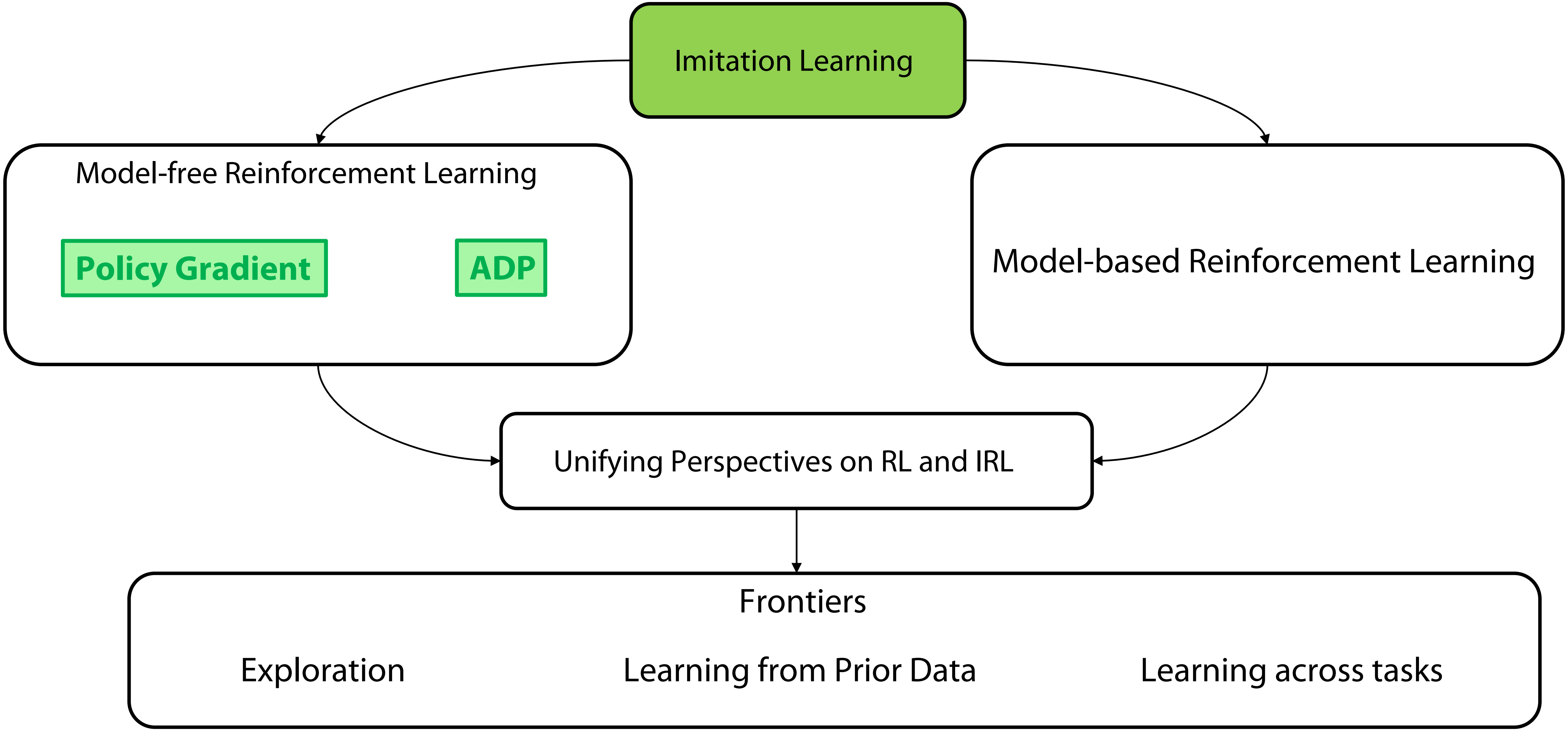
Reinforcement Learning
Spring 2024

Abhishek Gupta

TAs: Patrick Yin, Qiuyu Chen

Class Structure



Past Lecture Outline

The Anatomy of Model-Based Reinforcement Learning

↓

Model based RL v0 → random shooting + MPC

↓

Model based RL v1 → MPPI + MPC

↓

Model based RL v2 → uncertainty based models

↓

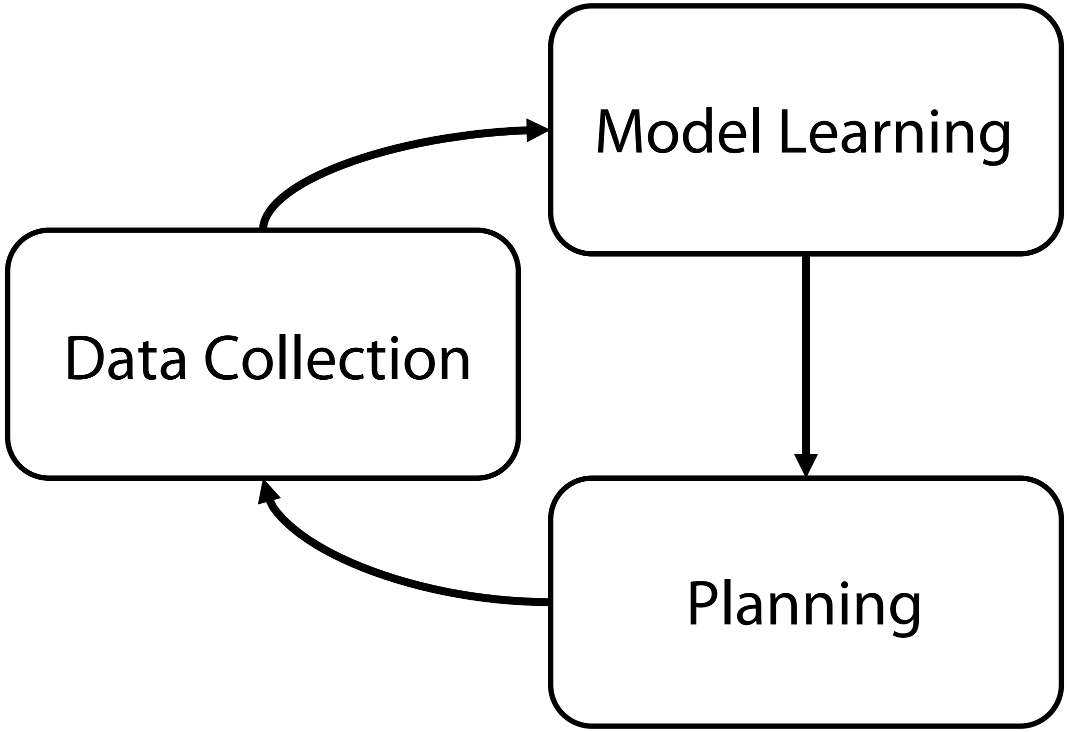
Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

Model Based RL v0 – Random Shooting + MPC

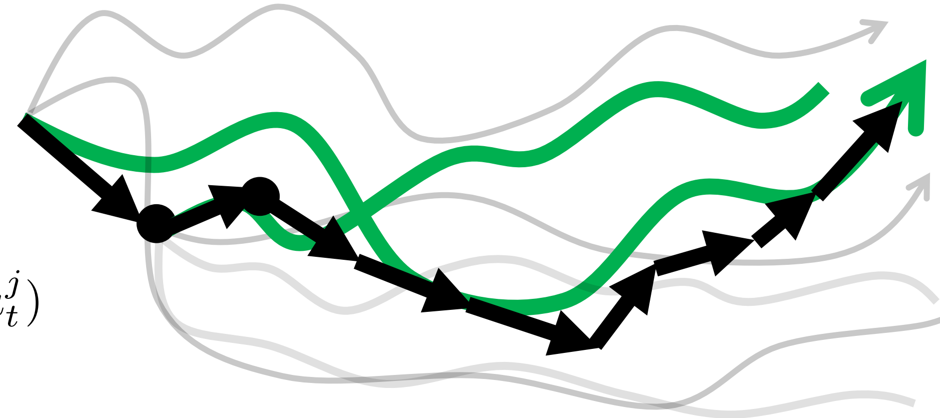
Data collected from planner



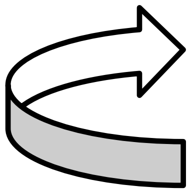
Maximum likelihood supervised Learning

$$\max_{\theta} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$

Planning with Shooting + MPC



Better than open loop planning because of feedback

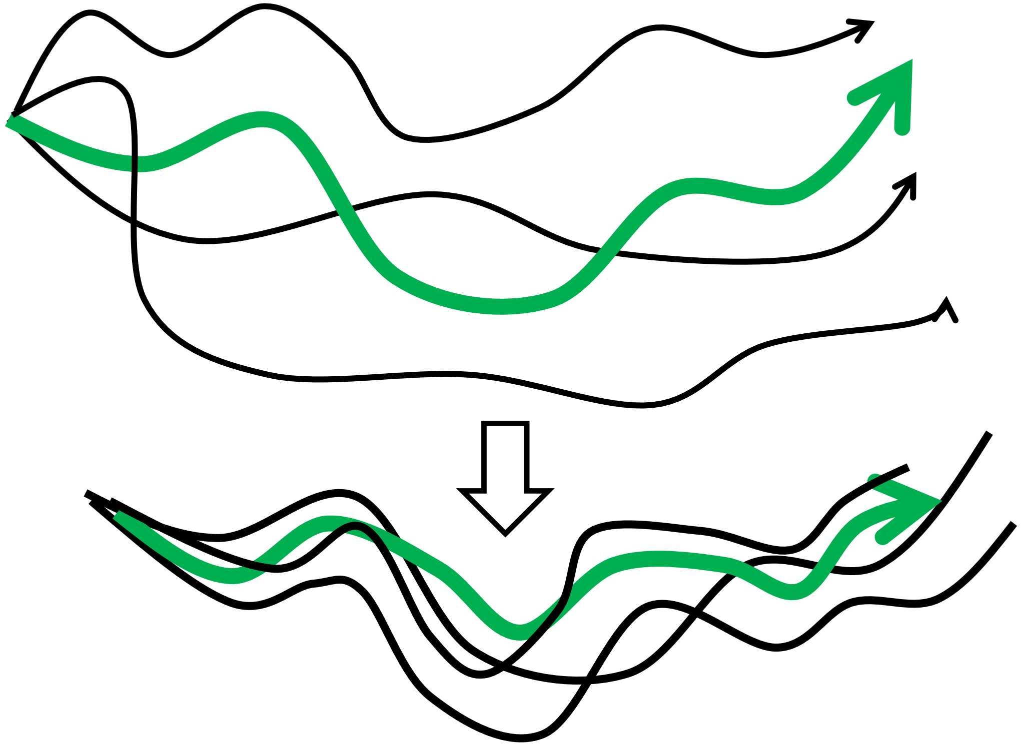


$$\arg \max_{a_0^j, a_1^j, \ldots, a_T^j} \sum_{t=0}^{T} r(\hat{s}_t^j, a_t^j)$$

$$\hat{s}_{t+1}^j \sim \hat{p}_\theta(.|\hat{s}_t^j, a_t^j)$$

Model Based RL v1 – MPPI

Idea: Iteratively upweight sampling distribution around the things that are higher returns



Referred to as **MPPI**, lower variance!

Sample N action sequences

$$(a_0^j, a_1^j, \ldots, a_T^j)_{j=1}^N \sim p(a)$$

Sample trajectories using these action sequences with the model $\hat{p}_\theta$

$$\hat{s}_{t+1} \sim \hat{p}_\theta(.|\hat{s}_t, a_t)$$

Update action sampler by upweighting high return actions

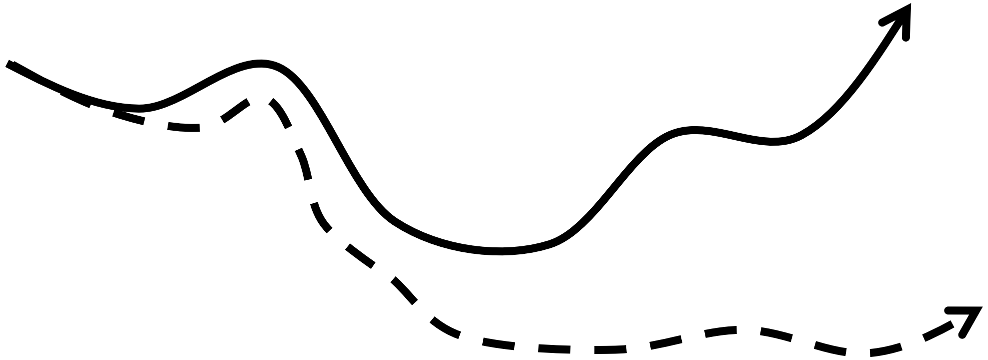
$$p(a) \leftarrow p(a)\frac{\exp(\sum_t r(s_t, a_t))}{Z}$$

Model Based RL v2 – Uncertainty Aware Models

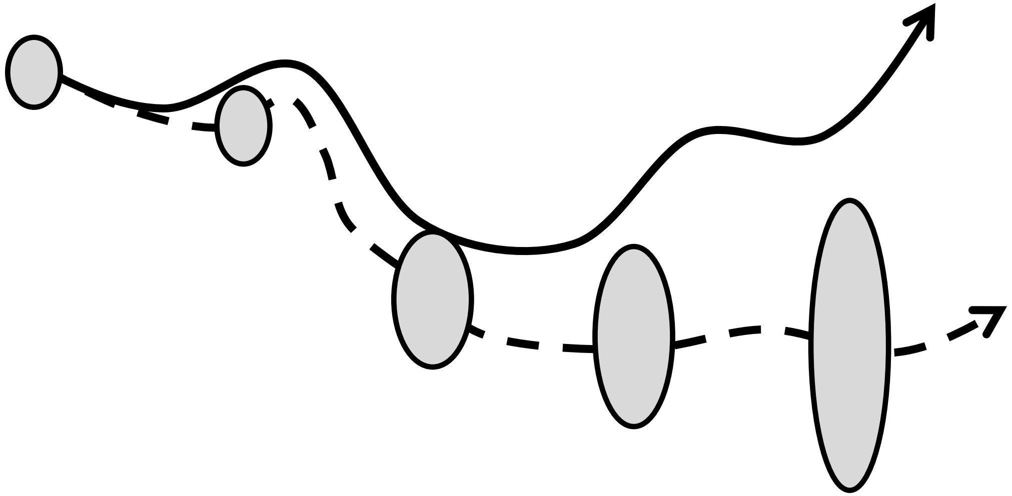
Idea: Estimate when OOD and account for it

→ Measure uncertainty!

<u>Maximum likelihood models</u>



<u>Uncertainty-aware models</u>



Being aware of uncertainty allows us to account for the effects of model bias!

Model Based RL v2 – Uncertainty Aware Models

<u>Alleatoric Uncertainty</u>

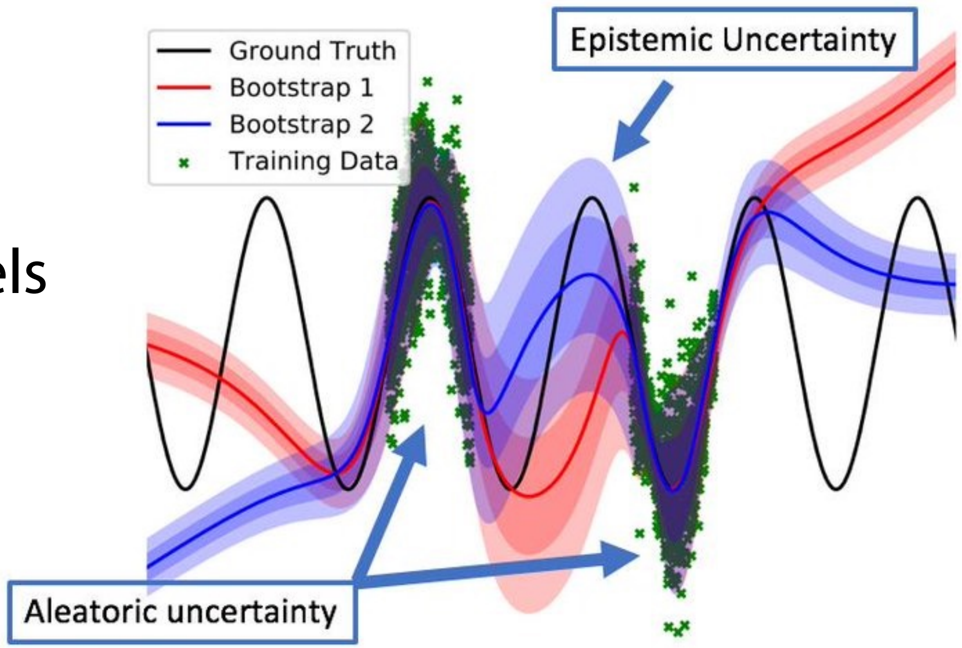
(environment stochasticity)

Easier, can use stochastic models

<u>Epistemic Uncertainty</u>

(Lack of data)

More challenging, need to compute posterior



Let's largely focus on epistemic uncertainty

Lecture outline

Model based RL v2 → uncertainty based models

↓

Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

↓

Control as Inference - Formulation

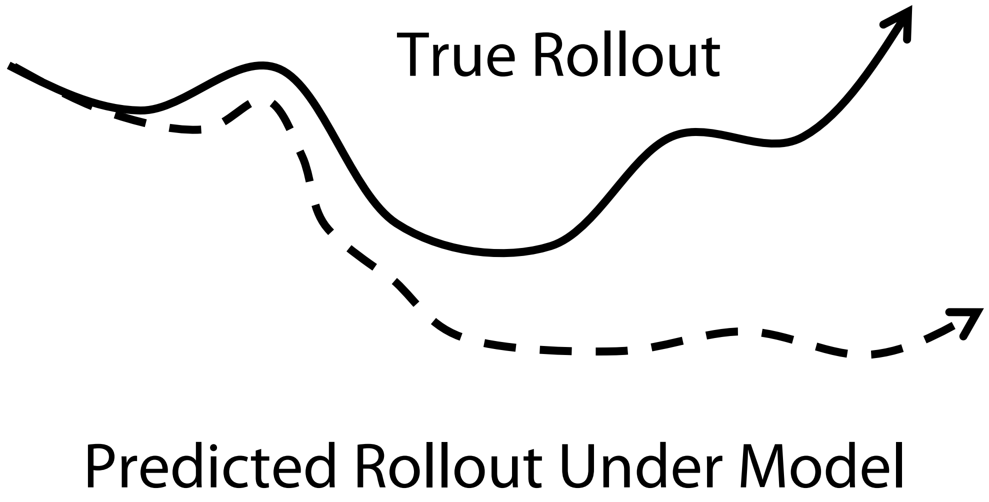
↓

Variational Inference

What might be the issue?

