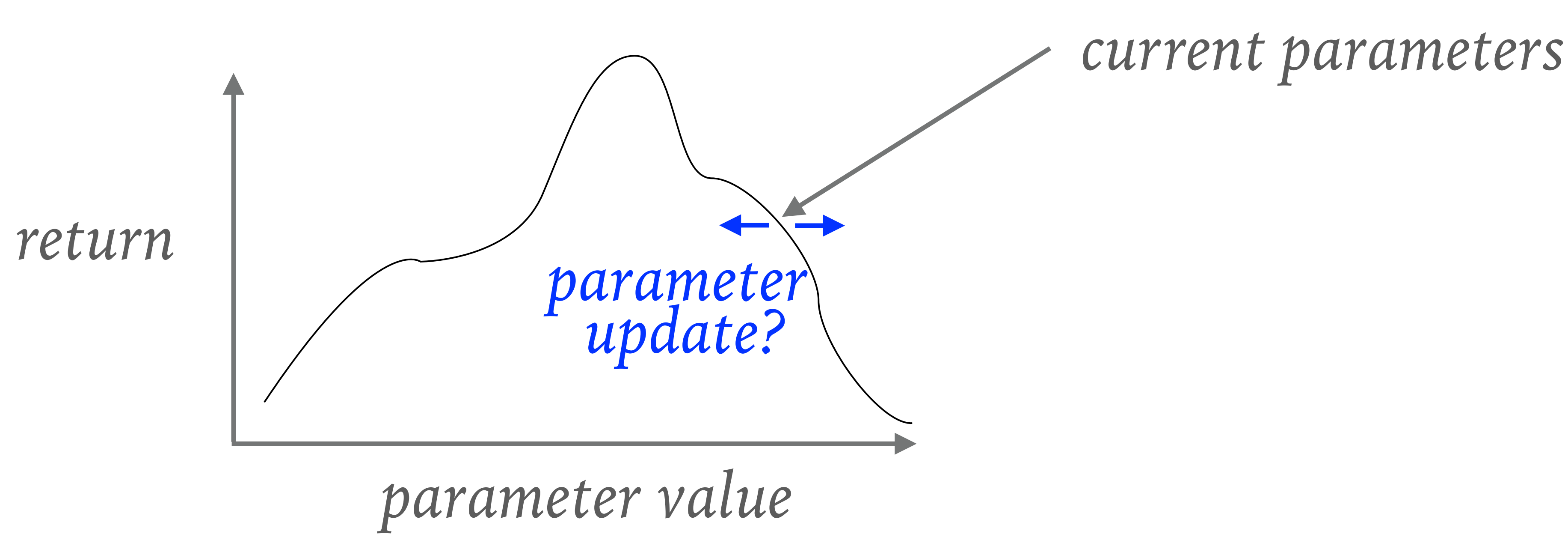
ADVANCED POLICY SEARCH METHODS

Herke van Hoof

RECAP: WHAT IS POLICY SEARCH

- Objective: find policy with maximum return
- Explicitly represent policy, usually parametric $\pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})$
- Expected return, e.g.
  - discounted cumulative reward $J(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{T}}\left[\sum_{i=1}^{T} \gamma^i r(\mathbf{s}_i, \mathbf{a}_i) \middle| \boldsymbol{\theta}\right]$
  - average reward $J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s},\mathbf{a}}\left[r(\mathbf{s}, \mathbf{a})|\boldsymbol{\theta}\right]$



RECAP: WHAT IS POLICY SEARCH

- Objective: find policy with maximum return
- Explicitly represent policy, usually parametric $\pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})$
- Expected return, e.g.
  - discounted cumulative reward $J(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{T}}\left[\sum_{i=1}^{T} \gamma^i r(\mathbf{s}_i, \mathbf{a}_i) \middle| \boldsymbol{\theta}\right]$
  - average reward $J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s},\mathbf{a}}\left[r(\mathbf{s}, \mathbf{a}) | \boldsymbol{\theta}\right]$
- Fix parameters over an episode: use any zero-order optimiser (direct policy search)
- Many parameters, or high variance: use intermediate steps (e.g. policy gradient theorem)

RECAP: WHY POLICY SEARCH INSTEAD OF POLICY ITERATION?

