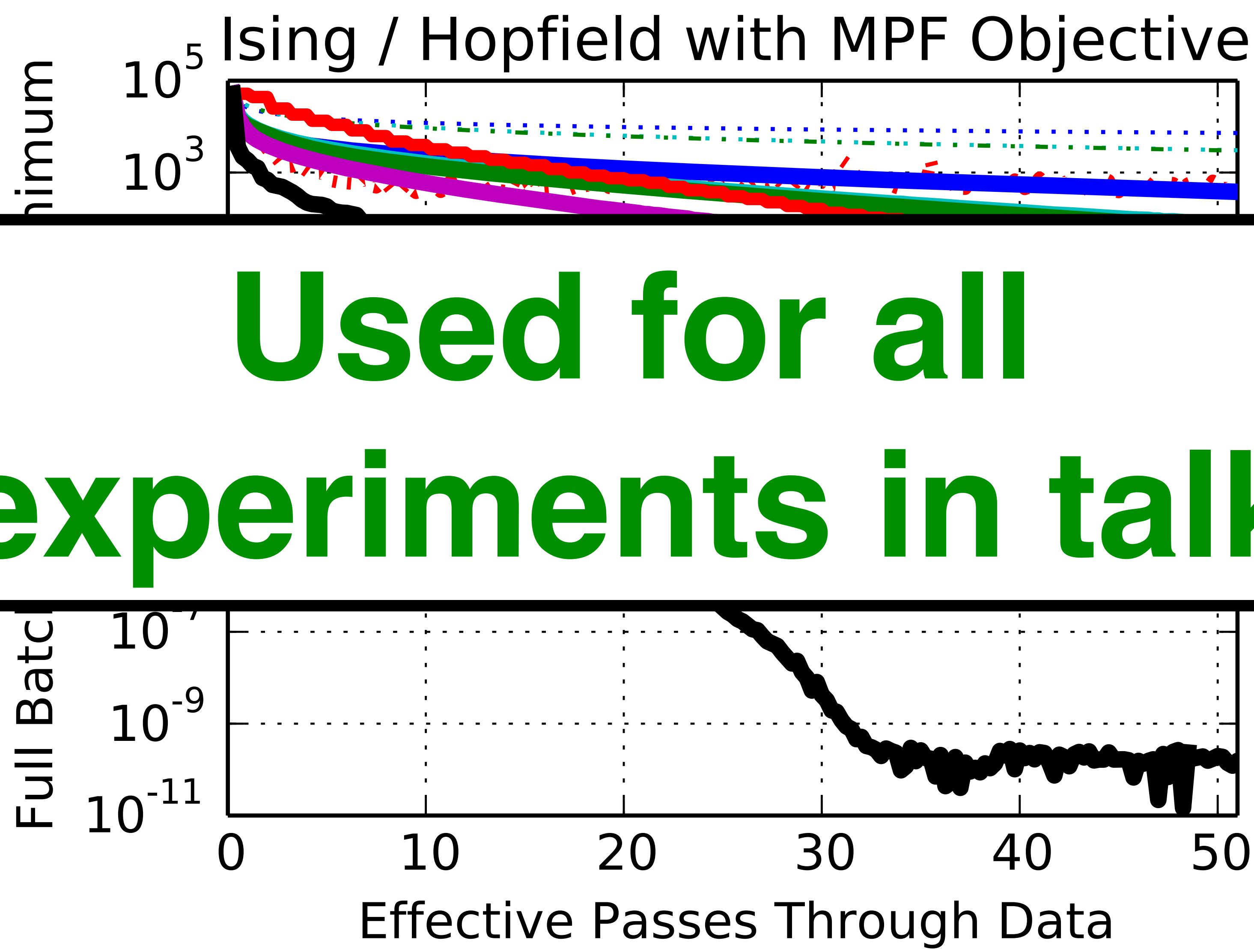
Theoretical Breakthroughs in Machine Learning

- Optimization: Combining SGD and quasi-Newton optimization (SFO optimizer) **[ICML 2014]**