Rollouts under learned model != Rollouts under true model

→ Model bias/compounding error



Why does this happen? → lack of data

1. Errors in state go to OOD next states
2. Deviations in actions go to OOD next states

↓

Model is bad on OOD states!

Most trained deep models can only roll out for 5-10 steps maximum!

How might we deal with compounding error?

Idea 1: Change the training objective of the model to directly account for this!

Equation error – 1 step prediction error

$$\max_{\theta} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$

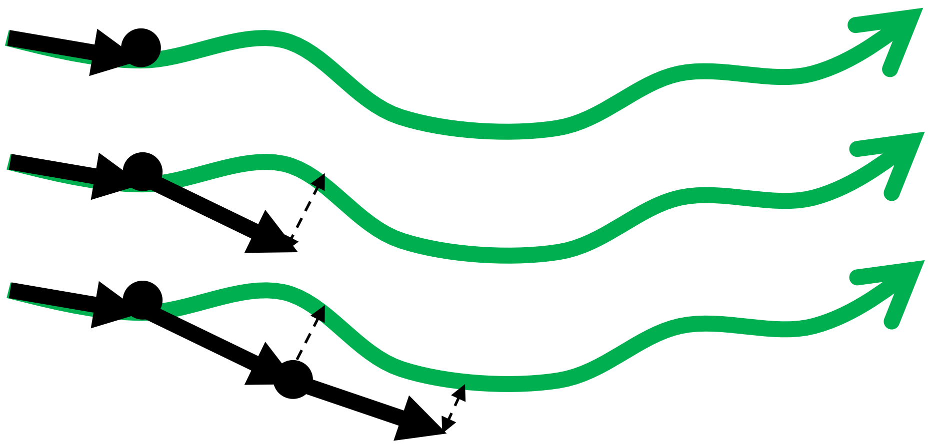
Simulation error – K step prediction error

$$\max_{\theta} \sum_t \log \hat{p}_\theta(s_{t+1}|\hat{s}_t, a_t)$$

$$\hat{s}_t \sim \hat{p}_\theta(.|\hat{s}_{t-1}, a_{t-1})$$

Model error under learned mode $\hat{p}_\theta$ rather under true model

Can be a challenging non-convex optimization!

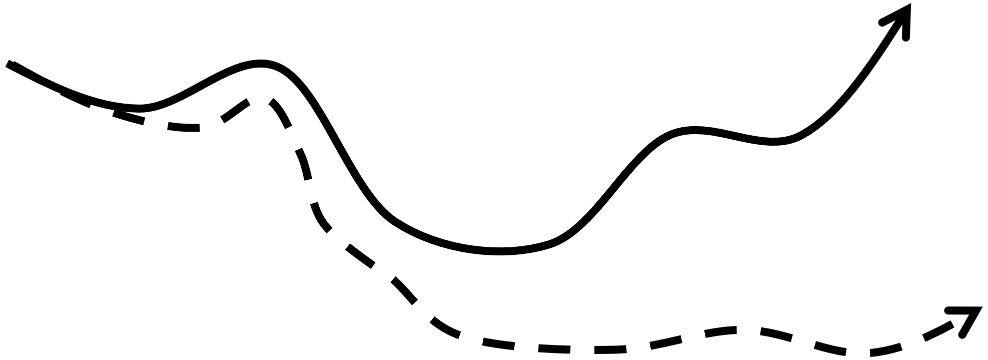


How might we deal with compounding error?

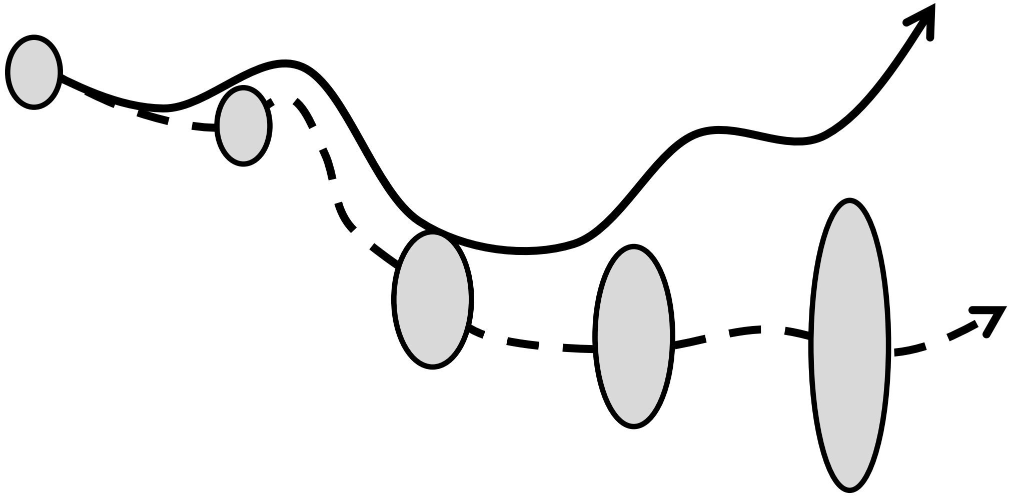
Idea 2: Estimate when OOD and account for it

→ Measure uncertainty!

Maximum likelihood models



Uncertainty-aware models



Being aware of uncertainty allows us to account for the effects of model bias!

What is uncertainty?

<u>Alleatoric Uncertainty</u>

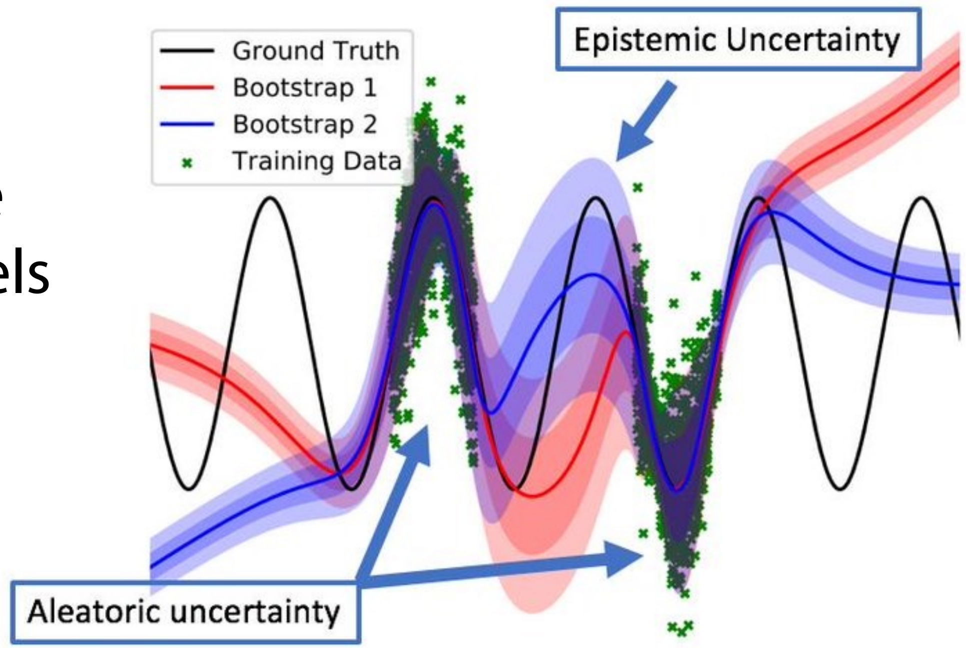
(environment stochasticity)

Easier, can use stochastic models

<u>Epistemic Uncertainty</u>

(Lack of data)

More challenging, need to compute posterior



Let's largely focus on epistemic uncertainty

How might we measure uncertainty?

$$p(\theta|\mathcal{D})$$

Difficult to estimate directly!

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta')p(\theta')d\theta'}$$

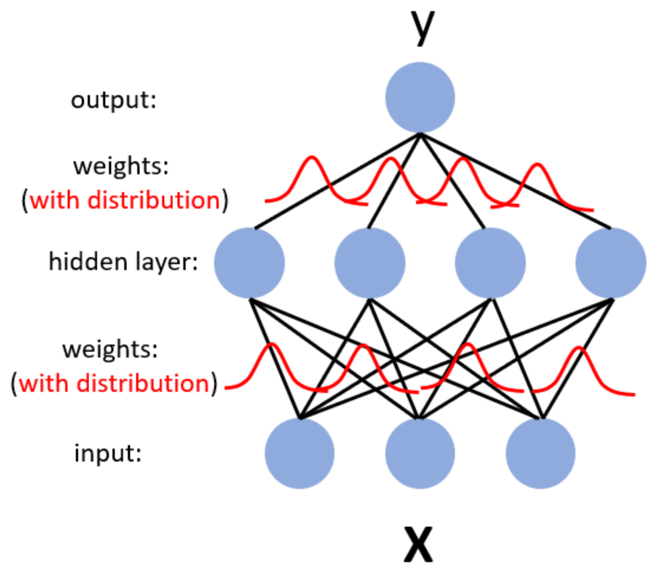
1. Bayesian neural networks
2. Ensemble methods
3. …

Directly model posterior distribution

Use variational inference to avoid computing partition function

$$\min_{q(\theta|\mathcal{D})} D_{KL}(q(\theta|\mathcal{D}) \,||\, p(\theta|\mathcal{D}))$$

Challenge: can be difficult to express rich distributions

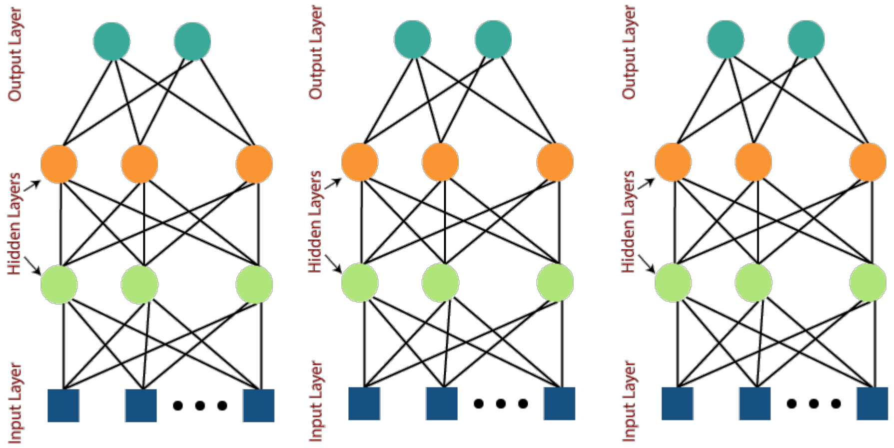


How might we measure uncertainty?

$p(\theta|\mathcal{D})$ Difficult to estimate directly!

Learn an ensemble of models

1. Bayesian neural networks
2. Ensemble methods →
3. …



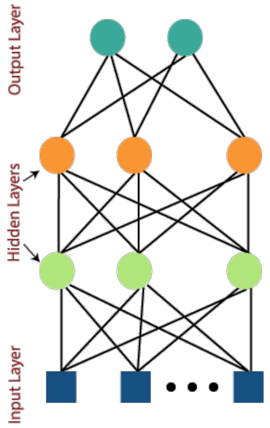
Low data regime → high ensemble variance

Approximate posterior

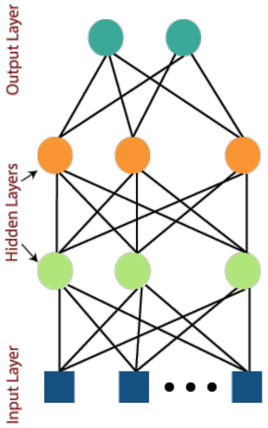
Easier and more expressive than BNNs!

Model Based RL – Learning Ensembles of Dynamics Models

Learn ensembles of dynamics models with MLE rather than a single model

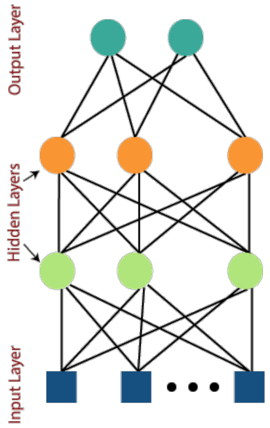


$$\max_\theta \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$



$$\max_\theta \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$

…

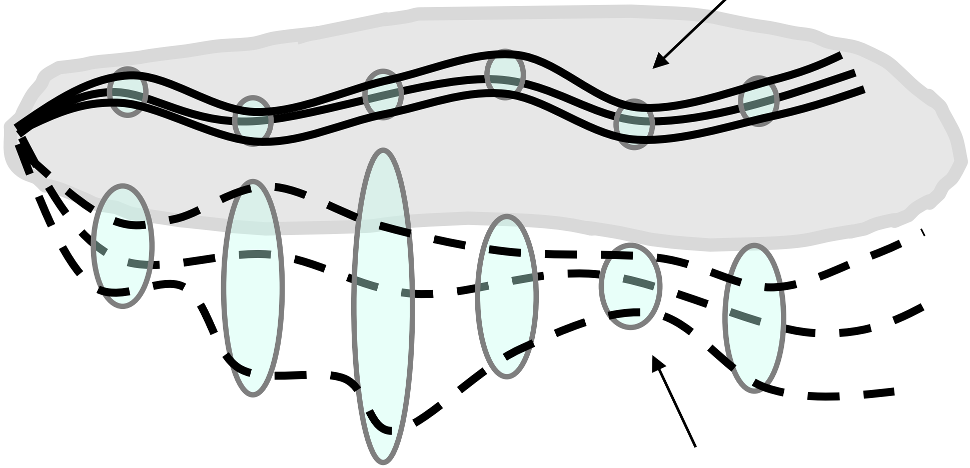


$$\max_\theta \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$

Learn ensembles by either subsampling the data or having different initializations

Model Based RL – Integrating Uncertainty into MBRL (v2)

Take expected value under the uncertain dynamics



Expected value over ensemble

$$\arg \max_{(a_0^j, a_1^j, \ldots, a_T^j)_{j=1}^N} \sum_{i=1}^{K} \sum_{t=0}^{T} r((\hat{s}_t^j)^i, a_t^j)$$

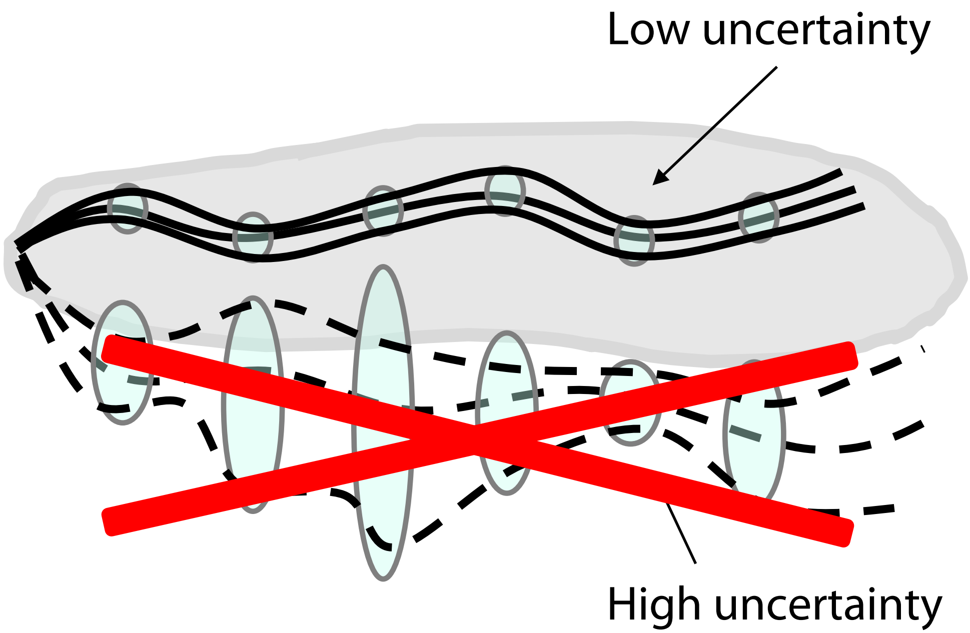
$$(\hat{s}_{t+1}^j)^i \sim \hat{p}_{\theta_i}(.|(\hat{s}_t^j)^i, a_t^j)$$

Can also swap which ensemble element is propagated at every step or just pick randomly amongst them

Avoids overly OOD settings since the expected reward is affected by uncertainty

Model Based RL – Integrating Uncertainty into MBRL (v2)

Take **pessimistic** value under the uncertain dynamics



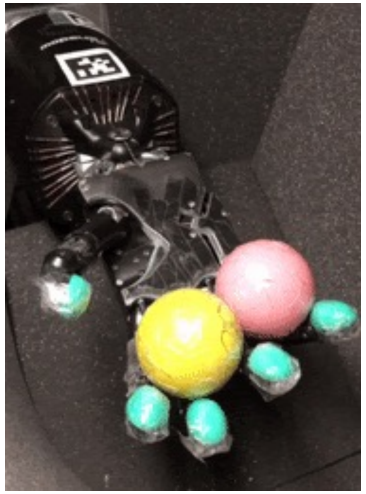
Penalize ensemble variance

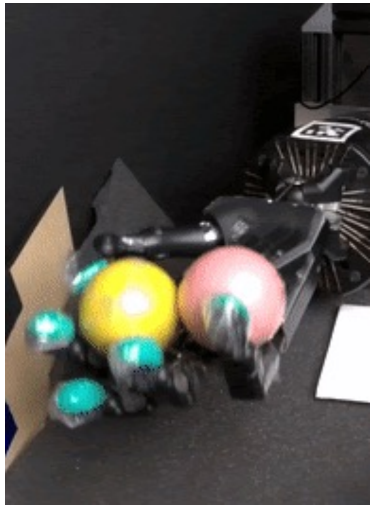
$$\arg \max_{(a_0^j, a_1^j, \ldots, a_T^j)_{j=1}^N} \sum_{i=1}^{K} \sum_{t=0}^{T} r((\hat{s}_t^j)^i, a_t^j) - \lambda \mathrm{Var}((\hat{s}_t^j)^i)$$