- Policy iteration: fit Q or V, then greedy policy wrt these

- Finding max at each step is costly with continuous actions
- PS converges to local optimum (approximate PI not always)
- Arguably easier to use prior knowledge as initial policy
- Staying close to previous policy tends to be more 'safe'
  - Knowledge is most reliable in frequently visited states
  - Do not forget what was previously learned

RECAP: WHY POLICY SEARCH INSTEAD OF POLICY ITERATION?

- These advantages especially important for physical systems!
  - Finding max costly with continuous actions
  - Stable convergence to local optimum
  - Arguably easier to use prior knowledge as initial policy
  - Staying close to data

RECAP: WHY POLICY SEARCH INSTEAD OF POLICY ITERATION?

- These advantages especially important for physical systems!
  - Finding max costly with continuous actions
    - Physical systems usually have continuous controls
  - Stable convergence to local optimum
    - Usually limited no. of samples, fitting V can be unstable
  - Arguably easier to use prior knowledge as initial policy
    - Demonstration or designed policy often available
  - Staying close to data
    - One ‘wild’ rollout could destroy something!

STAYING CLOSE TO PREVIOUS POLICIES

- Staying close to previous policy tends to be more ‘safe’
- Estimated value function can be imprecise (approximation or estimation errors)
- So we don’t want to fully trust the current best guess!

STAYING CLOSE TO PREVIOUS POLICIES

- Staying close to previous policy tends to be more ‘safe’
- Estimated value function can be imprecise (approximation or estimation errors)
- So we don’t want to fully trust the current best guess!

World	+1	+0	+0	+0	+0.9

Initial policy:
50% left or right

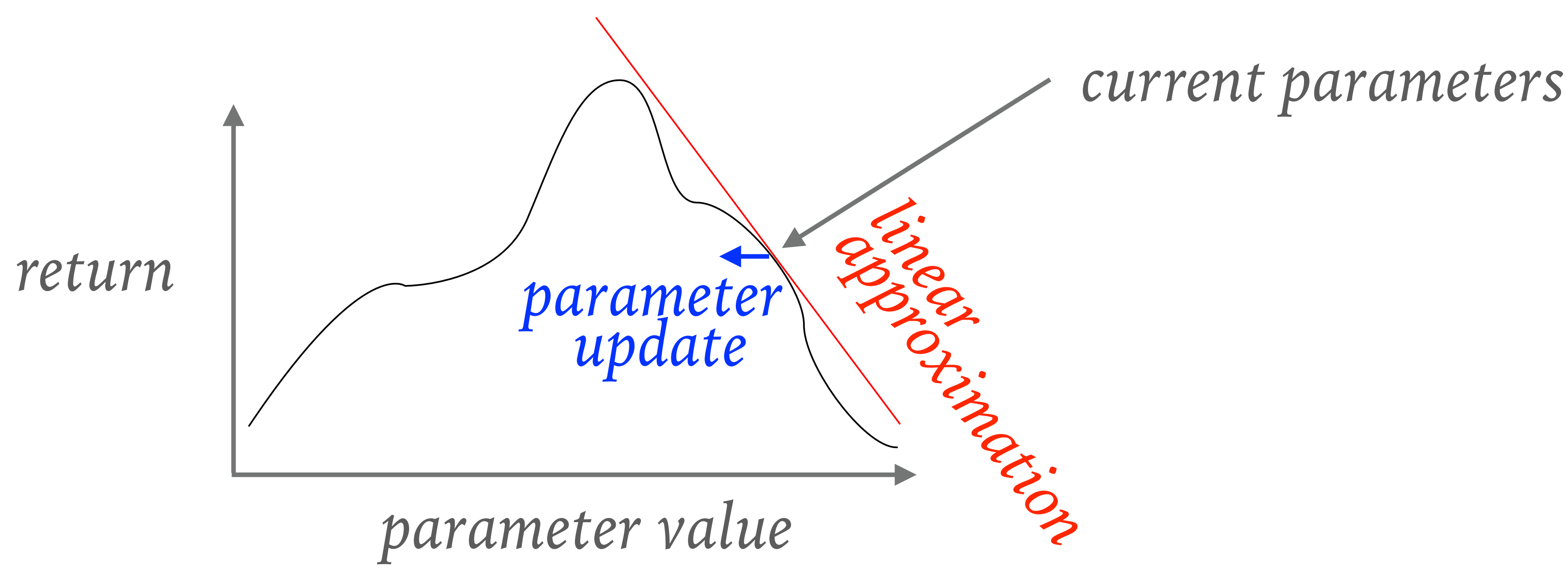
True V	+10	+9	+8.1	+8.1	+9
Estimated V	+8.1	7.3	+7.3	+8.1	+9