Theoretical Breakthroughs in Machine Learning

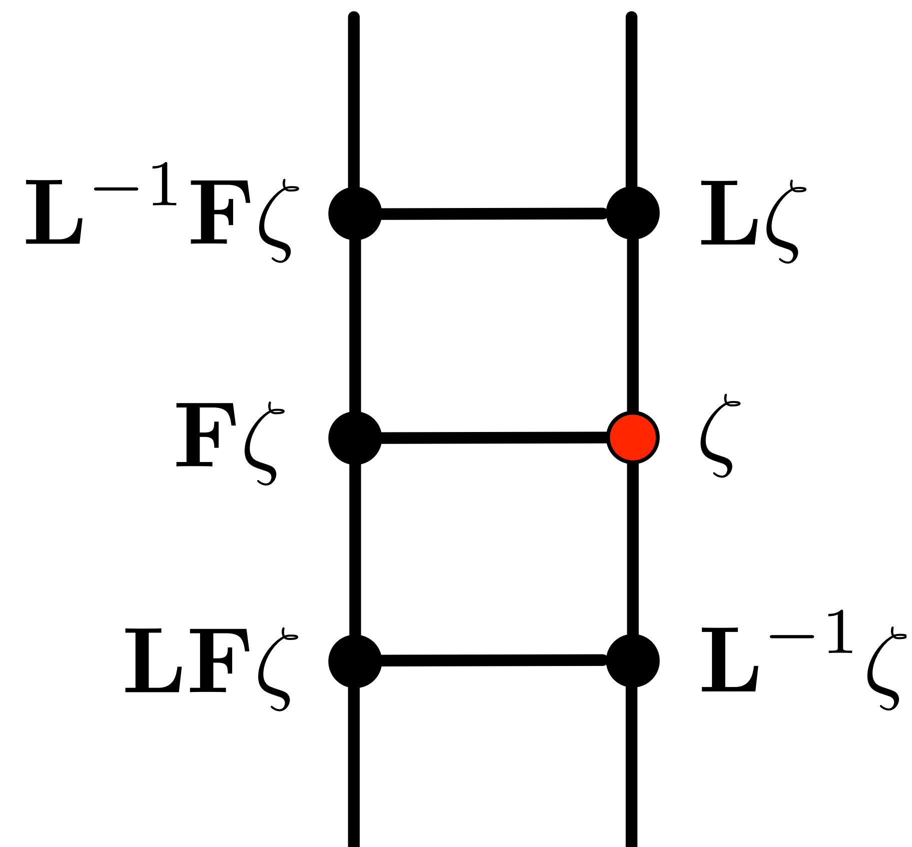
- Optimization: Combining SGD and quasi-Newton optimization (SFO optimizer) **[ICML 2014]**



Used for all experiments in talk

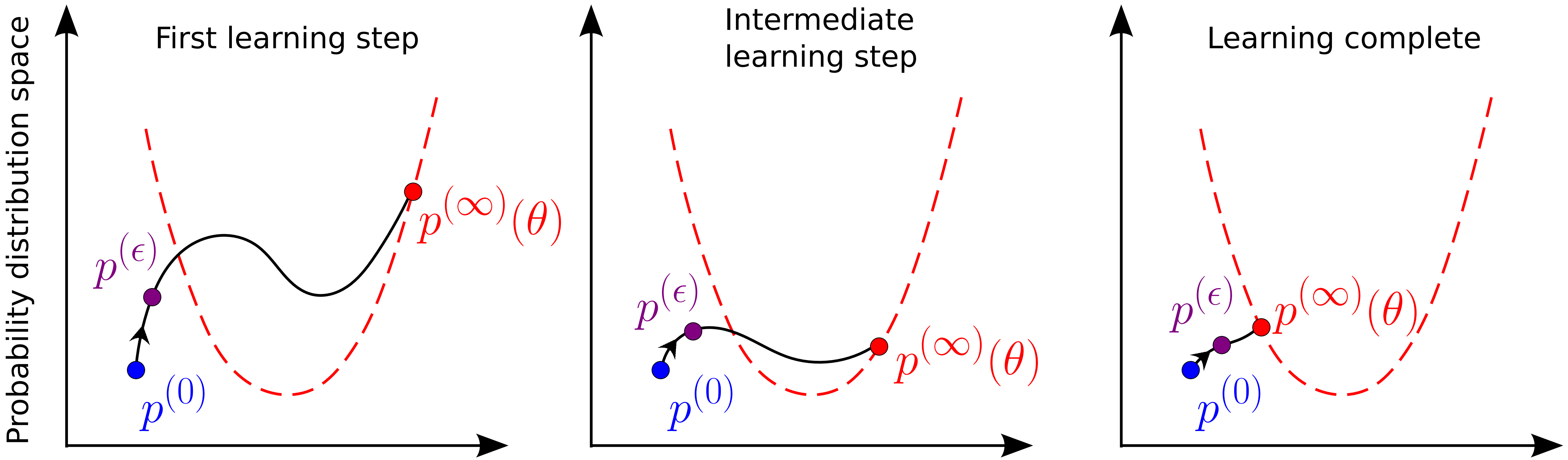
Theoretical Breakthroughs in Machine Learning

- Sampling and evaluation: Hamiltonian Monte Carlo without detailed balance **[ICML 2014]** and for log likelihood evaluation **[Tech Report 2012]**, fast sampling for natural image models **[NIPS 2012]**