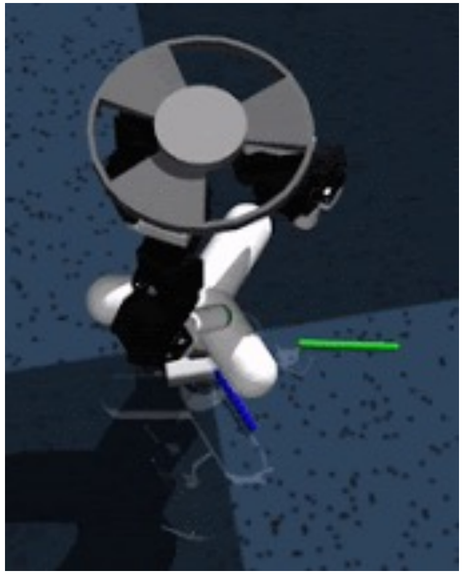
$$(\hat{s}_{t+1}^j)^i \sim \hat{p}_{\theta_i}(.|(\hat{s}_t^j)^i, a_t^j)$$

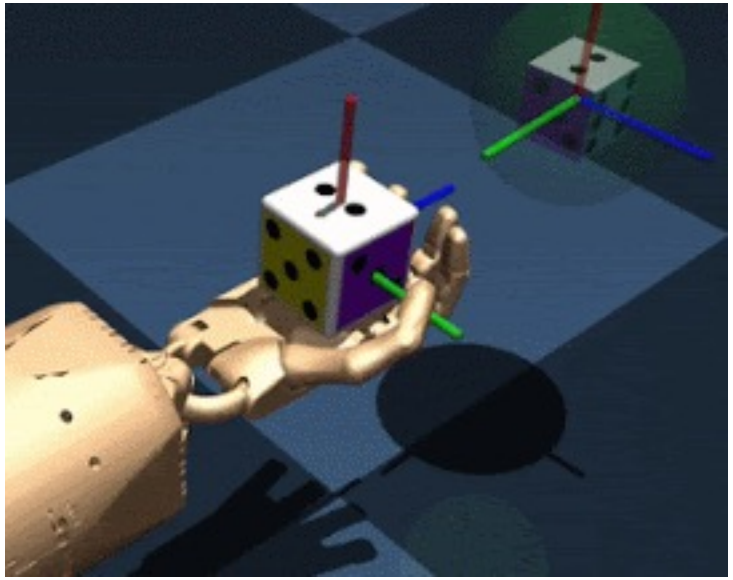
Avoids overly OOD settings since these states are explicitly penalized

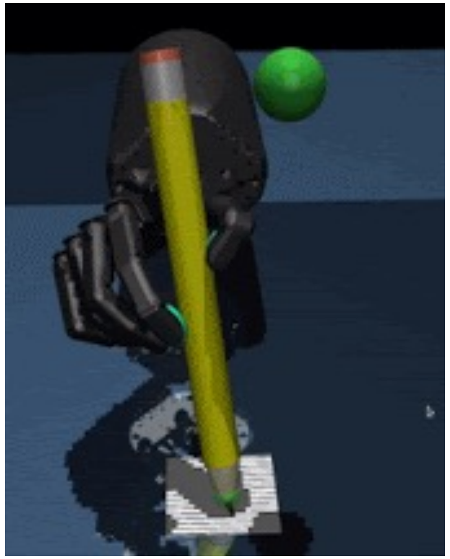
Does this work?











Lecture outline

Model based RL v2 → uncertainty based models

↓

Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

↓

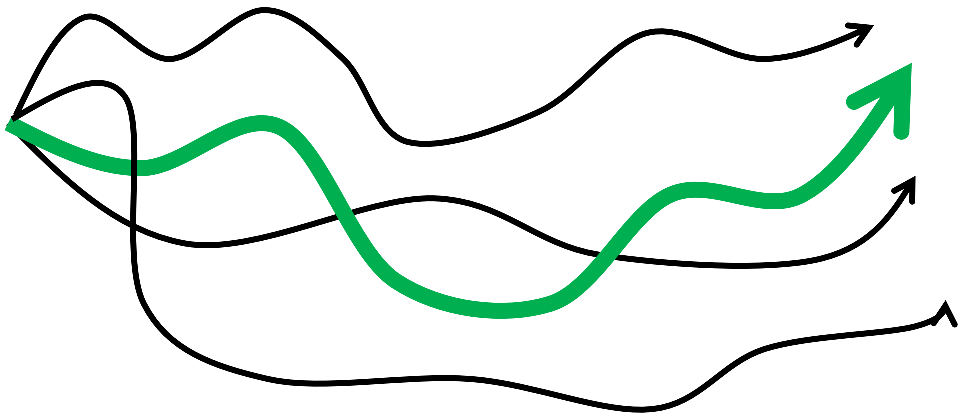
Control as Inference - Formulation

↓

Variational Inference

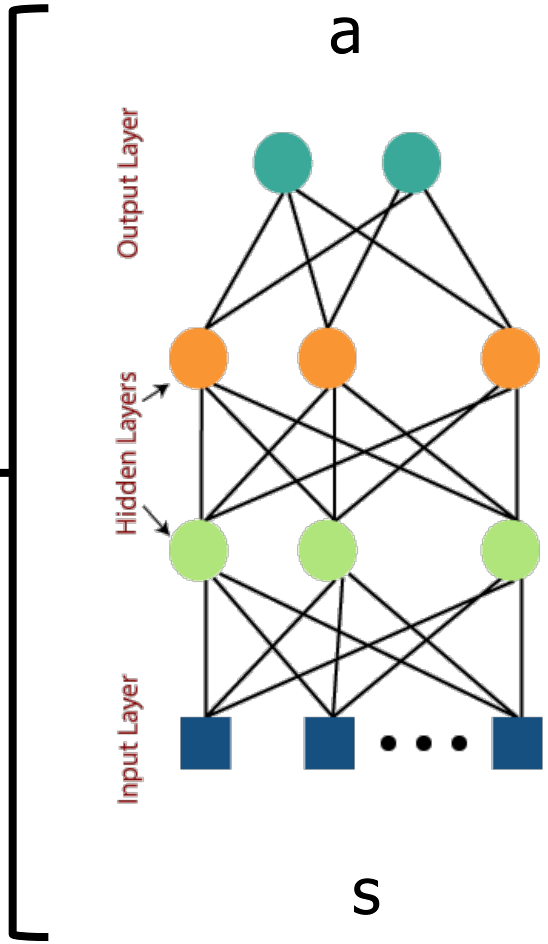
What might be the issue?

Huge number of samples needed to reduce variance



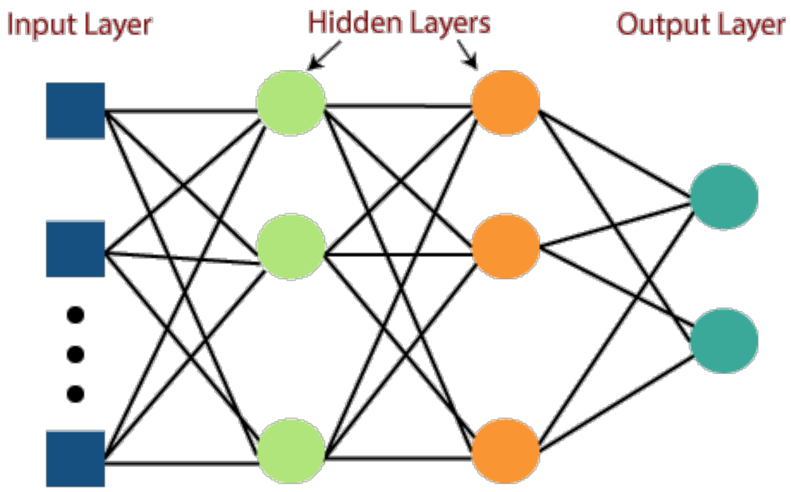
Extremely slow, hard to run in real time

Amortize planning into a policy

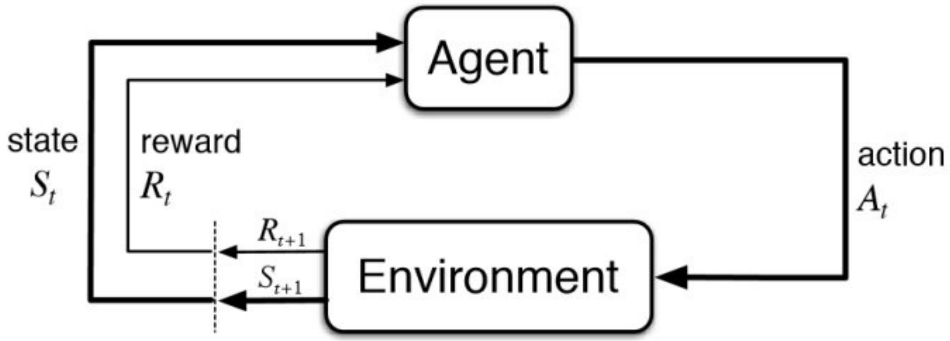


Speeding Up Model-Based Planning

$$\max_{\theta} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$

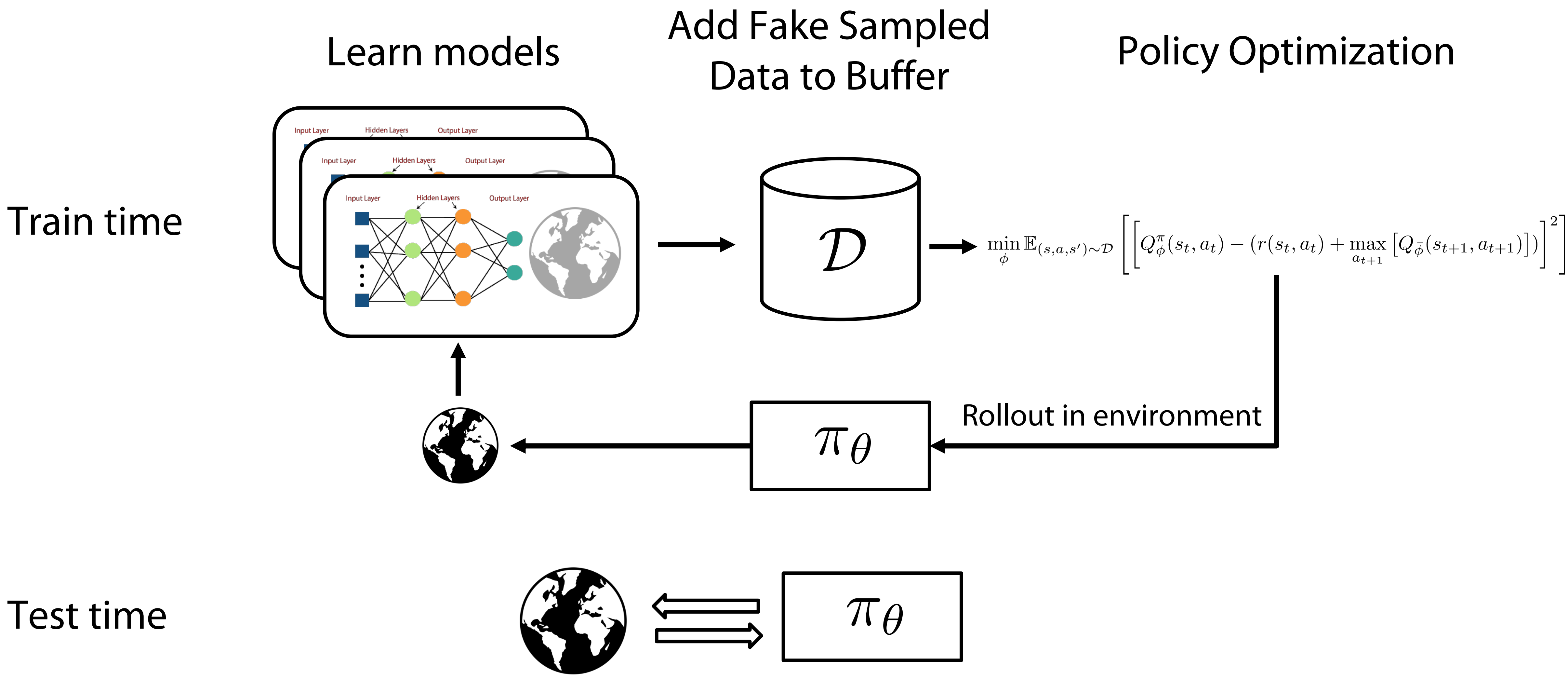


Use model(s) to generate data for policy optimization

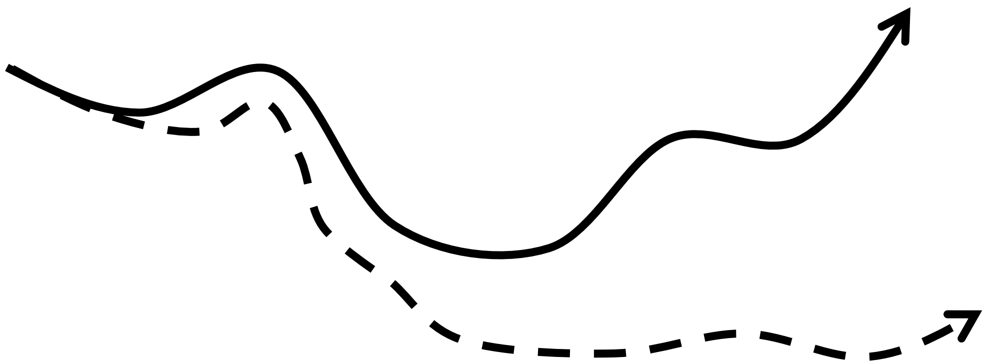


Can use PG or off-policy!

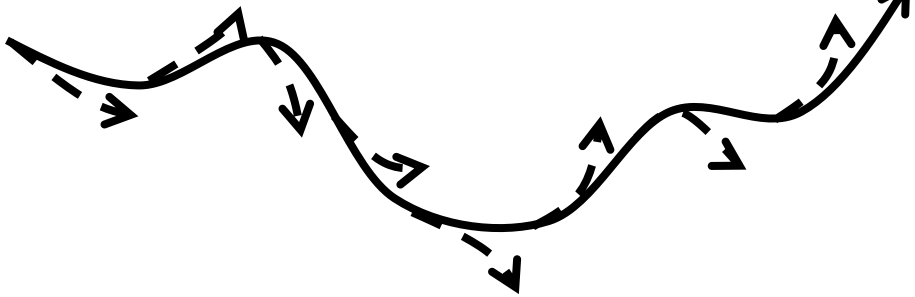
Generating Data for Policy Optimization



What matters in generating data from models?



Long horizon rollouts can deviate

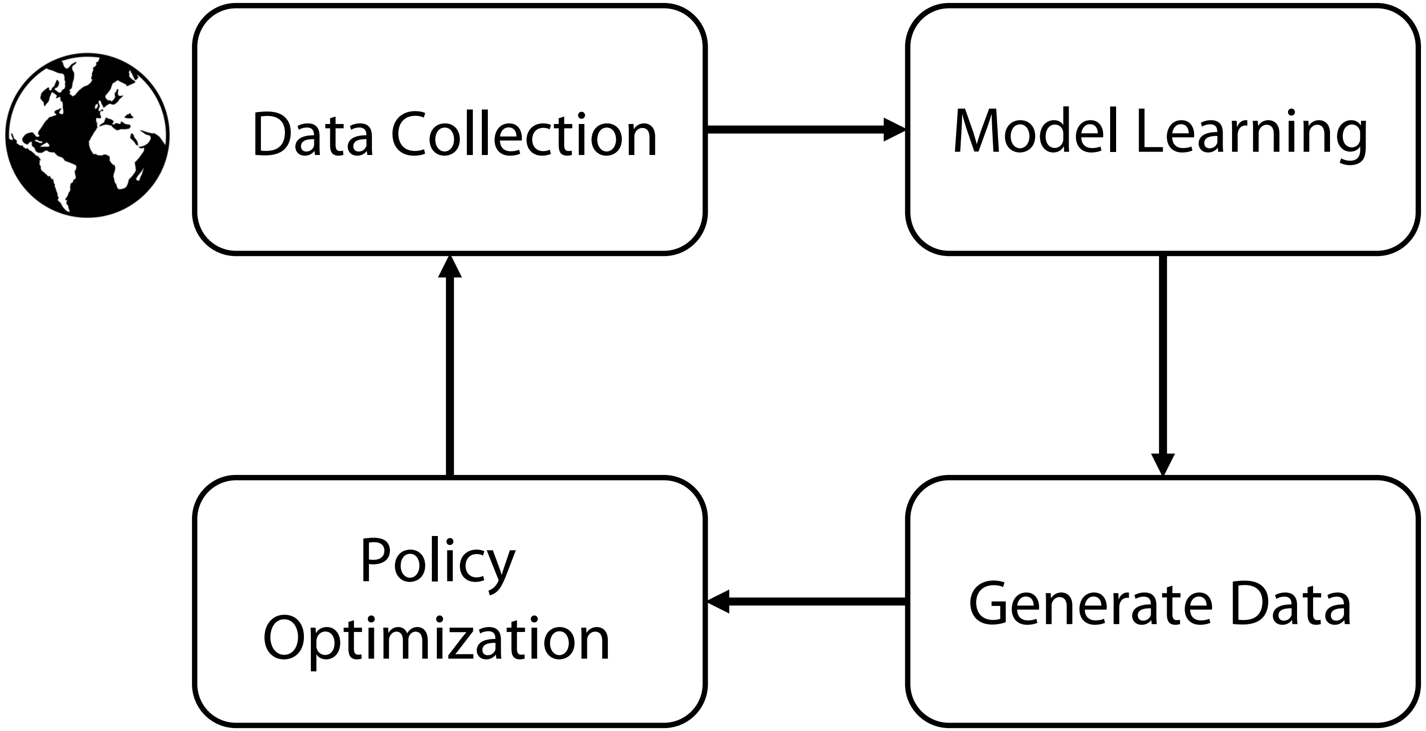


Short horizon rollouts deviate far less

Balance between off-policy coverage and compounding error

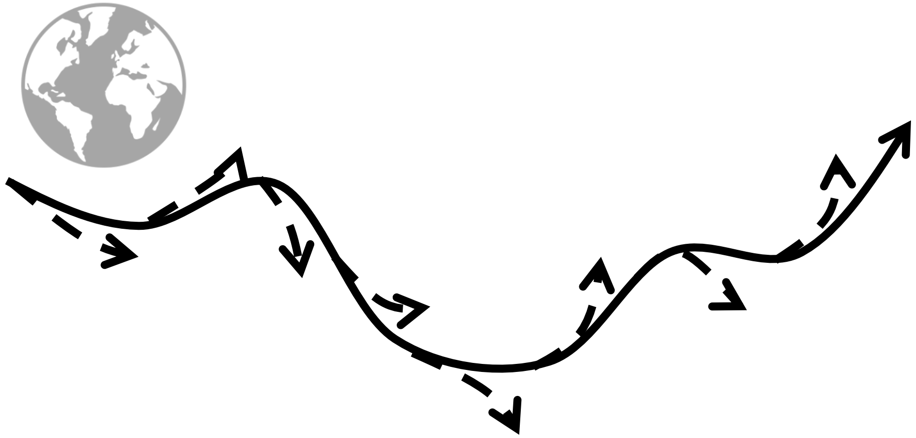
More in the readings!