greedy policy

Updated policy:
100% right

STAYING CLOSE TO PREVIOUS POLICIES

- Staying close to previous policy tends to be more 'safe'
- Estimated value function can be imprecise (approximation or estimation errors)
- So we don't want to fully trust the current best guess!
- Normal policy gradient: small step in direction of best policy



STAYING CLOSE TO PREVIOUS POLICIES

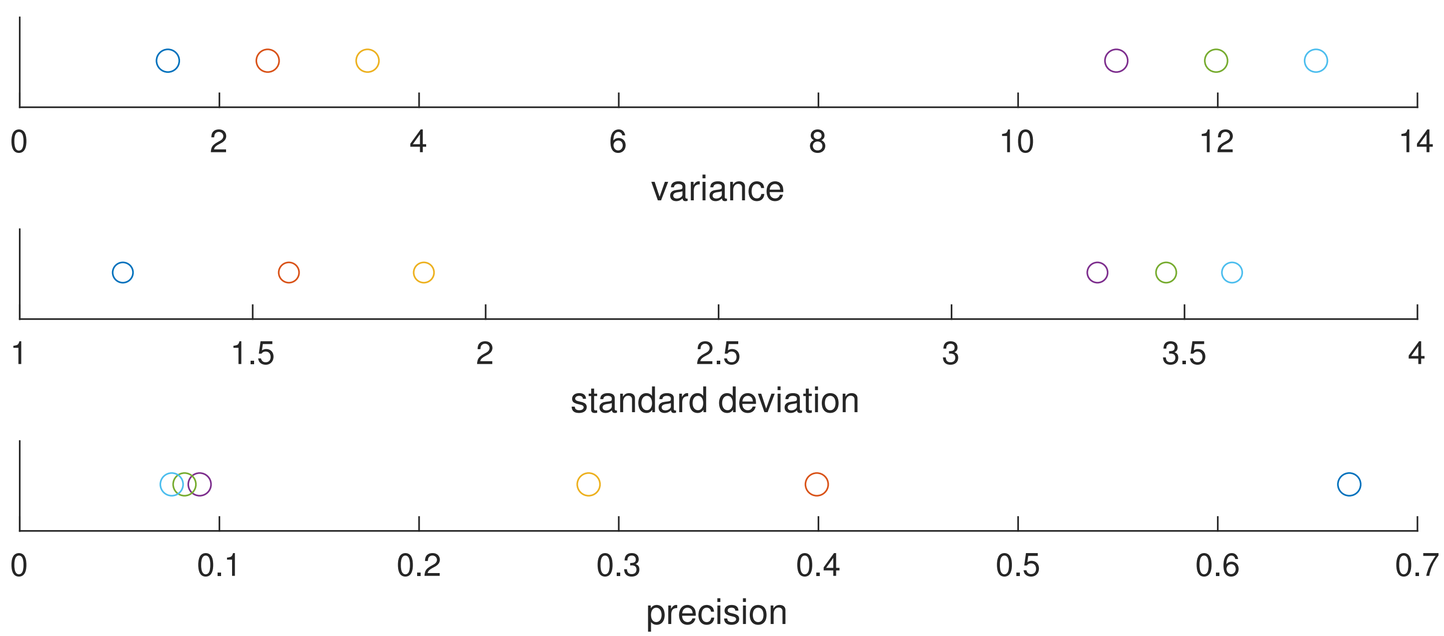
- Staying close to previous policy tends to be more 'safe'
- How close are subsequent policies?
- Limit update norm

$$
\begin{aligned}
\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 &= \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) && \text{s.t. } d\boldsymbol{\theta}^T d\boldsymbol{\theta} = c \\
&\approx \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) + (\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0))^T d\boldsymbol{\theta} && \text{s.t. } d\boldsymbol{\theta}^T d\boldsymbol{\theta} = c \\
&\propto \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)
\end{aligned}
$$

- Policy gradient!
- Standard ('vanilla') policy gradients maximise Taylor expansion of J s.t. update is on norm sphere!