Theoretical Breakthroughs in Machine Learning

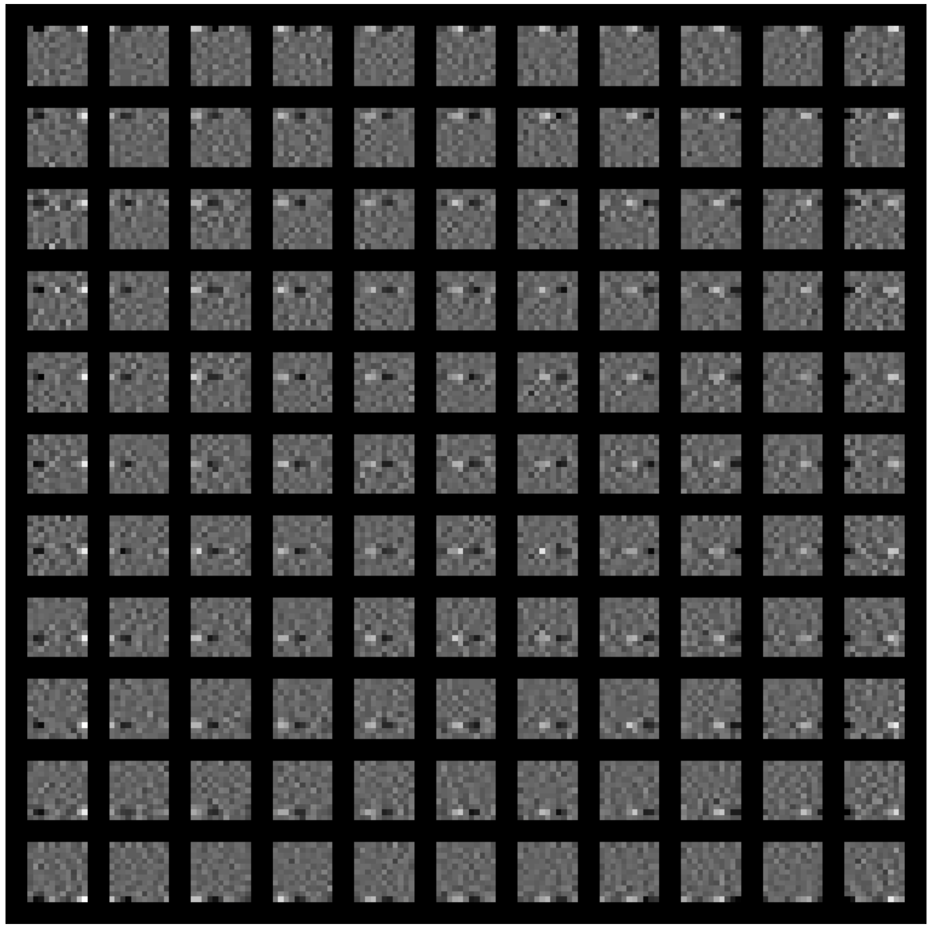
- Training energy-based models: Minimum Probability Flow learning **[ICML 2011] [PRL 2011]**

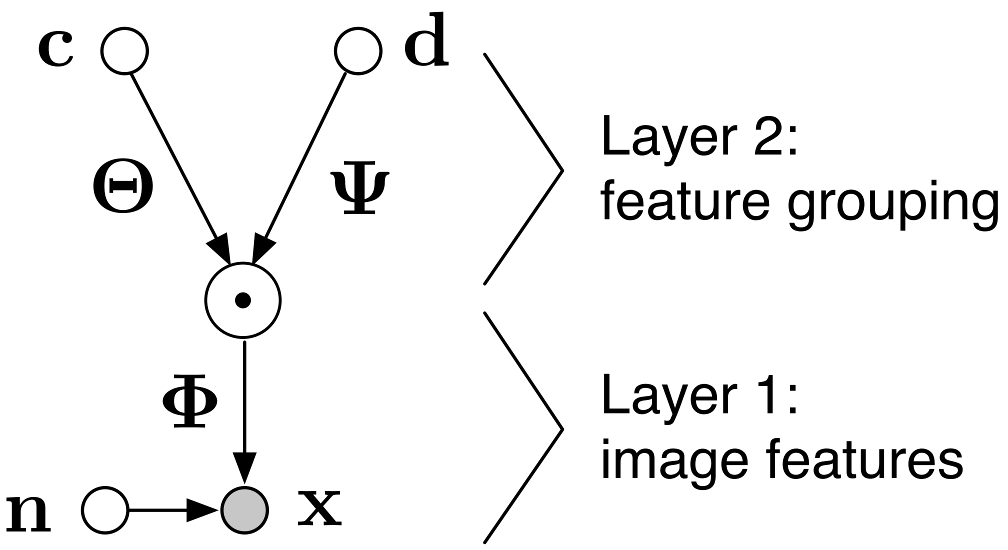


Theoretical Breakthroughs in Machine Learning

- Model design: capturing dynamics with Lie groups **[Under Revision at NECO]**, bilinear generative models **[ICCV 2011]**