Model Based RL – Using Models for Policy Optimization (v3)



Maximum likelihood supervised Learning

$$\max_{\theta} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\log \hat{p}_\theta(s'|s,a)\right]$$



$$\min_{\phi} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\left[Q^{\pi}_{\phi}(s_t, a_t) - \left(r(s_t, a_t) + \max_{a_{t+1}} \left[Q_{\bar{\phi}}(s_{t+1}, a_{t+1})\right]\right)\right]^2 \right]$$

More expensive/harder at training time, faster at test time

Does this work?

Lecture outline

Model based RL v2 → uncertainty based models

↓

Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

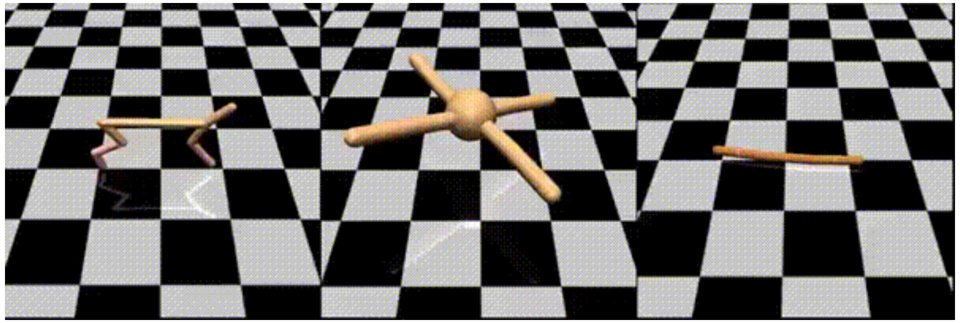
↓

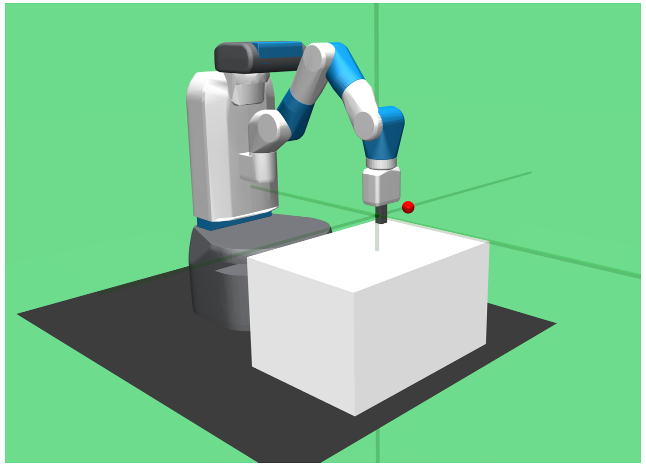
Control as Inference - Formulation

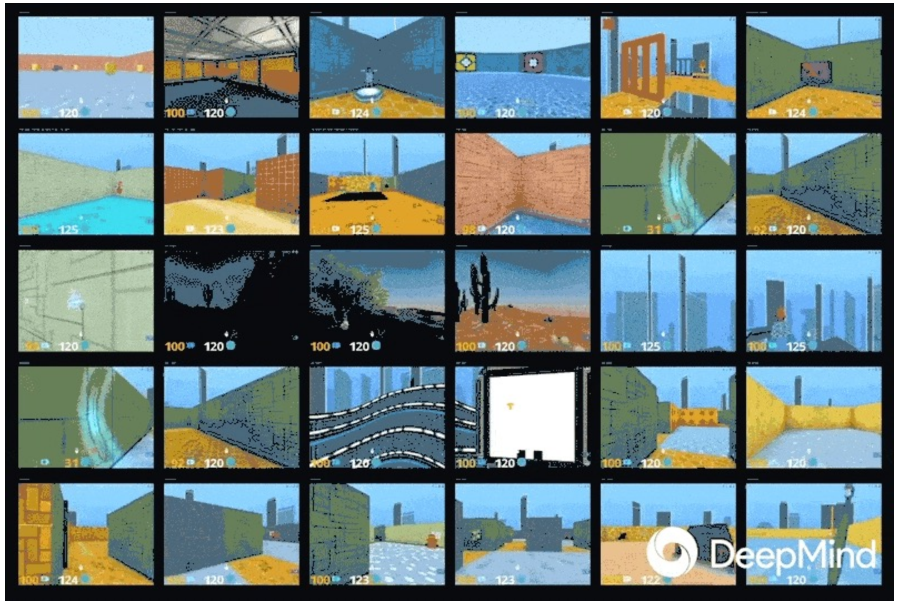
↓

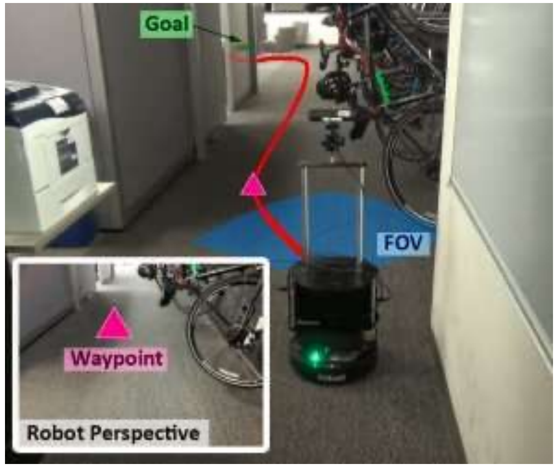
Variational Inference

What about images?







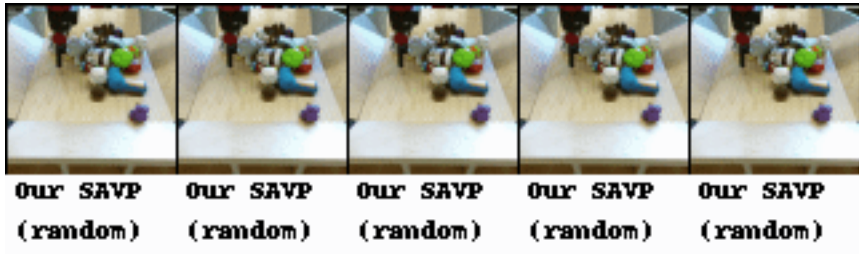


State based domains

Image based domains

Why is learning from images hard?

Generative modeling is videos, challenging to model multimodal correlated predictions

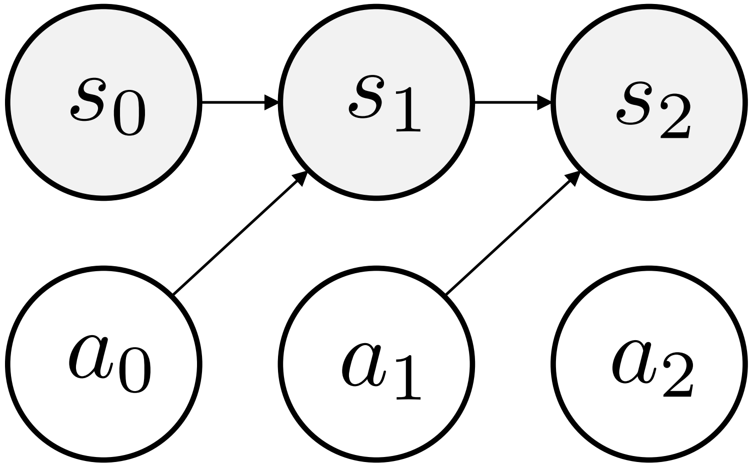


Partially observable!

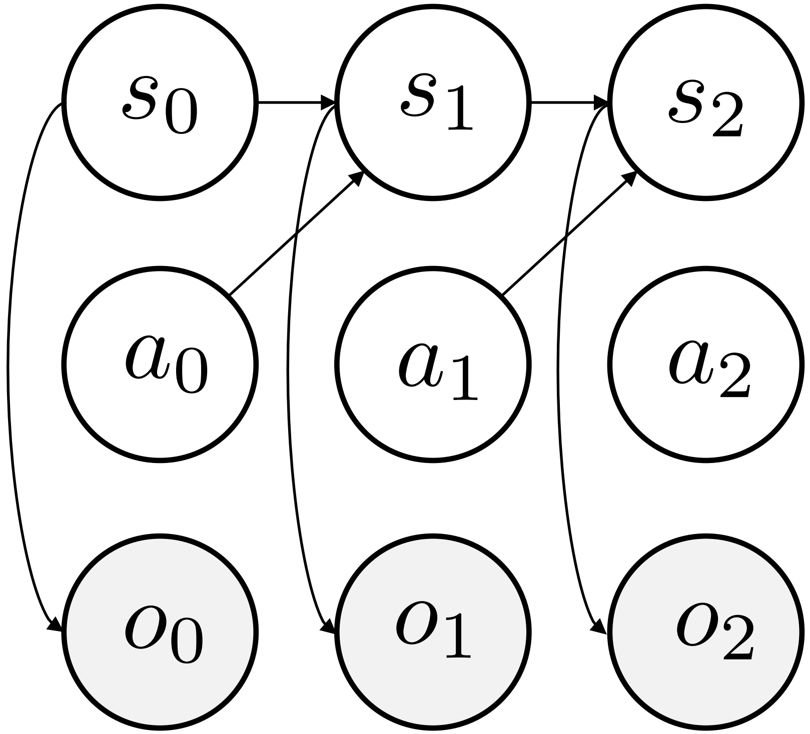
Long horizon predictions in video space can be challenging!

Model Based RL – Latent Space Models for Image Based RL (v4)

Fully observed – Markovian case



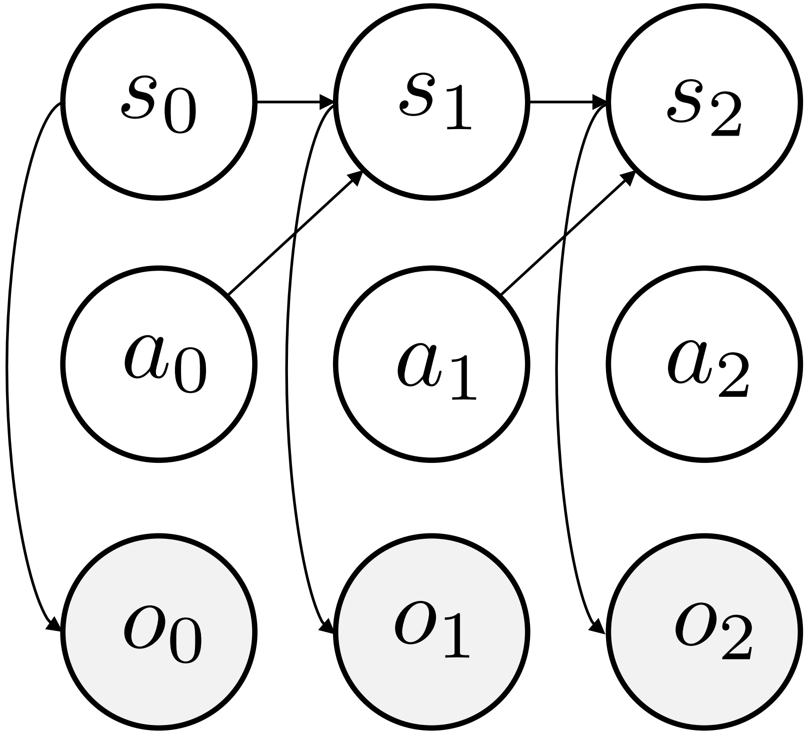
Partially observed – Non-Markovian case



If we can infer latent state and learn dynamics, then we can plan in a much smaller space

How do we infer latent state and learn dynamics in this space?

How do we **train** latent space models?



Learn latent encoder to infer latent state from observations $q_\phi(s_t|o_{1:t})$

Learn action conditioned latent transition model $p_\eta(s_{t+1}|s_t, a_t)$

$$\log p_\eta(q_\phi(s_{t+1}|o_{1:t+1})|q_\phi(s_t|o_{1:t}), a_t)$$

Learn latent decoder to reconstruct observations $p_\psi(o_t|s_t)$

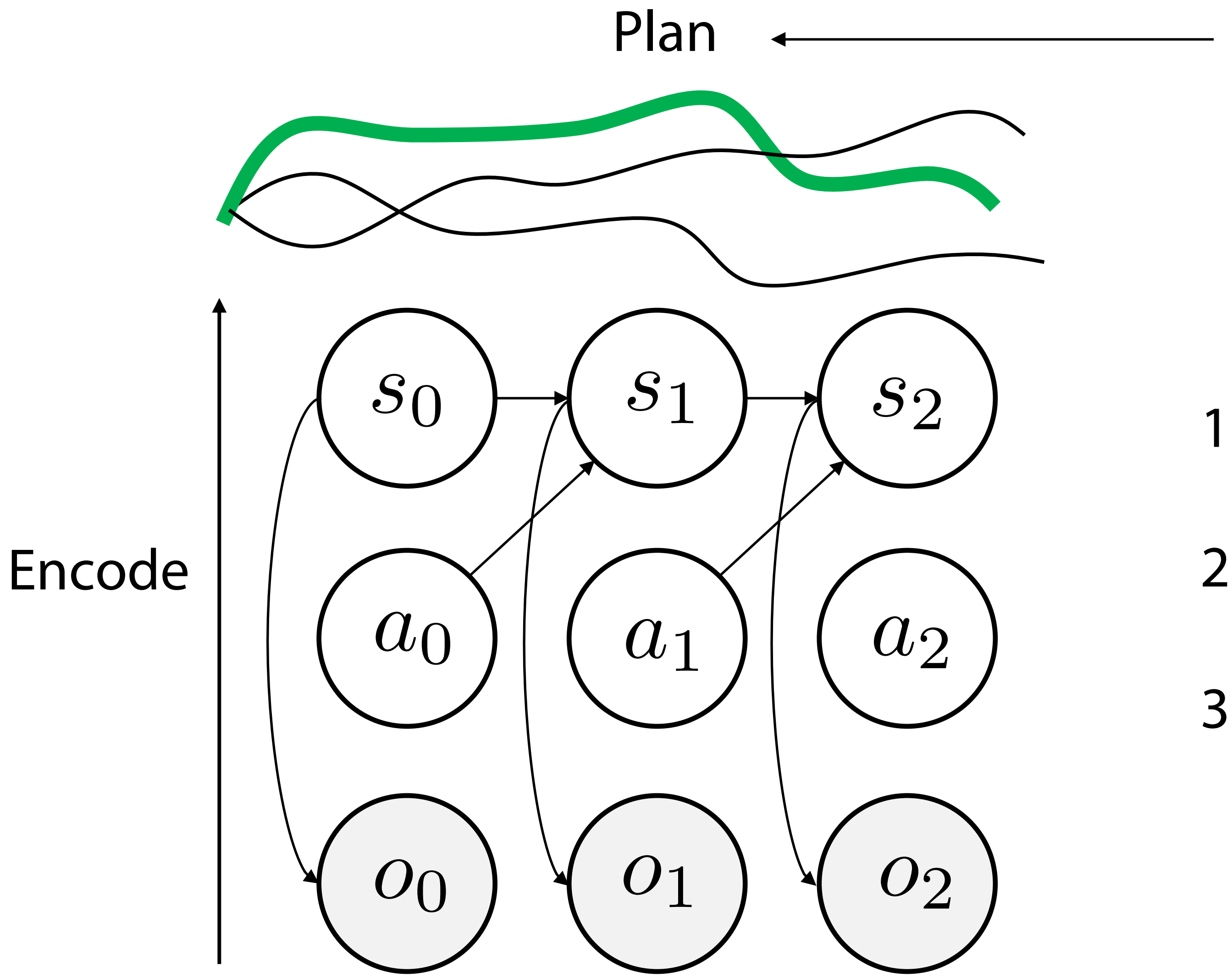
$$\log p_\psi(o_t|s_t)$$

Learn reward predictor from latent state $p_\zeta(r_t|s_t)$

$$\log p_\zeta(r_t|q_\phi(s_t|o_{1:t}))$$

Can derive the whole thing from first principles using variational inference!

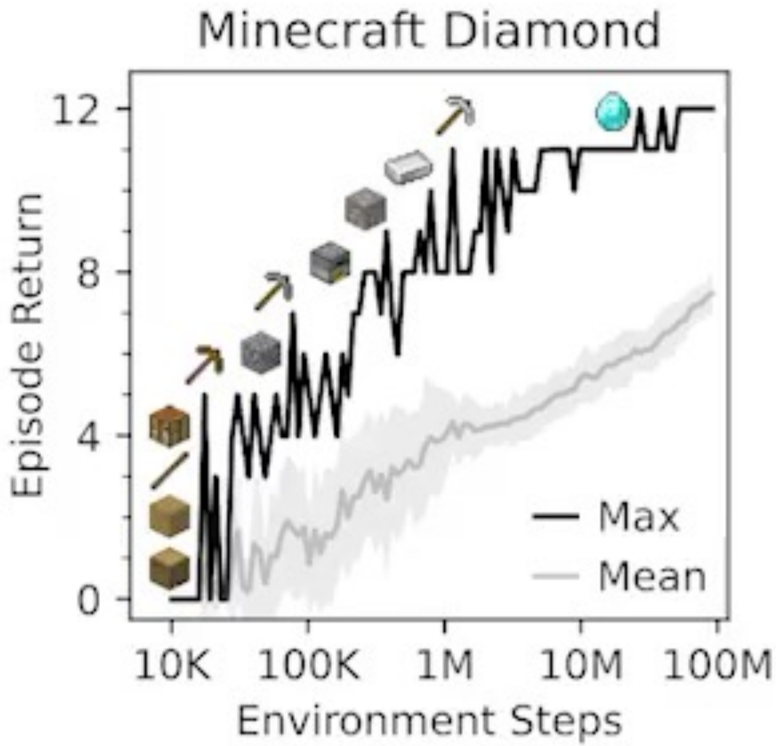
How do we **use** latent space models?



Apply any of the methods from this lecture, just in latent space!

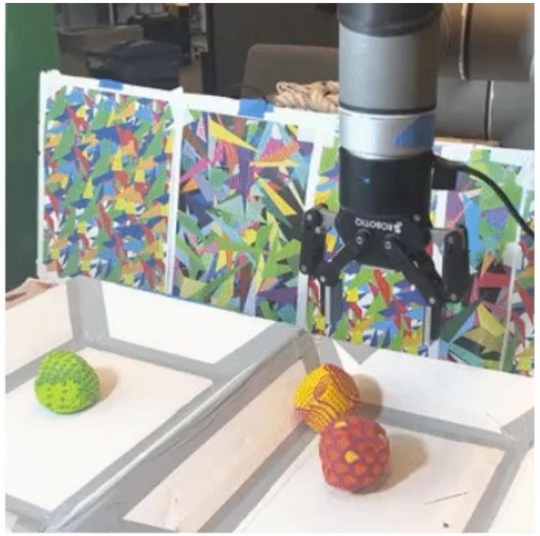
1. Avoids predicting image frames at planning time
2. Scales much better than image prediction
3. Allows for longer horizon predictions

Does this work?

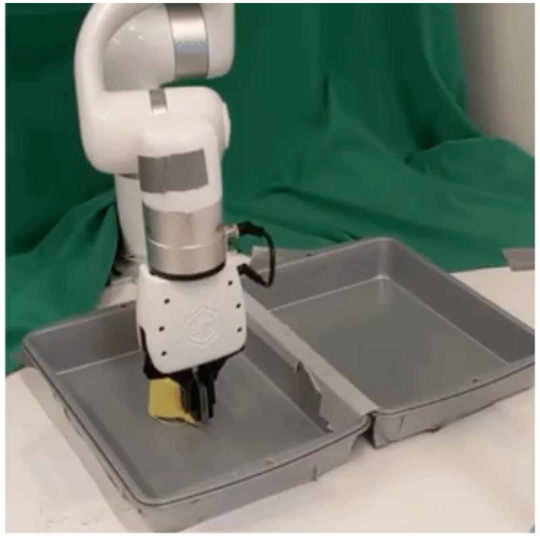


Does this work?

A1 Quadruped Walking



UR5 Multi-Object Visual Pick Place



XArm Visual Pick and Place

Sphero Ollie Visual Navigation

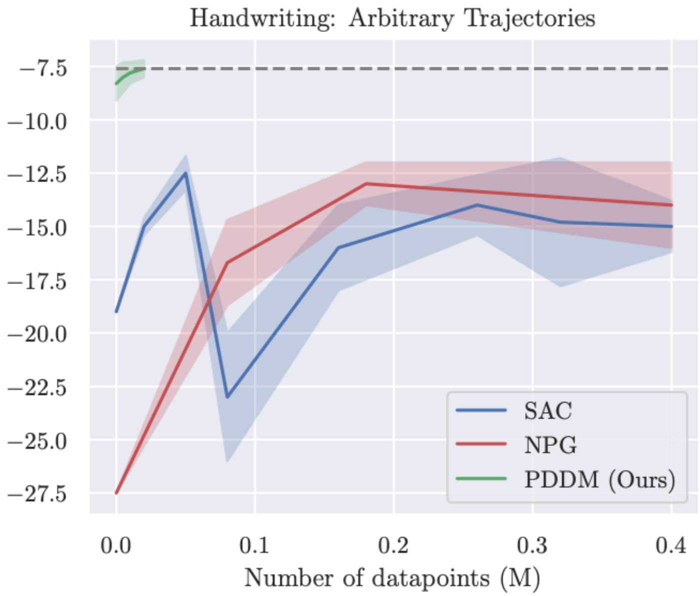
Training from images in < 1 hour!

Why should you care?

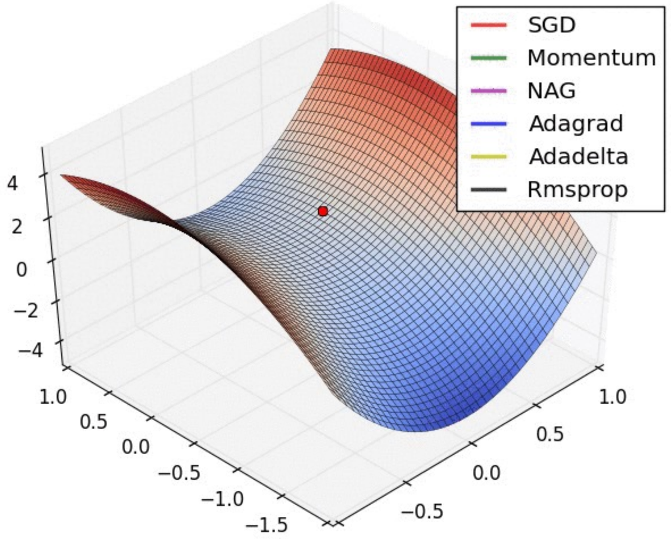
Model based RL **<u>may be</u>** a much more practical path to real world robotics

Transfer/Adaptive

Efficiency



Simplicity



Likely to be the most future proof one!

Are models really that different than Q-functions?

	Models	Q-functions
Similar	1. Off-policy 2. Models the future Very different than PG methods → on-policy, models current given future	
Different	1. 1-step modeling 2. Models states 3. Can evaluate arbitrary policies 4. Parametric storage of training data	1. Cumulative modeling 2. Models returns 3. Can evaluate only policy π 4. Non-parametric storage of data

Lecture outline

Model based RL v2 → uncertainty based models

↓

Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

↓

Control as Inference - Formulation

↓

Variational Inference

Ok, let's talk about "optimality"

Optimal control problems aim to find the "max" reward policy

$$\arg \max_{a_0^j, a_1^j, \ldots, a_T^j} \sum_{t=0}^{T} r(\hat{s}_t^j, a_t^j)$$

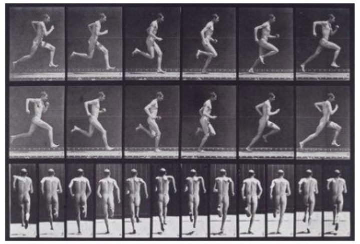
$$\hat{s}_{t+1}^j \sim \hat{p}_\theta(.|\hat{s}_t^j, a_t^j)$$

$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r(s_t, a_t) \right]$$

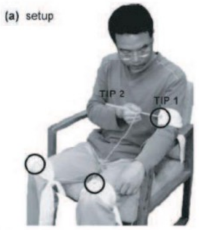
No notion of smooth suboptimality

People are not perfectly rational, "noisily" rational

Video of someone doing something irrational


Muybridge (c. 1870)


Mombaur et al. '09

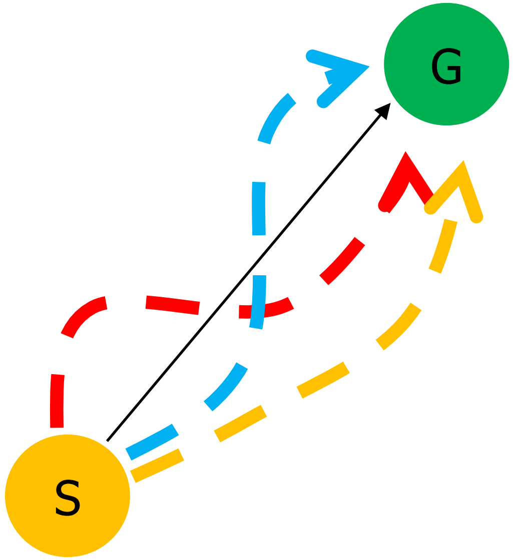


Li & Todorov '06

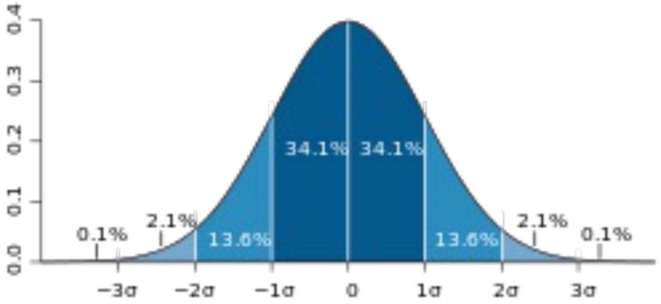

Ziebart '08

Can we think about "soft optimality"?

So how can we properly model suboptimality?



Some mistakes are more important than others



Let's use probability as a tool to represent "soft optimality"

- Going from deterministic to stochastic policies
- Better reward trajectories are "higher" likelihood
- Probabilistic measure of optimality, rather than an optimization one

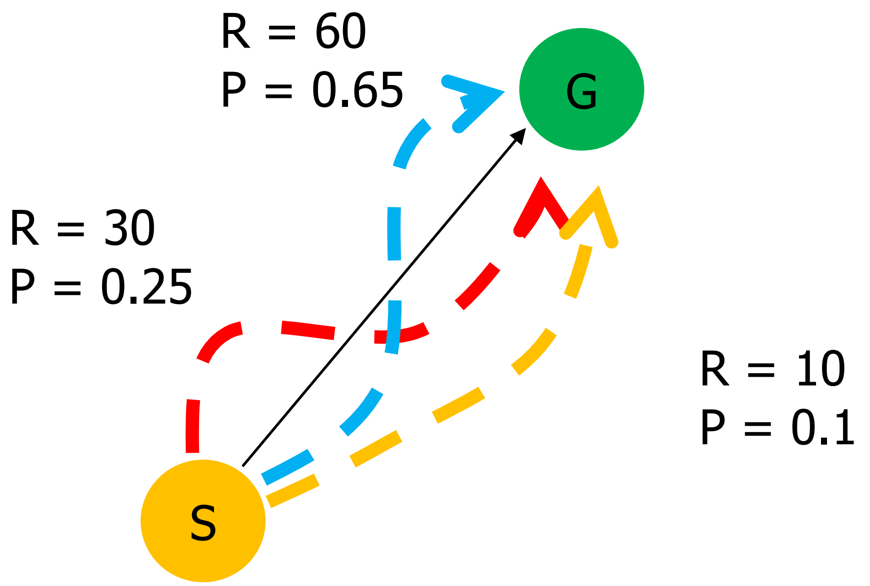
Let's use probabilistic inference as a tool

$$\arg\max_{a_0^j, a_1^j, \ldots, a_T^j} \sum_{t=0}^{T} r(\hat{s}_t^j, a_t^j)$$

$$\hat{s}_{t+1}^j \sim \hat{p}_\theta(.|\hat{s}_t^j, a_t^j)$$

$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{T} r(s_t, a_t)\right]$$

Rather than taking max wrt returns, sample proportional to returns

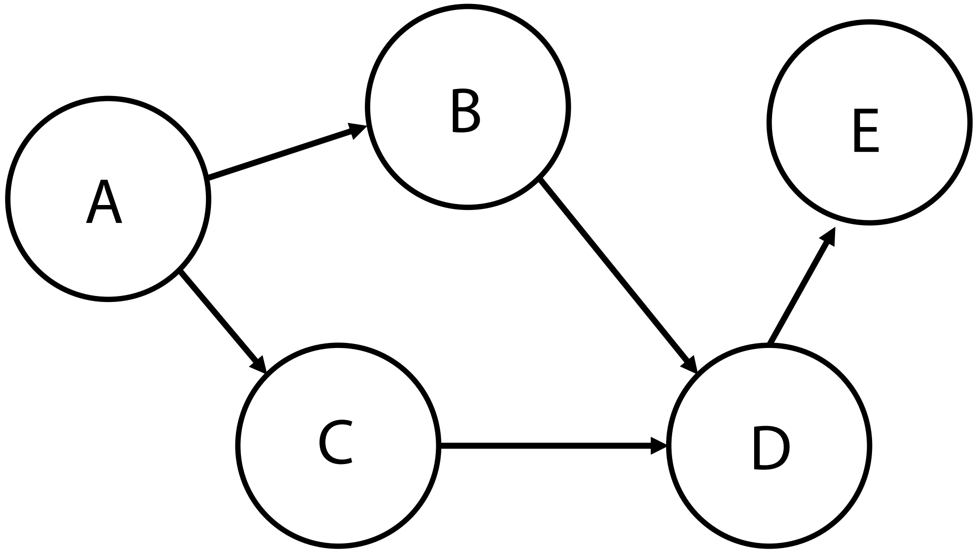


Soft RL/IRL

Sampling ⇄ Optimization

Langevin Dynamics

Probabilistic Graphical Models



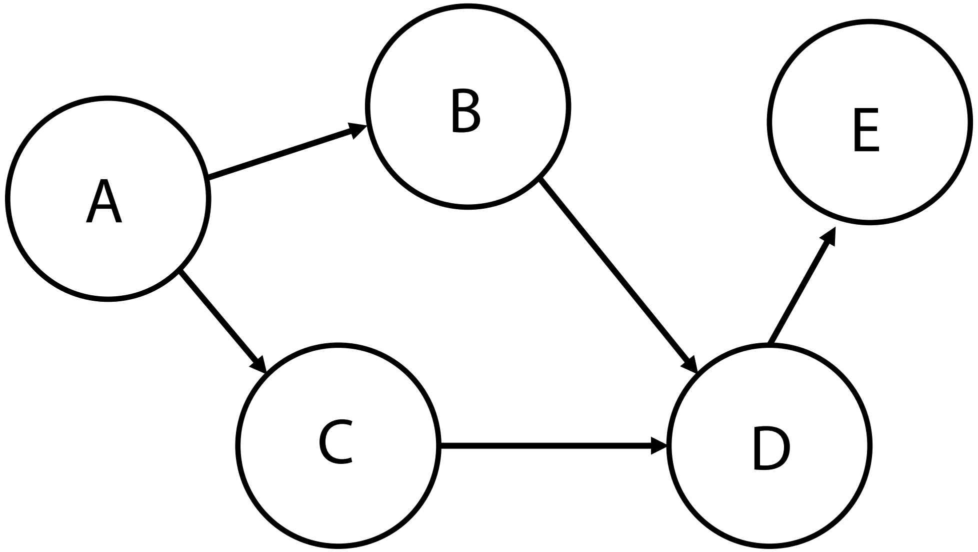
Convenient way to encode joint probability distribution

Encodes probabilities and conditional independences

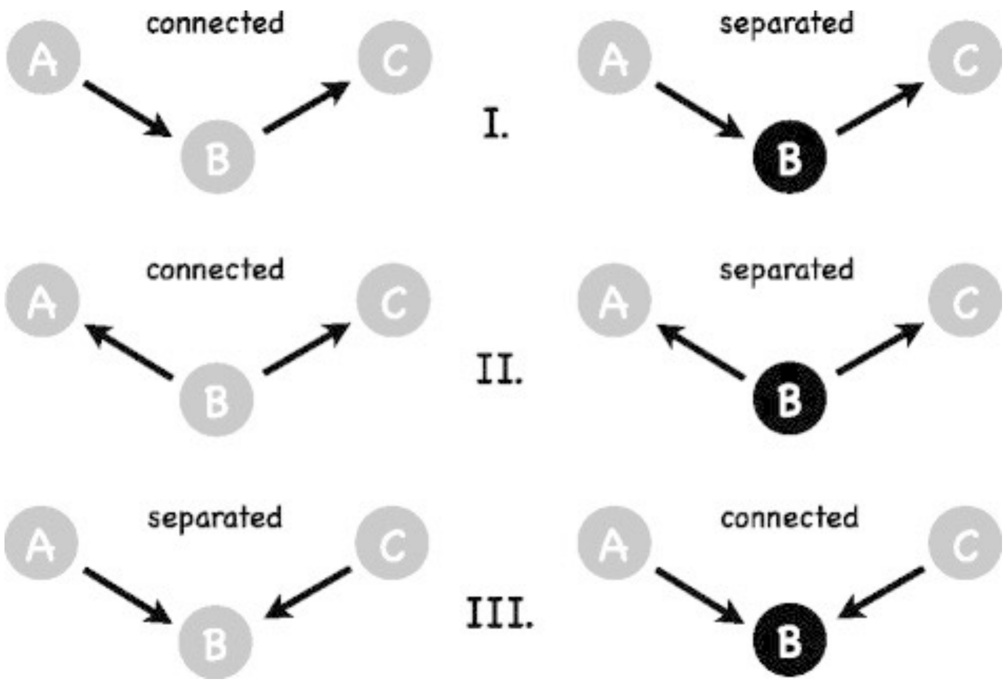
$$P(A, B, \ldots,) = \Pi_X P(X|\text{Parents}(X))$$

$$P(A, B, \ldots,) = P(A)P(B|A)P(C|A)P(D|B, C)P(E|D)$$

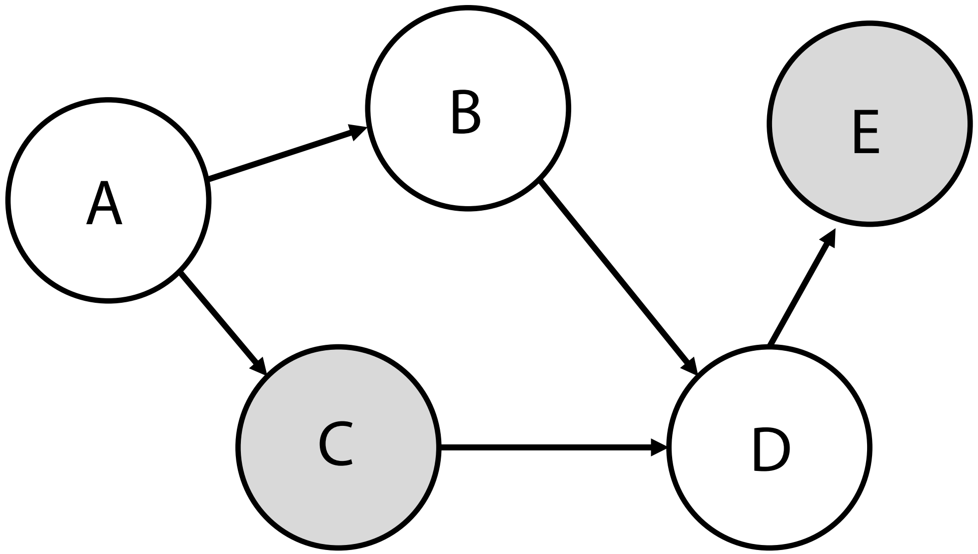
Probabilistic Graphical Models



Establish conditional independencies via d-separation (just read the graph)



Probabilistic Graphical Models



So what can you do with a probabilistic graphical model?