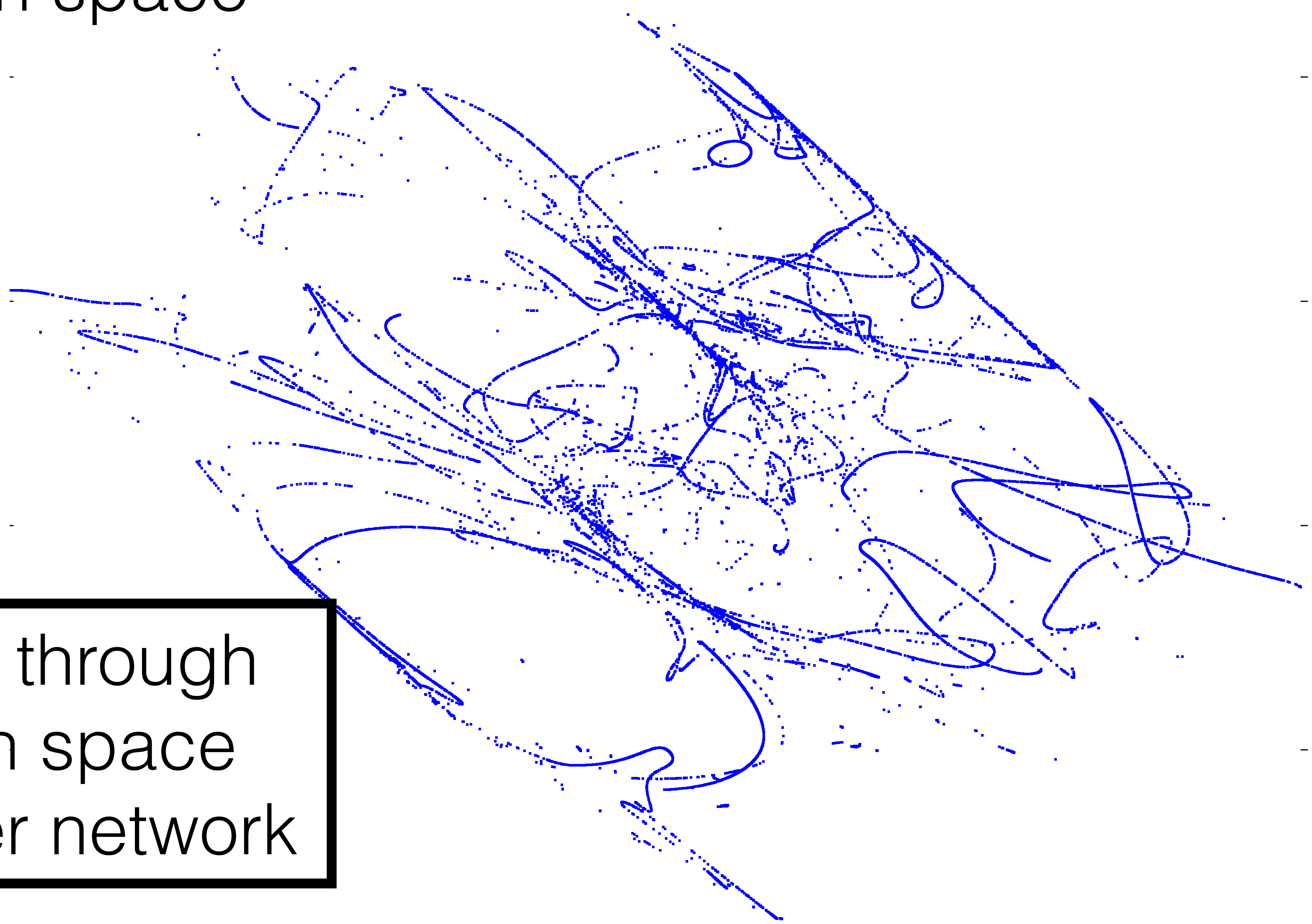
Horizontal Translation





Theoretical Breakthroughs in Machine Learning

- Properties of deep networks: Characterization in function space