P(B|C, E)

P(A, B|C, E)

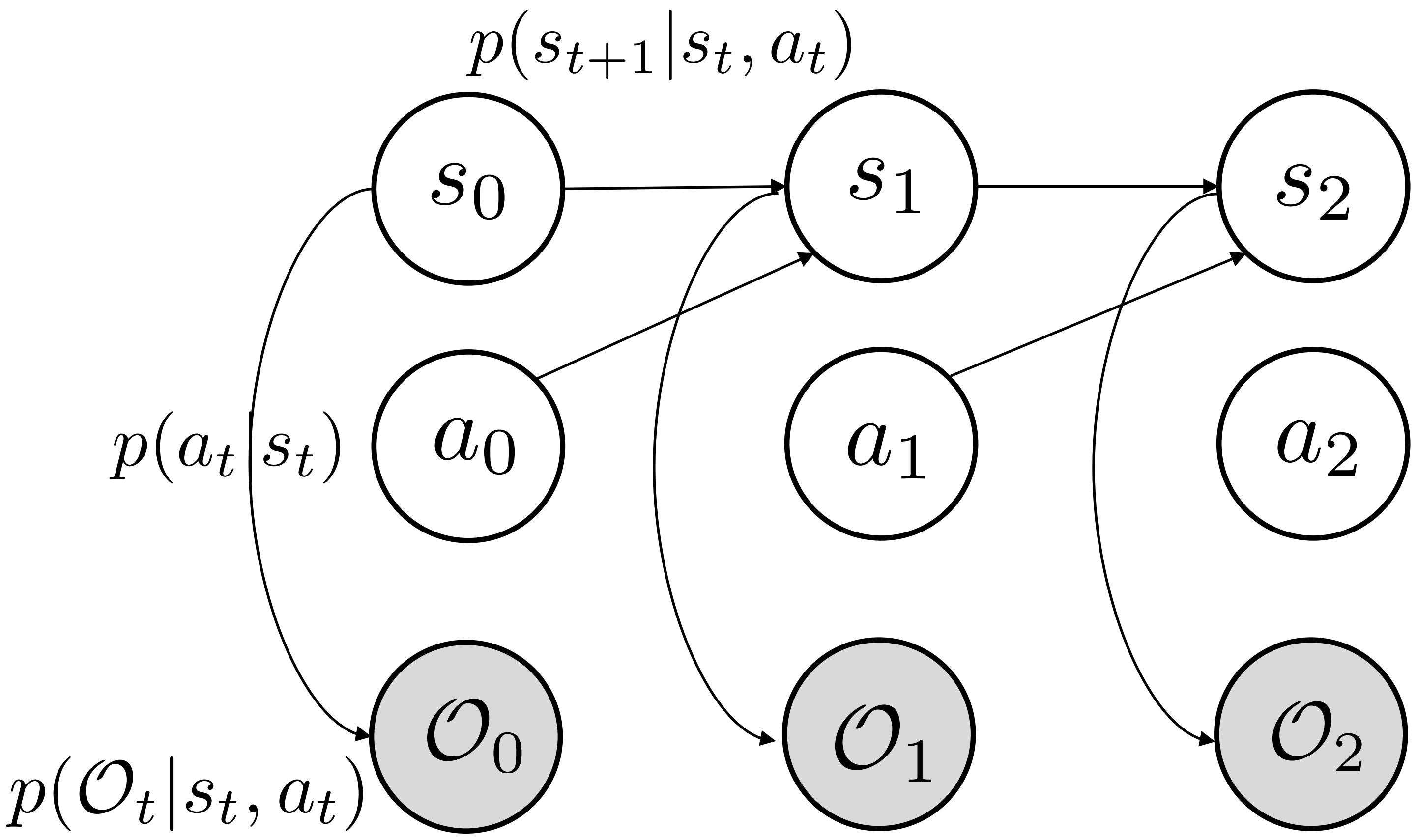
Answer posterior inference queries

What does this have to do with RL?

Isn't RL about maximizing expected reward?

Need to "eliminate" variables and use Bayes rule
→ Easy in discrete space, challenging in continuous

Using Probabilistic Graphical Models for Decision Making



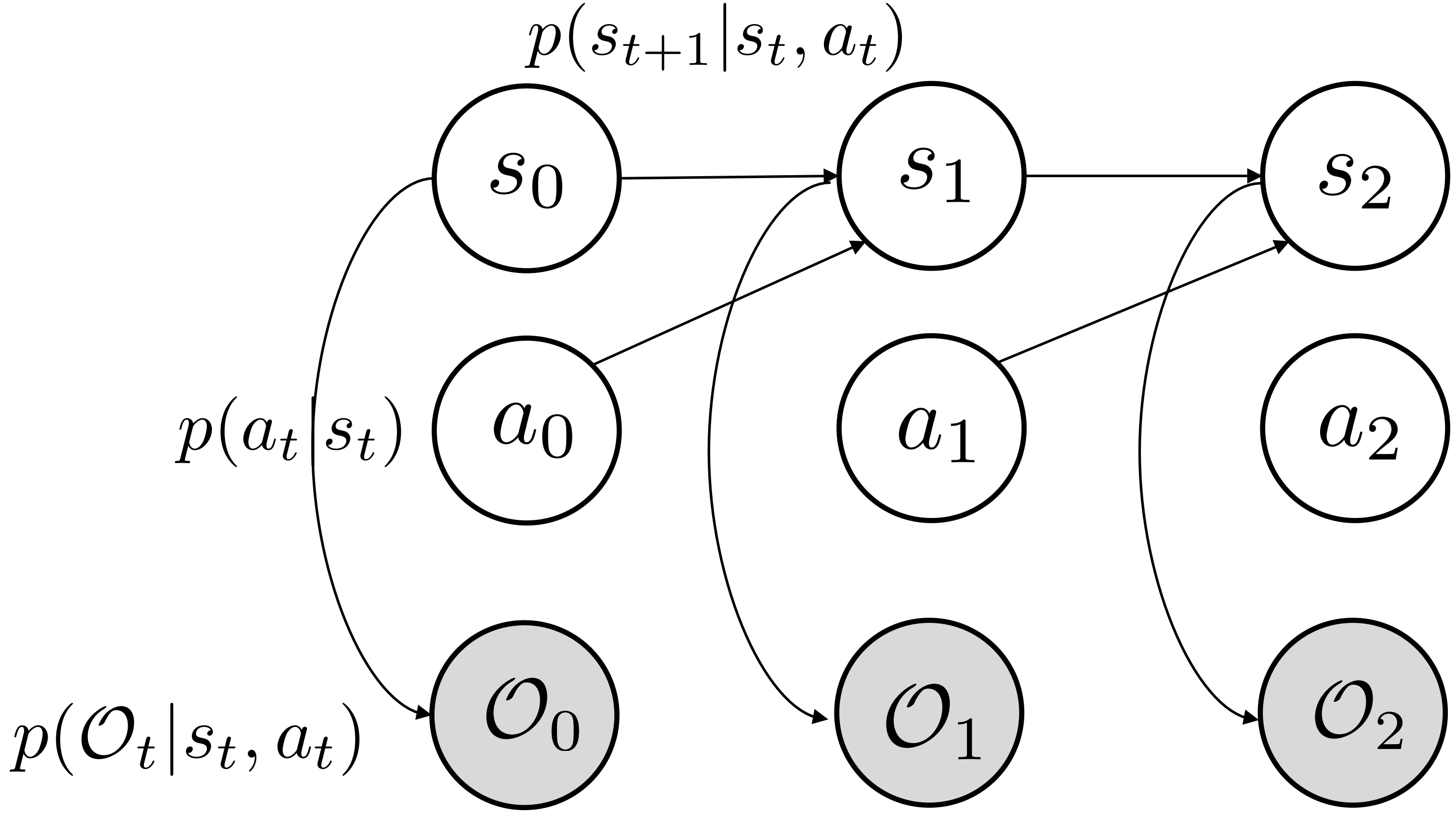
$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

Rewards must be negative (subtract max reward WLOG)

Introduce binary "optimality" variables – optimal if O=1, suboptimal if O=0

Agents are observed to be **optimal**

Ok so how can we cast decision making as a PGM?



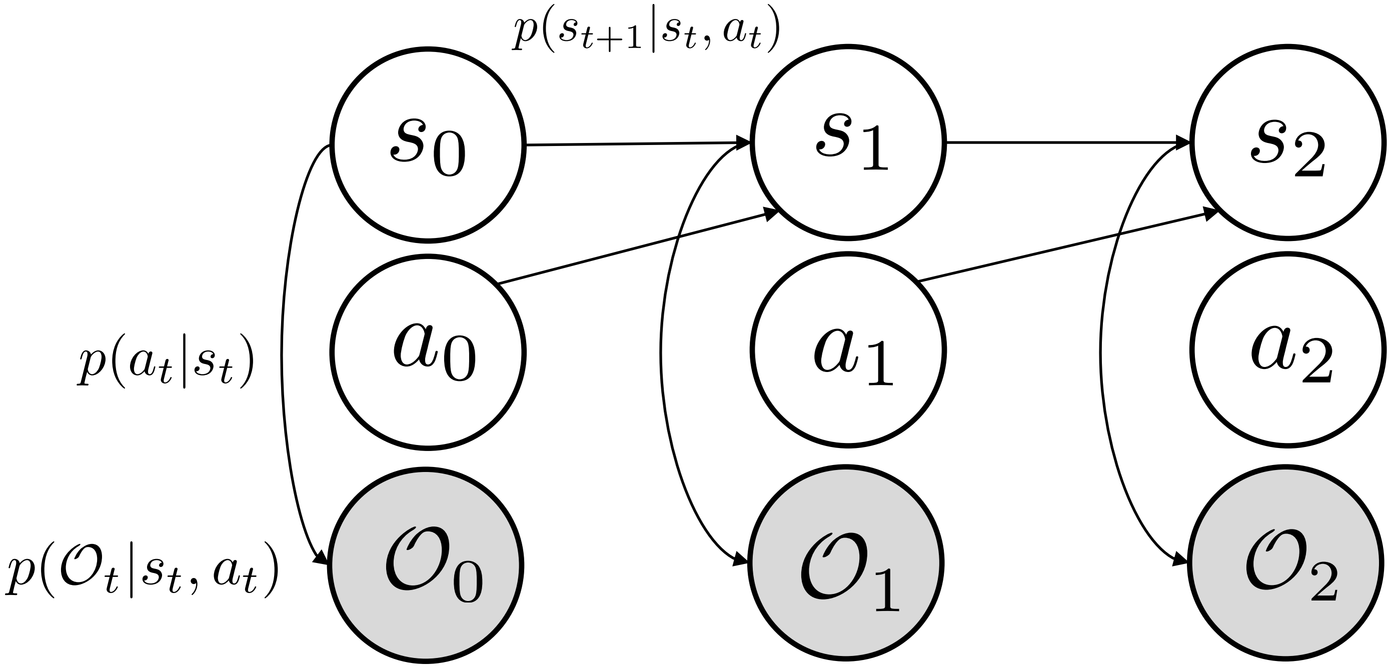
$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau)p(\mathcal{O}_{0:T}|\tau) = p(s_0)\prod_{t=0}^{T} p(s_{t+1}|s_t, a_t)p(a_t|s_t)p(\mathcal{O}_t|s_t, a_t)$$

$$= p(\tau)\exp(\sum_{t=0}^{T} r(s_t, a_t))$$

"Soft" optimality – higher return trajectories are higher likelihood

Ok big whoop, what do we do this?



$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau) \exp(\sum_{t=0}^{T} r(s_t, a_t))$$

Use case 1:

Derive soft RL algorithms

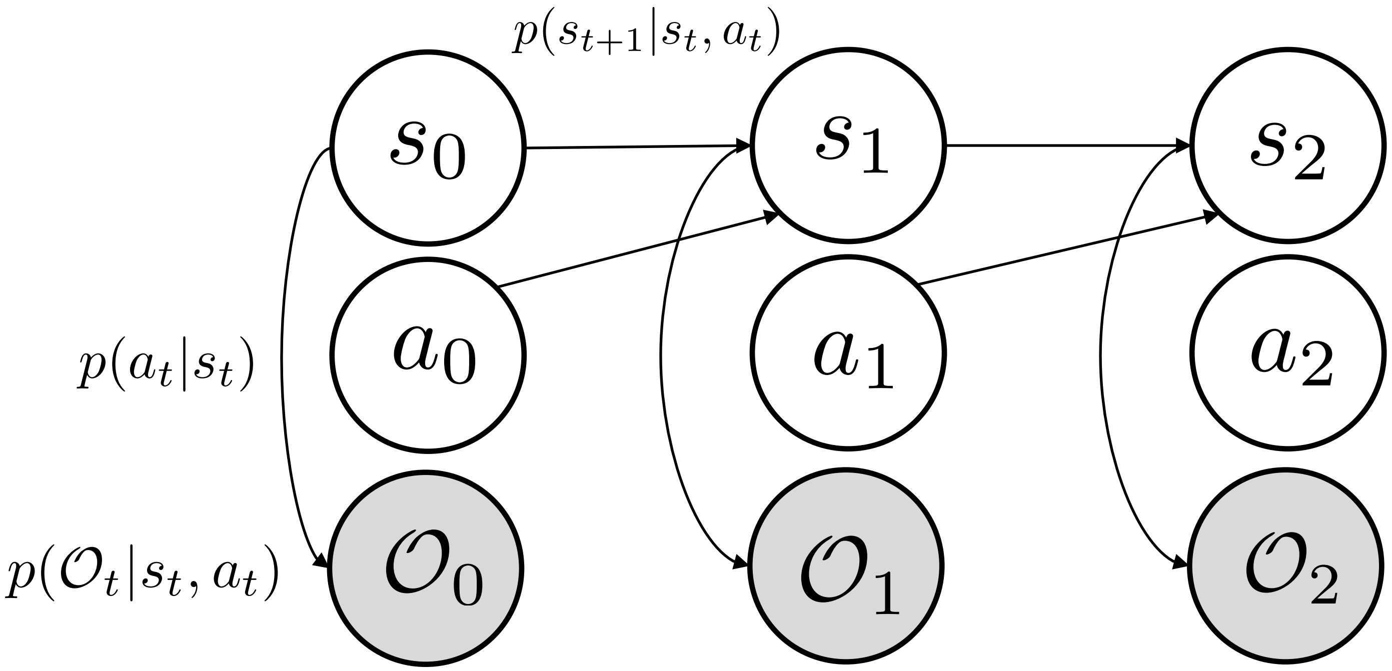
Use case 2:

Derive soft inverse RL algorithms

Use case 3:

Great algorithms for transfer

So what are we doing inference over?



$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau)\exp(\sum_{t=0}^{T} r(s_t, a_t))$$

Use case 1:

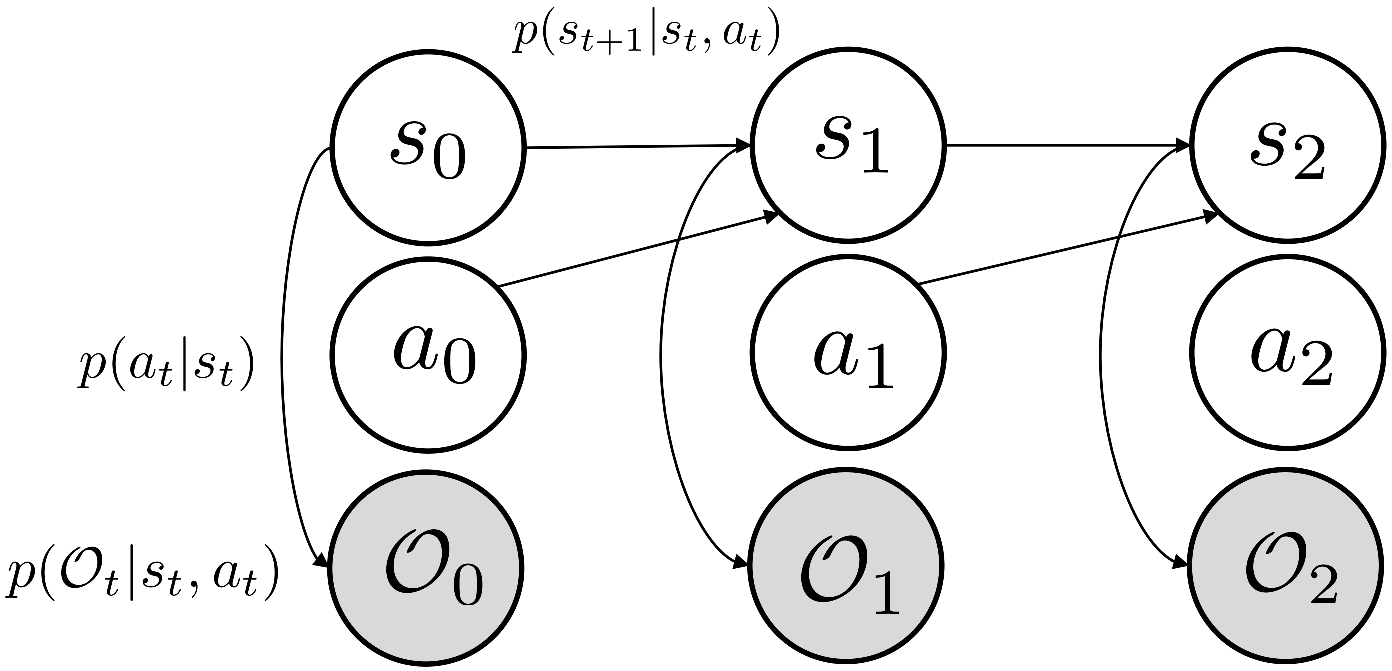
Derive soft RL algorithms

Insight: Computing optimal policy → posterior inference

$$p(a_t|s_t, \mathcal{O}_{t:T} = 1)$$

"Given that you are acting optimally, what is the likelihood of a particular action at a state"

So what are we doing inference over?



$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau) \exp(\sum_{t=0}^{T} r(s_t, a_t))$$

Use case 1:

Derive soft RL algorithms

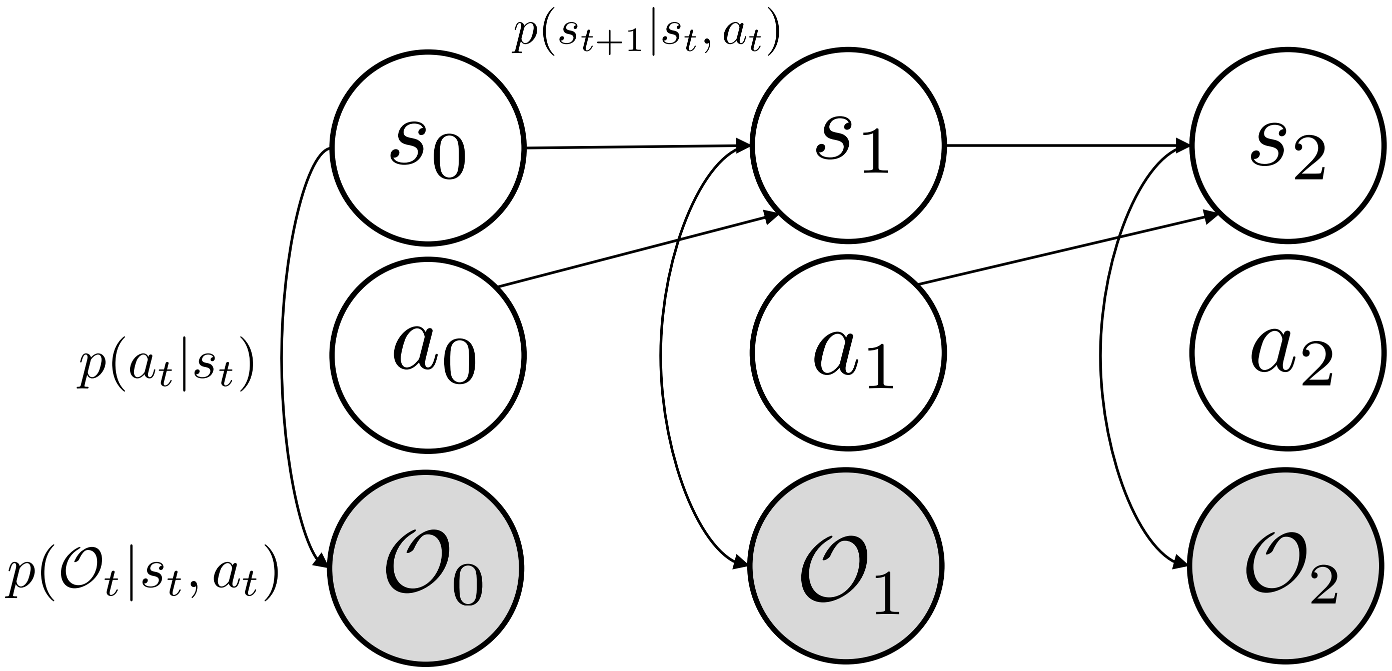
Analogues for optimal Q and V

$$V(s_t) = \log p(\mathcal{O}_{t:T} = 1|s_t)$$

$$Q(s_t, a_t) = \log p(\mathcal{O}_{t:T} = 1|s_t, a_t)$$

"Likelihood of being optimal in the future at some state, action"

Why isn't this trivial?



$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau)\exp\left(\sum_{t=0}^{T} r(s_t, a_t)\right)$$

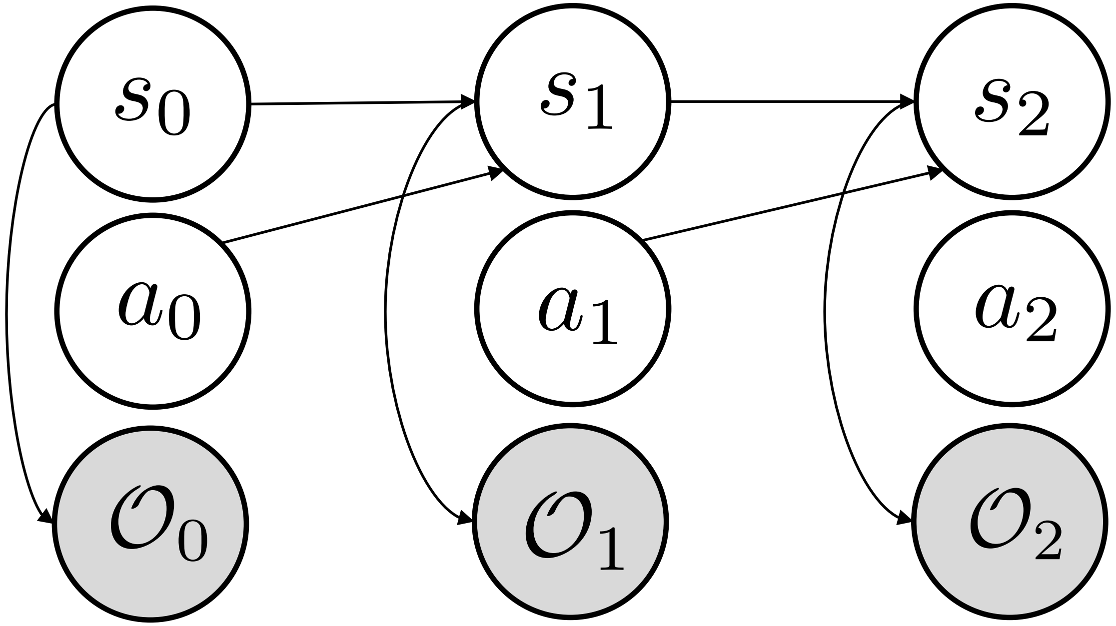
Optimal Policy → Posterior Inference

$$p(a_t|s_t, \mathcal{O}_{t:T} = 1) = \frac{p(a_t, \mathcal{O}_{t:T} = 1|s_t)}{p(\mathcal{O}_{t:T} = 1|s_t)} = \frac{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T})ds_{t+1:T}da_{t+1:T}}{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T})ds_{t+1:T}da_{t:T}}$$

"Given that you are acting optimally, what is the likelihood of a particular action at a state"

Difficult/intractable to compute
→ Most RL algorithms are approximations to this

What makes this so cool?



Optimal Policy → Posterior Inference

$$p(a_t|s_t, \mathcal{O}_{t:T} = 1)$$

$$= \frac{p(a_t, \mathcal{O}_{t:T} = 1|s_t)}{p(\mathcal{O}_{t:T} = 1|s_t)}$$

$$= \frac{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T})ds_{t+1:T}da_{t+1:T}}{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T})ds_{t+1:T}da_{t:T}}$$

<u>Policy Gradient</u>

Variational Inference lower bound solved with Gradient Ascent

<u>Approximate DP</u>

Variational Inference lower bound solved with dynamic programming

<u>Model-Based RL</u>

Posterior Inference Approximated with Monte-Carlo Samples

Can derive old algorithms + new classes of algorithms from the same framework!

Lecture outline

Model based RL v2 → uncertainty based models

↓

Model based RL v3 → policy optimization with models

↓

Model based RL v4 → latent space models with images

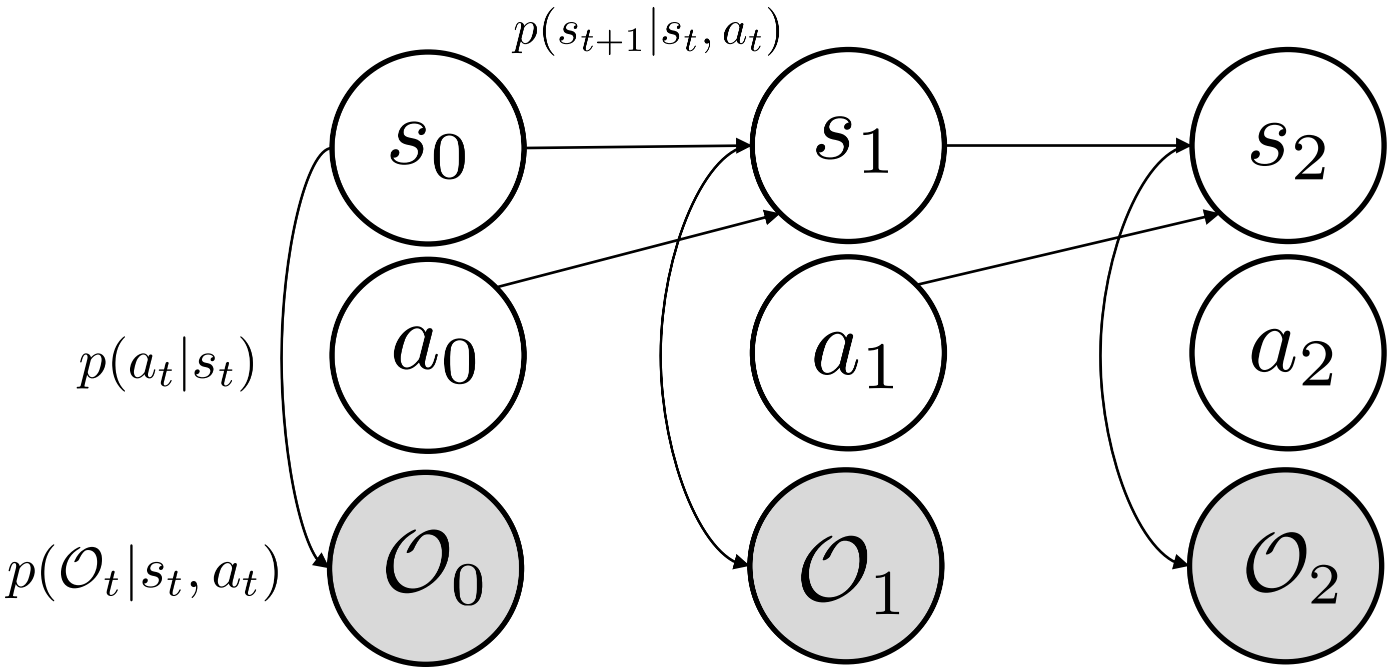
↓

Control as Inference - Formulation

↓

Variational Inference

Why isn't this trivial?



$$p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$$

$$p(\tau|\mathcal{O}_{0:T} = 1) \propto p(\tau) \exp(\sum_{t=0}^{T} r(s_t, a_t))$$

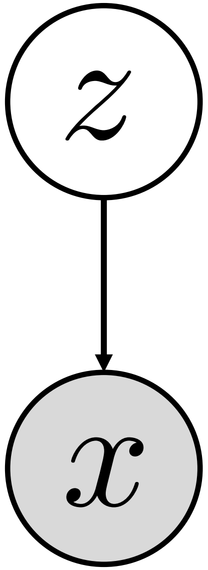
Optimal Policy → Posterior Inference

$$p(a_t|s_t, \mathcal{O}_{t:T} = 1) = \frac{p(a_t, \mathcal{O}_{t:T} = 1|s_t)}{p(\mathcal{O}_{t:T} = 1|s_t)} = \frac{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T}) ds_{t+1:T} da_{t+1:T}}{\int\int\cdots\int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T}) ds_{t+1:T} da_{t:T}}$$

"Given that you are acting optimally, what is the likelihood of a particular action at a state"

Difficult/intractable to compute
→ Most RL algorithms are approximations to this

Let's take the simplest possible example



Standard latent-variable model

Let us assume $p(x|z)$ is known, as is $p(z)$

Goal: Infer posterior $p(z|x)$

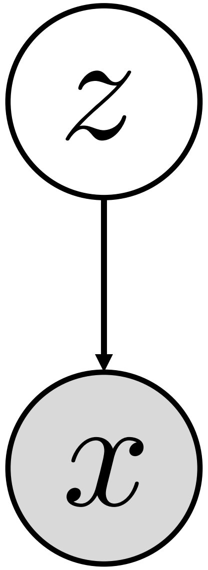
$$p(z|x) = \frac{p(x,z)}{p(x)} = \frac{p(x|z)p(z)}{p(x)}$$

$$= \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Challenging to compute efficiently with samples

Blei et al

So how can we solve this posterior inference problem?



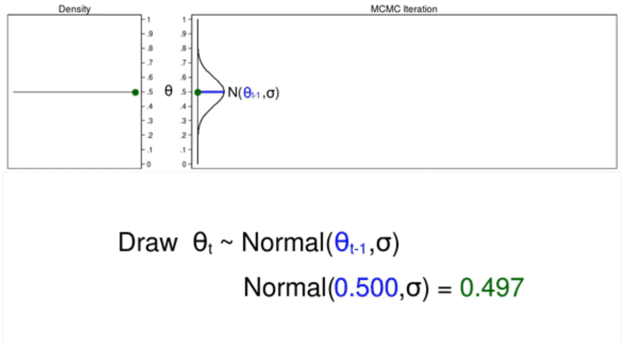
Let us assume $p(x|z)$ is known, as is $p(z)$

Goal: Infer posterior $p(z|x)$

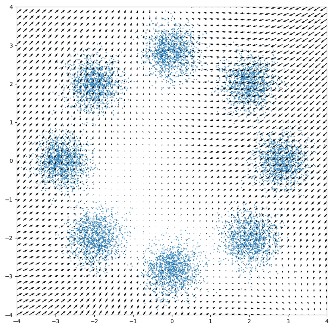
$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Challenging to compute efficiently with samples

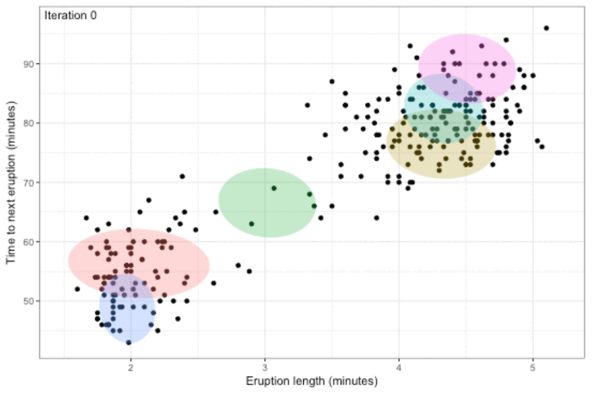
MCMC



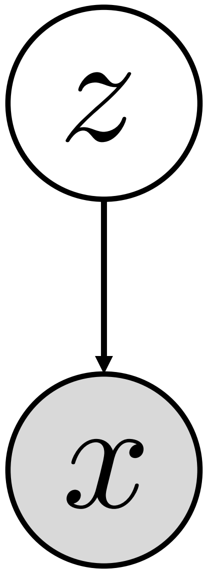
EBMs and Score Matching



Variational Inference



So how can we solve this posterior inference problem?



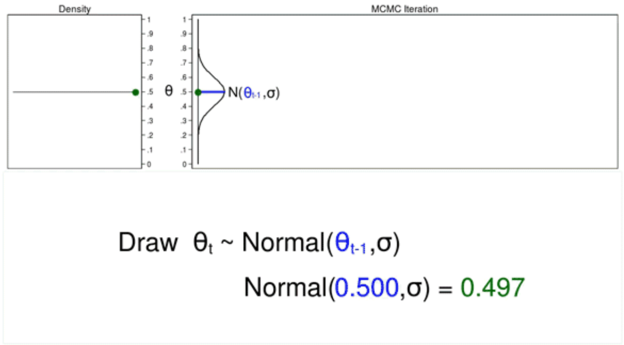
Let us assume $p(x|z)$ is known, as is $p(z)$

Goal: Infer posterior $p(z|x)$

$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Challenging to compute efficiently with samples

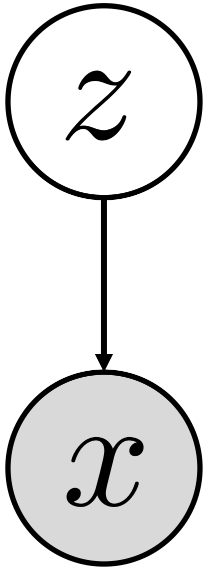
<u>MCMC</u>



Construct a Markov chain whose stationary distribution = desired distribution

Sample by just running Markov chain forward

So how can we solve this posterior inference problem?



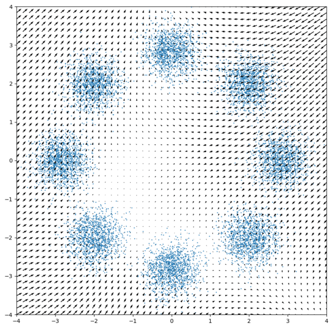
Let us assume $p(x|z)$ is known, as is $p(z)$

Goal: Infer posterior $p(z|x)$

$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Challenging to compute efficiently with samples

<u>EBMs and Score Matching</u>



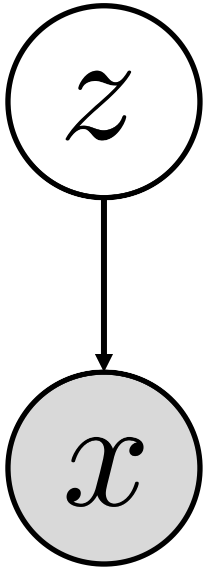
Partition function hard to compute → compute score function

$$\nabla_z \log p(z|x) = \nabla_z (\log p(x|z) + \log p(z) - \cancel{\log p(x)})$$

Known quantities

Can sample using Langevin dynamics → “noisy” gradient descent

So how can we solve this posterior inference problem?