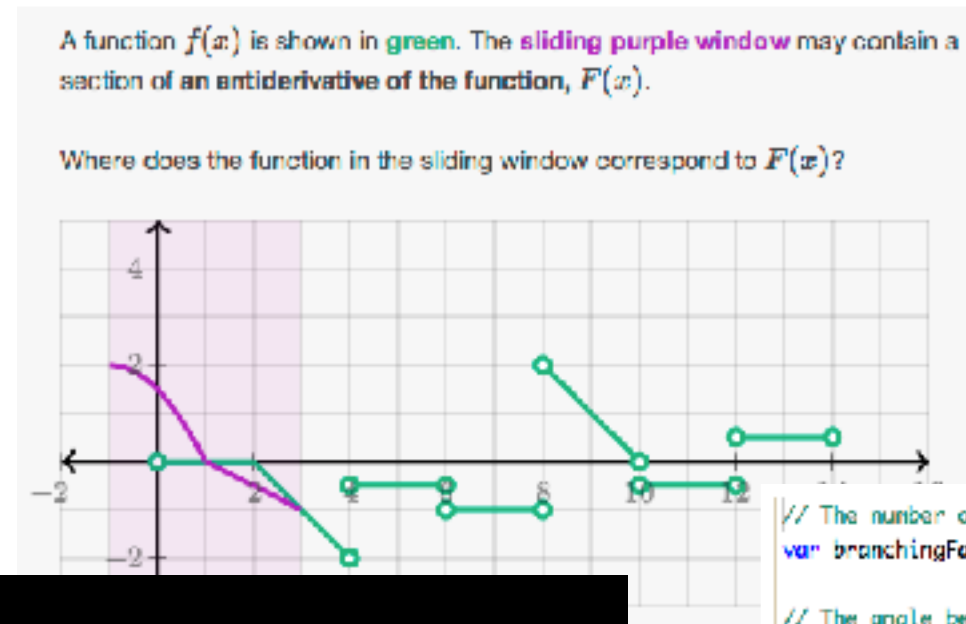


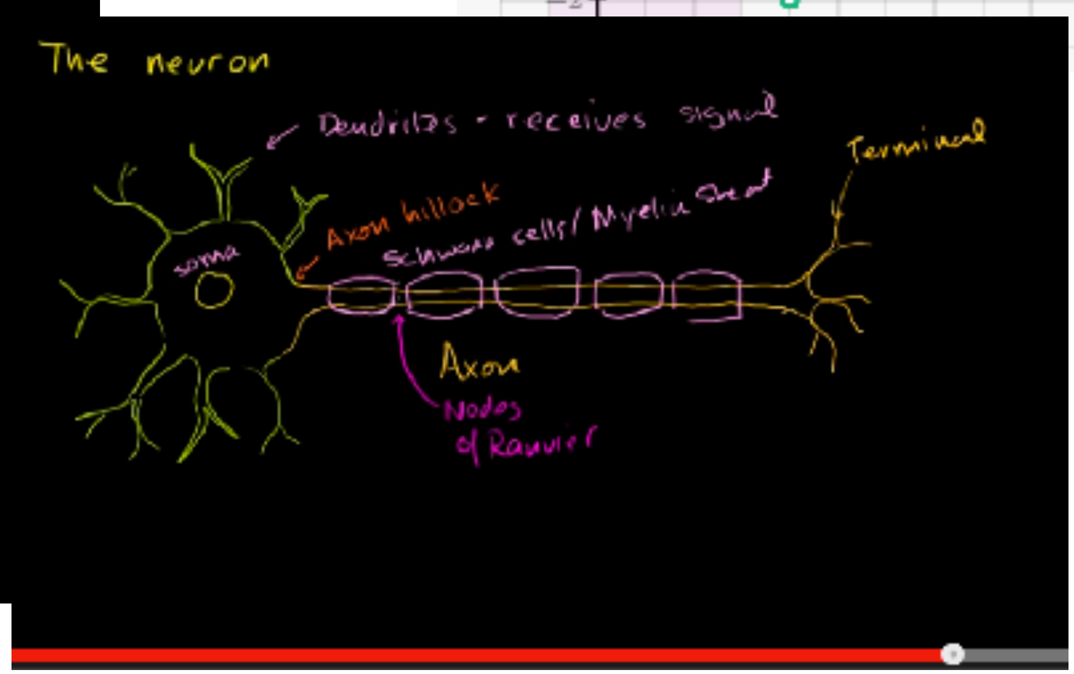
2d slice through function space for 2-layer network

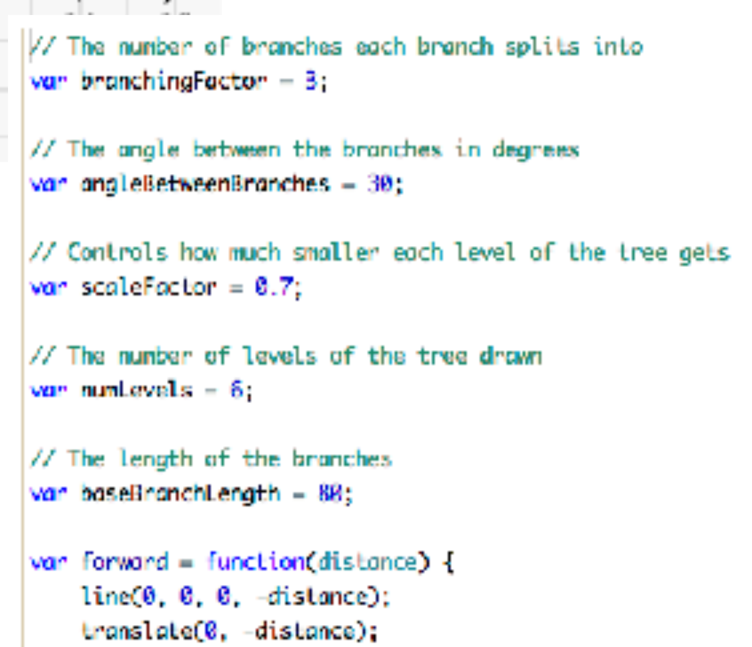
Understanding Real Data

- Online education data

KHAN ACADEMY

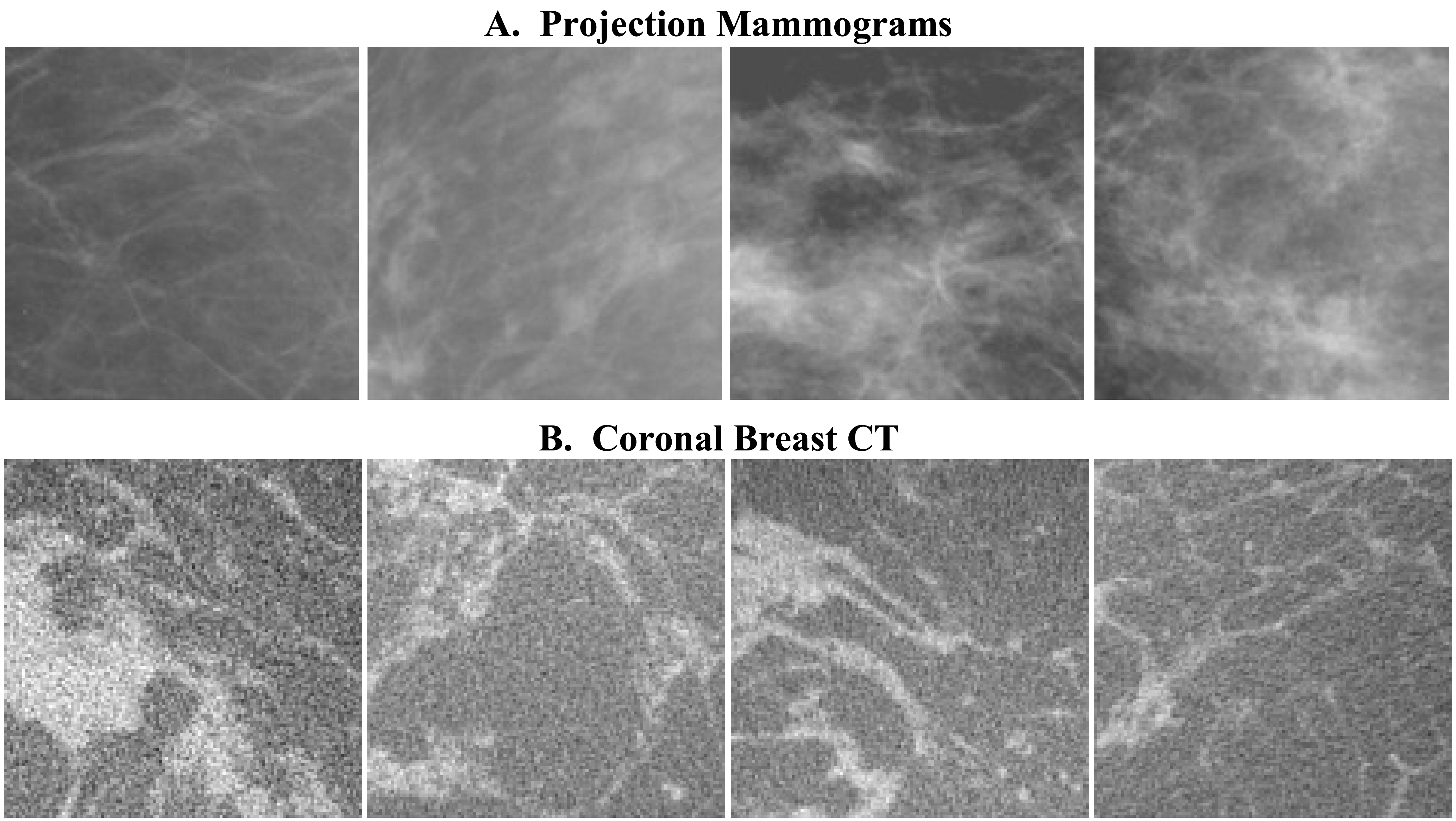






Understanding Real Data

- Medical imaging data **[SPIE 2009] [Med Phys 2014]**

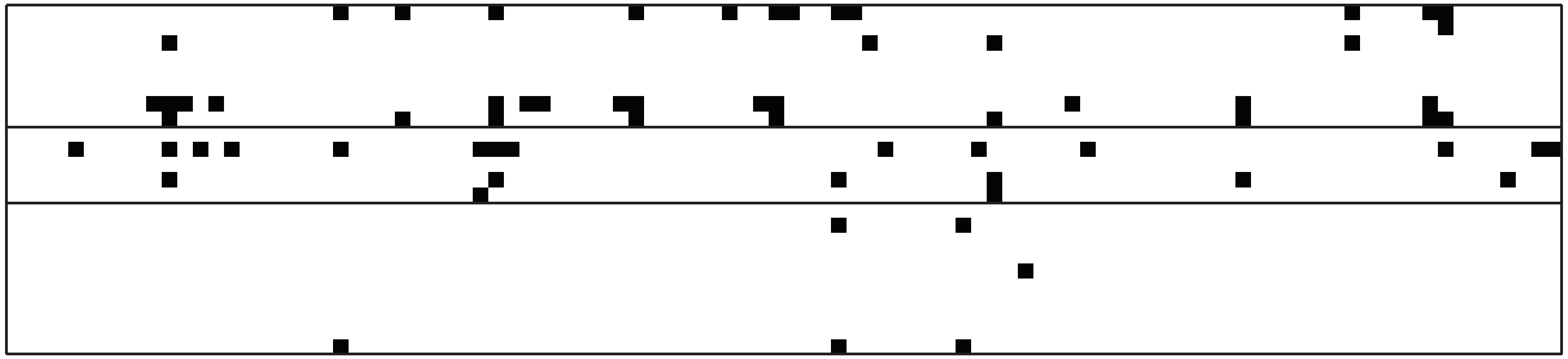


Understanding Real Data

- Neuroscience electrophysiology data: **[PLoS Comp Bio 2014] [Neuron 2013]**

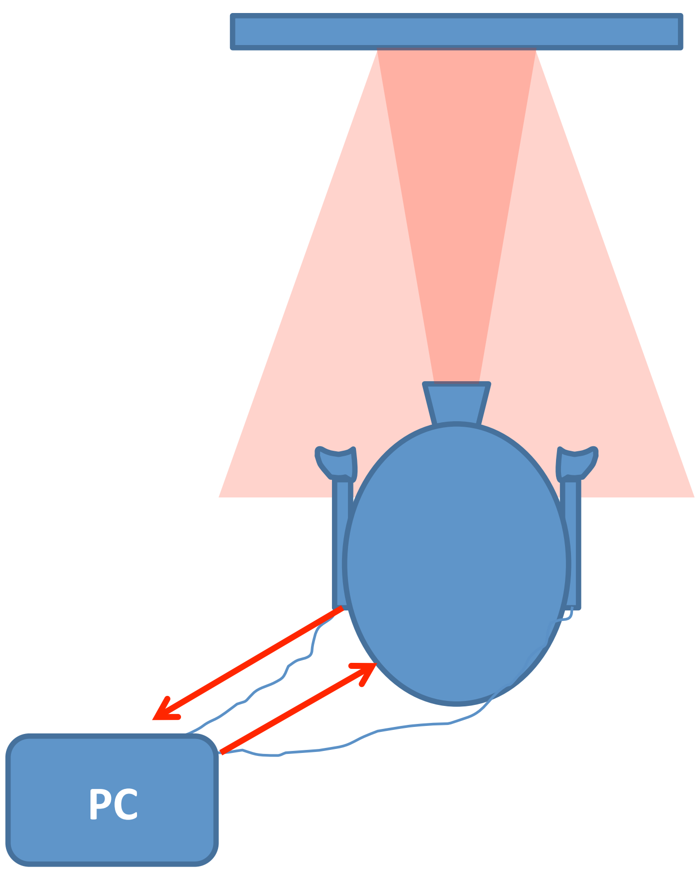
a) Stimulus frames

b) Example data, 2s of data in 20ms bins



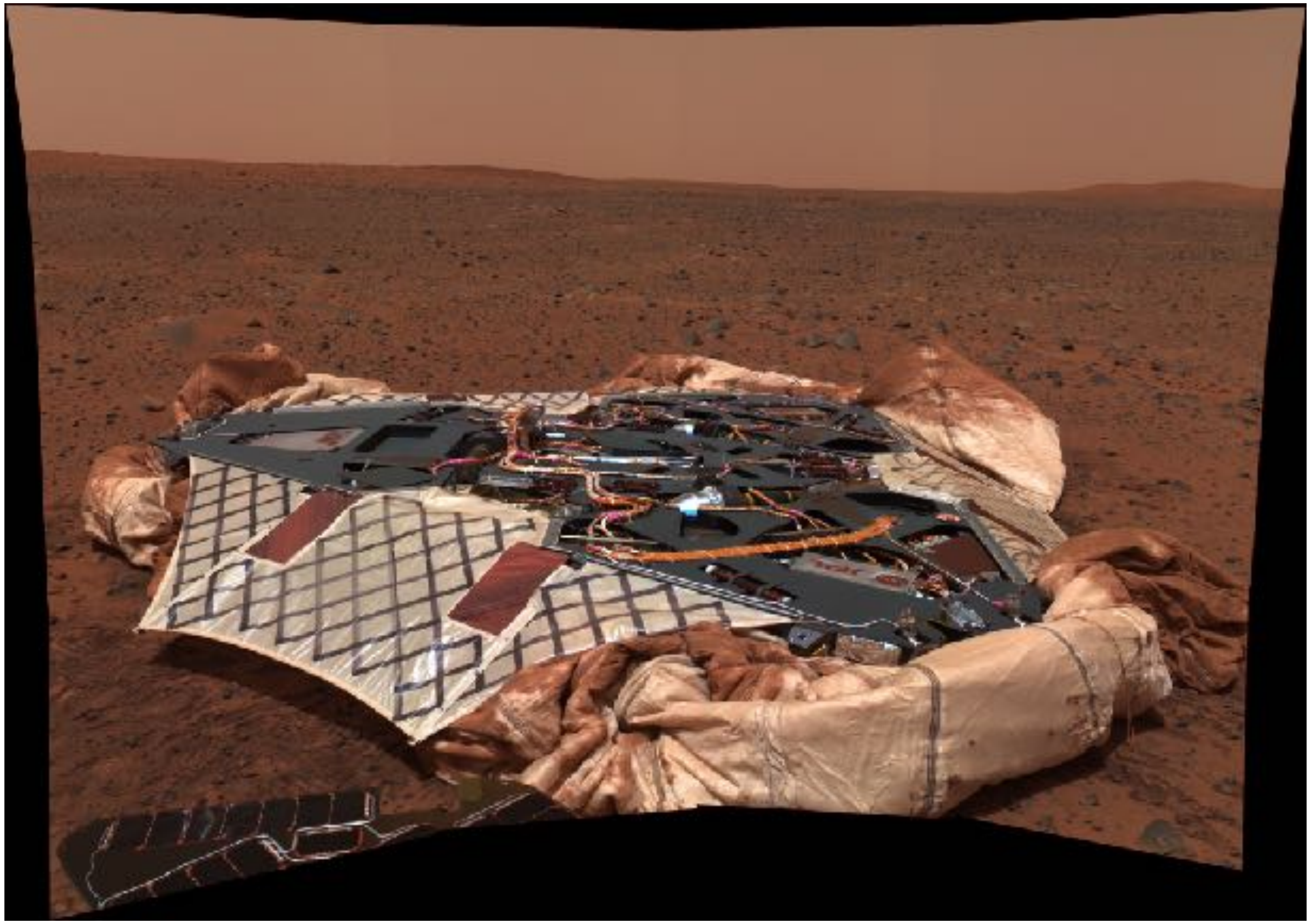
Understanding Real Data

- Human ultrasonic echolocation: Blind assistive device
  [TBME 2015]



Understanding Real Data

- Planetary science: multispectral observations
  [Science 2004a] [Science 2004b]



Thanks!

Collaborators

- Craig Abbey
- Peter Battaglino
- Shaowen Bao
- Matthias Bethge
- Jack Culpepper
- Liberty Hamilton
- Chris Hillar
- Alex Huth
- Kilian Koepsell
- Urs Köster
- Niru Maheswaranathan
- Mayur Mudigonda
- Ben Poole
- Lucas Theis
- Jimmy Wang
- Eric Weiss

Mentors

- Surya Ganguli
- Bruno Olshausen
- Michael R. DeWeese
- James F. Bell III