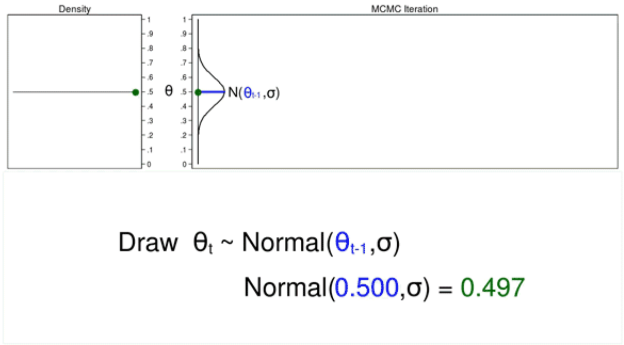
Let us assume $p(x|z)$ is known, as is $p(z)$

Goal: Infer posterior $p(z|x)$

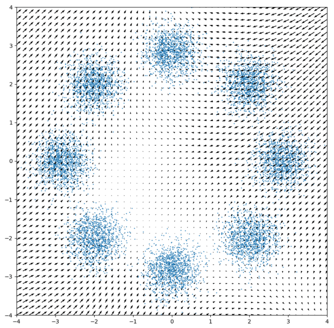
$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Challenging to compute efficiently with samples

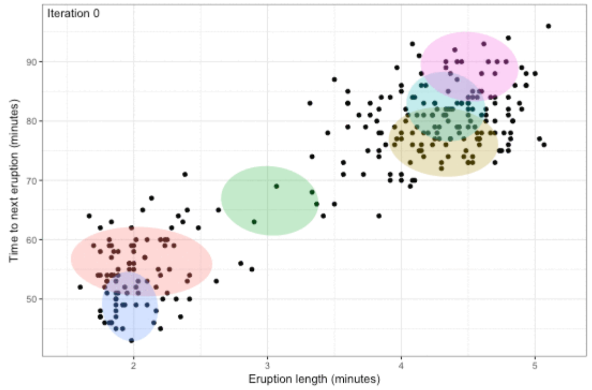
<u>MCMC</u>



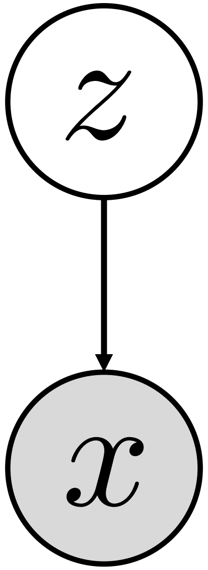
<u>EBMs and Score Matching</u>



<u>Variational Inference</u>

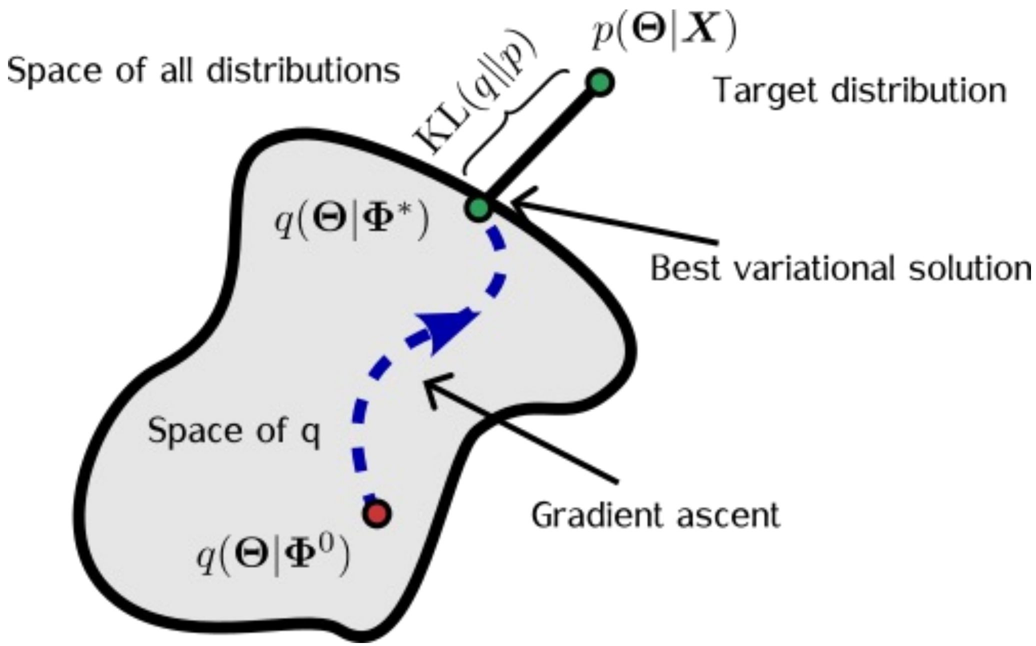


What is the key idea behind variational inference?



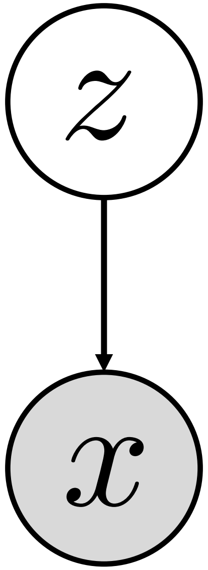
$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Intractable!



Approximate challenging posterior with closest possible “tractable” posterior

Let's derive the Evidence Lower Bound



$$p(z|x) = \frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}$$

Introduce a "tractable" approximatino $q(z|x)$
e.g. Gaussian

Intractable!

Can choose **whatever** variational family you want
→ it's an approximation!

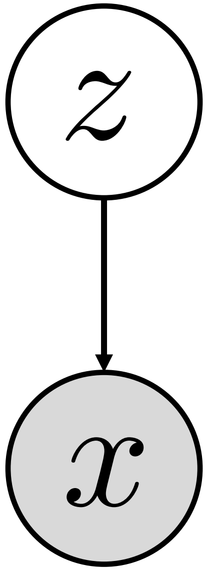
$$\phi^* \leftarrow \arg\min_{\phi} D_{KL}(q_\phi(z|x)||p(z|x))$$

Unknown

Known

How can we tractably approximate this objective?

Let's derive the Evidence Lower Bound



$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Intractable!

Unknown

$$\phi^* \leftarrow \arg\min_{\phi} D_{KL}(q_\phi(z|x)||p(z|x))$$

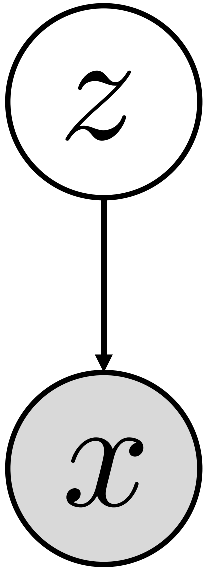
Known

$$D_{KL}(q_\phi(z|x)||p(z|x)) = \int q(z|x) \log \frac{q(z|x)}{p(z|x)} dz = \int q(z|x) \log \frac{q(z|x)p(x)}{p(x|z)p(z)} dz$$

$$= \int q(z|x) \log \frac{q(z|x)}{p(z)} dz - \int q(z|x) \log p(x|z) dz + \log p(x)$$

$$= D_{KL}(q(z|x)||p(z)) - \mathbb{E}_{z \sim q(z|x)} [\log p(x|z)] + \log p(x)$$

Let's derive the Evidence Lower Bound



$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Intractable!

Unknown

$$\phi^* \leftarrow \arg\min_{\phi} D_{KL}(q_\phi(z|x)||p(z|x))$$

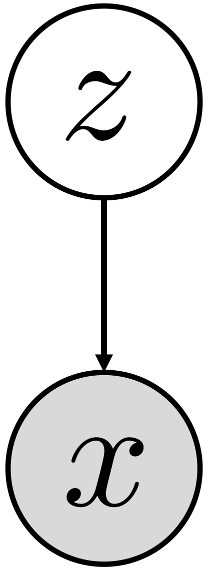
Known

$$D_{KL}(q_\phi(z|x)||p(z|x)) = D_{KL}(q(z|x)||p(z)) - \mathbb{E}_{z\sim q(z|x)}[\log p(x|z)] + \log p(x)$$

View 1: Find best posterior

View 2: Maximize marginal likelihood

Evidence Lower Bound: Best Posterior



$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Intractable!

View 1: Find best posterior

$$D_{KL}(q_\phi(z|x)||p(z|x))$$

$$= D_{KL}(q(z|x)||p(z)) - \mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] + \log p(x)$$

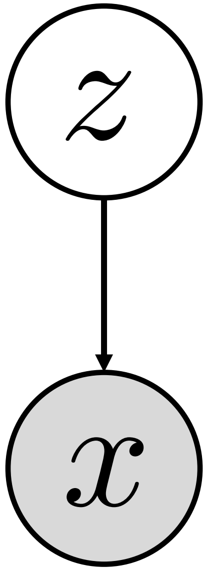
Likelihood/prior known – posterior hard to compute

Maximum likelihood — Stay close to the prior

$$\max_q \mathbb{E}_{x\sim p(x)}\left[\mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] - D_{KL}(q(z|x)||p(z))\right]$$

Learn a tractable posterior q(z|x) with known likelihood and sampling

Evidence Lower Bound: Max Marginal Likelihood



$$p(z|x) = \frac{p(x|z)p(z)}{p(x)}$$

Intractable!

View 2: Maximize marginal likelihood

$$D_{KL}(q_\phi(z|x)||p(z|x))$$

$$= D_{KL}(q(z|x)||p(z)) - \mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] + \log p(x)$$

Likelihood unknown and posterior hard to compute

$$\log p(x) - D_{KL}(q(z|x)||p(z|x)) = \mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] - D_{KL}(q(z|x)||p(z))$$

$$D_{KL}(p||q) \geq 0$$

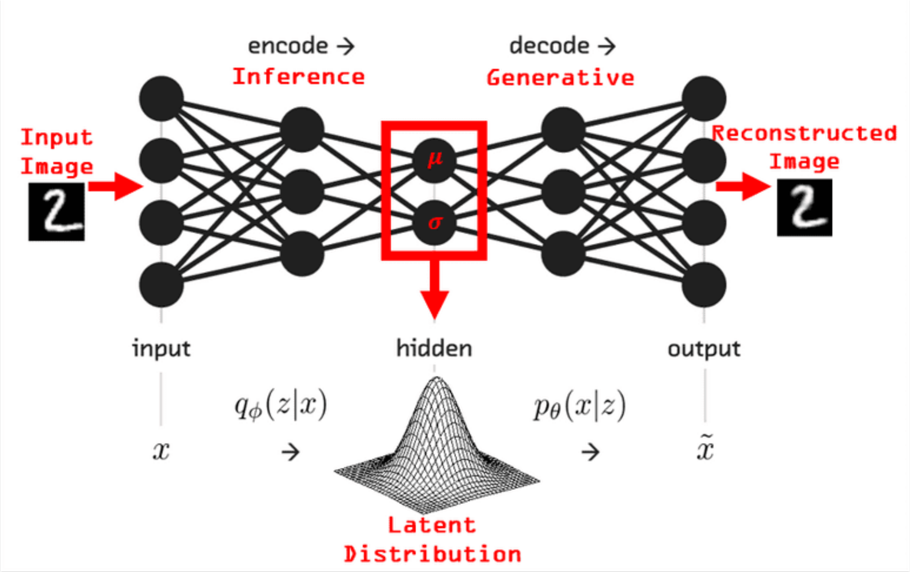
$$\log p(x) \geq \mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] - D_{KL}(q(z|x)||p(z))$$

Evidence **lower** bound – maximize to maximize likelihood

Learned

Aside: Connection to Variational Autoencoders

Popular technique for generative modeling – variational autoencoders



Encoder $q(z|x)$ Decoder $\mathrm{p}(z|x)$ Prior $\mathrm{p}(z)$

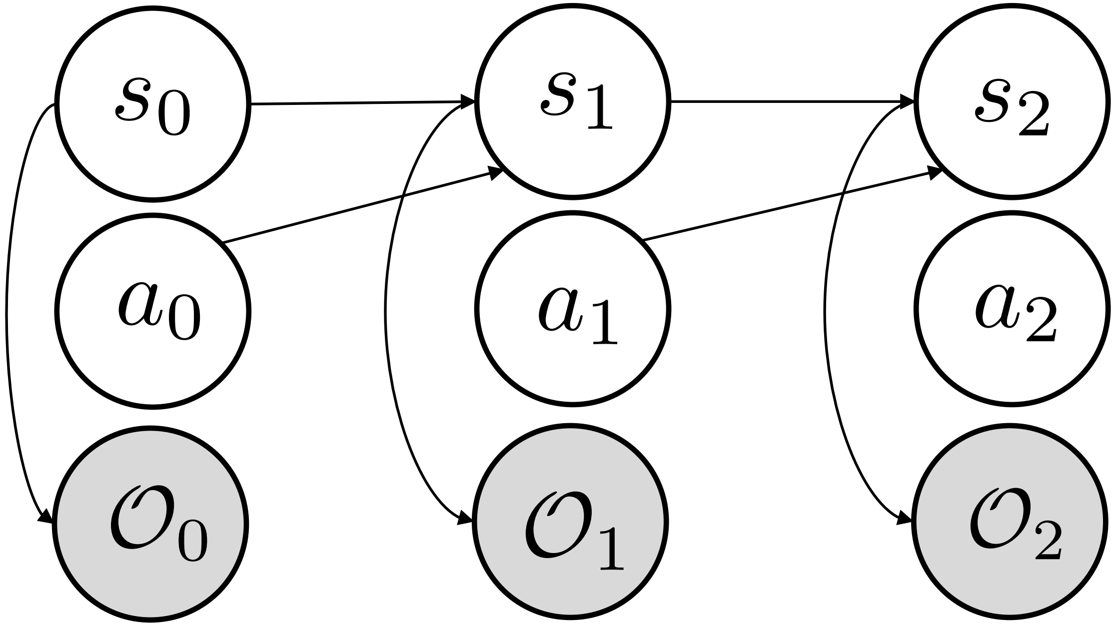
$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)} [\log p(x|z)] - D_{KL}(q(z|x)||p(z))$$

Reconstruction Prior Matching

This is one specific instantiation where encoder and decoder are both learned, goal is to sample from multimodal p(x)

Kingma et al

Lets revisit our original inference problem in control



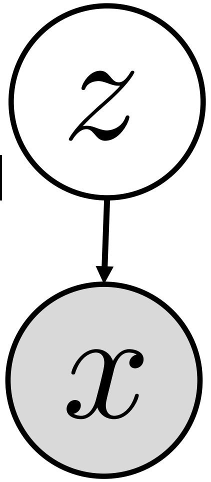
Optimal Policy → Posterior Inference

$$p(a_t|s_t, \mathcal{O}_{t:T} = 1)$$

$$= \frac{p(a_t, \mathcal{O}_{t:T} = 1|s_t)}{p(\mathcal{O}_{t:T} = 1|s_t)}$$

$$= \frac{\int\int \cdots \int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T}) ds_{t+1:T} da_{t+1:T}}{\int\int \cdots \int p(a_{t:T}, \mathcal{O}_{t:T} = 1, s_{t:T}) ds_{t+1:T} da_{t:T}}$$

Variational Inference



Approximate $p(a_t|s_t, \mathcal{O}_{t:T} = 1)$ by $q(a_t|s_t, \mathcal{O}_{t:T} = 1)$

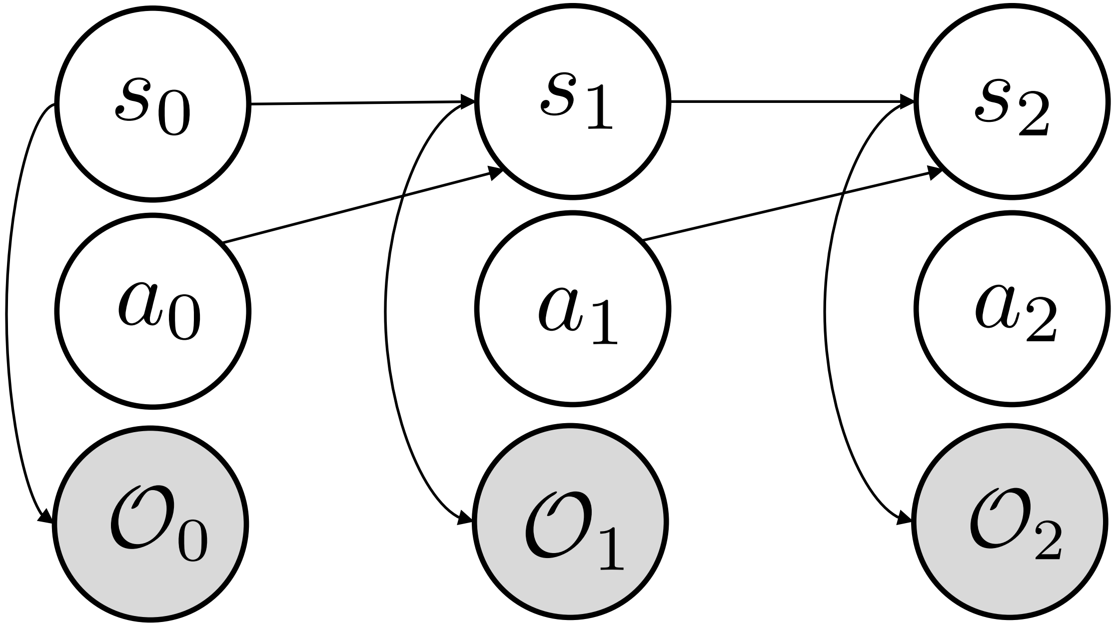
$$\max_q \mathbb{E}_{x \sim p(x)} \left[\mathbb{E}_{z \sim q(z|x)} \left[\log p(x|z) \right] - D_{KL}(q(z|x)||p(z)) \right]$$

x $\updownarrow$ $(\mathcal{O}_0, \mathcal{O}_1, \ldots, \mathcal{O}_T)$

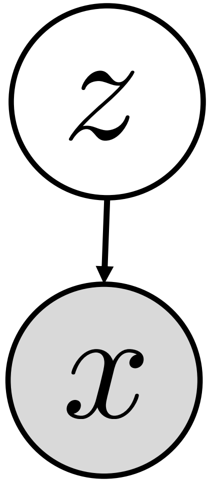
Tractable techniques for posterior policy computation

z $\updownarrow$ $(s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$

Lets revisit our original inference problem in control



Variational Inference



$$\max_q \mathbb{E}_{x\sim p(x)}\left[\mathbb{E}_{z\sim q(z|x)}\left[\log p(x|z)\right] - D_{KL}(q(z|x)||p(z))\right]$$

$$x \quad\quad\quad\quad z$$

$$\updownarrow \quad\quad\quad\quad \updownarrow$$

$$(\mathcal{O}_0, \mathcal{O}_1, \ldots, \mathcal{O}_T) \quad (s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$$

Next lecture –
derive ELBO and work out how to compute
Policy gradient/Actor-Critic

Class Structure