Endless discussion

- The Redwood Center for Theoretical Neuroscience
- The Ganguli Gang

Eric Weiss

Niru Maheswaranathan

Surya Ganguli

Differences from Variational Autoencoders

- Can analytically evaluate KL divergence between steps in forward and reverse trajectories.
- Can multiply with other distributions, and compute posteriors
- Erases structure, rather than transforming it
- Thousands of layers or time steps, rather than only a small handful
- Connections to nonequilibrium statistical mechanics

Continuous Time

$$q\left(\mathbf{x}^t|\mathbf{x}^0,\mathbf{x}^{t+dt}\right) = \mathcal{N}\left(\mathbf{x}^t;\mathbf{x}^{t+dt} - \mathbf{x}^{t+dt}\frac{\exp\left(-\beta t\right)}{1-\exp\left(-\beta t\right)}\beta dt - \frac{1}{2}\mathbf{x}^{t+dt}\beta dt + \frac{1}{2}\mathbf{x}^0 \operatorname{csch}\left(\frac{\beta t}{2}\right)\beta dt, \beta dt\right)$$

$$p\left(\mathbf{x}^t|\mathbf{x}^{t+dt}\right) = \mathcal{N}\left(\mathbf{x}^t;\mathbf{x}^{t+dt} - \mathbf{x}^{t+dt}\frac{\exp\left(-\beta t\right)}{1-\exp\left(-\beta t\right)}\beta dt - \frac{1}{2}\mathbf{x}^{t+dt}\beta dt + \frac{1}{2}f_0\left(\mathbf{x}^{t+dt},t\right) \operatorname{csch}\left(\frac{\beta t}{2}\right)\beta dt, \beta dt\right)$$

$$\begin{aligned} D_{KL}\left(q\left(\mathbf{x}^t|\mathbf{x}^0,\mathbf{x}^{t+dt}\right)||p\left(\mathbf{x}^t|\mathbf{x}^{t+dt}\right)\right) &= \frac{1}{2}\frac{\Sigma_q}{\Sigma_p} + \frac{1}{2}\log\frac{\Sigma_p}{\Sigma_q} + \frac{1}{2}\frac{\left(\mu_p-\mu_q\right)^2}{\Sigma_p} - \frac{1}{2} \\ &= \frac{1}{8}\left(f_0\left(\mathbf{x}^{t+dt},t\right) - \mathbf{x}^0\right)^2 \operatorname{csch}^2\left(\frac{\beta t}{2}\right)\beta dt \end{aligned}$$

Denoising autoencoder penalty

Related Methods

- Generative stochastic networks
- Variational autoencoders
- (Deep) (Recurrent) Neural Autoregressive Distribution Estimators

- Variational Bayesian(e.g. variational autoencoder)
  - Posterior over intermediate layers has analytic form — > KL divergence has analytic form
  - Can multiply distributions
  - Generative model is small perturbation around inference model — makes learning easier
  - Models have *thousands* of layers (or time steps)