STAYING CLOSE TO PREVIOUS POLICIES

- Standard ('vanilla') policy gradients maximise Taylor expansion of J s.t. update is on norm sphere!
- Euclidean norm is sensitive to parametrisation:



- Can we express policy closeness covariantly?
- (covariant: independent of choice of parametrisation)

STAYING CLOSE TO PREVIOUS POLICIES

- Why do we want covariant norm (invariant to parametrisation)?
  - Don't waste time tuning parametrisation
  - Parameters with different 'meaning': mean and precision
    - does a norm in this space make sense?
    - step size never right on all parameters if scale different (have to take step small enough for most sensitive direction)
  - Correlations between parameters ignored (feature modulated by more parameters easier to change)
- Conceptually, it's not the change in parameters we care about!
  - Limit change in trajectories, states, and/or actions?

STAYING CLOSE TO PREVIOUS POLICIES

- How to express policy closeness covariantly?
- Kullback-Leibler (KL) divergence is information-theoretic quantification of difference between probability distributions

$$D_{\mathrm{KL}}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x$$

- Asymmetric, minimal value of 0 when p=q
- KL is invariant under parameter transformations
- Idea: KL between policies, state-action distributions, or trajectory distributions to limit policy change

STAYING CLOSE TO PREVIOUS POLICIES

- Idea: use KL to specify how the policy can change in one step
- Several algorithms can be understood using this idea
    - Natural policy gradient
    - Trust region policy optimization (TRPO)
    - Relative entropy policy search (REPS)

STAYING CLOSE TO PREVIOUS POLICIES

- Idea: use KL to specify how the policy can change in one step
- Several algorithms can be understood using this idea
  - Natural policy gradient
  - Trust region policy optimization (TRPO)
  - Relative entropy policy search (REPS)

NATURAL POLICY GRADIENT

- Idea: make policy gradients covariant [Kakade 2002]
- this yields an algorithm that exploits structure of parameters
- Here, will look how it relates to KL [Bagnell 2003]

NATURAL POLICY GRADIENT

- Recall vanilla policy gradients

$$
\begin{aligned}
\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 &= \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) && \text{s.t. } d\boldsymbol{\theta}^T d\boldsymbol{\theta} = c \\
&\approx \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) + (\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0))^T d\boldsymbol{\theta} && \text{s.t. } d\boldsymbol{\theta}^T d\boldsymbol{\theta} = c \\
&\propto \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)
\end{aligned}
$$

- replace constraint by quadratic expansion of KL divergence

$$
\begin{aligned}
c &= \mathbb{E}_{\mathbf{s}} \left[D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0) || \pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right] = \mathrm{EKL}(\boldsymbol{\theta}) \\
&\approx \mathrm{EKL}(\boldsymbol{\theta}_0) + d\boldsymbol{\theta}^T \left(\nabla_{d\boldsymbol{\theta}} \mathrm{EKL}\right)(\boldsymbol{\theta}_0) + d\boldsymbol{\theta}^T \left(\nabla^2_{d\boldsymbol{\theta}} \mathrm{EKL}\right)(\boldsymbol{\theta}_0) d\boldsymbol{\theta} \\
&\approx 0 \qquad\qquad\qquad\qquad\qquad\qquad\quad + d\boldsymbol{\theta}^T \left(\nabla^2_{d\boldsymbol{\theta}} \mathrm{EKL}\right)(\boldsymbol{\theta}_0) d\boldsymbol{\theta}
\end{aligned}
$$

- since minimal value of 0 is reached if parameter doesn’t change
- direction of KL does not matter for quadratic expansion

NATURAL POLICY GRADIENT

$$c = \mathbb{E}_{\mathbf{s}}\left[D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)||\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right] = \mathrm{EKL}(\boldsymbol{\theta})$$
$$\approx d\boldsymbol{\theta}^T \left(\nabla^2_{d\boldsymbol{\theta}}\,\mathrm{EKL}\right)(\boldsymbol{\theta}_0)d\boldsymbol{\theta}$$

- This is the squared length with respect to matrix

$$\nabla^2_{d\boldsymbol{\theta}}\,\mathrm{EKL} = \mathbb{E}_s\left[\nabla^2_{d\boldsymbol{\theta}} D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)||\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right], \text{ where}$$
$$\nabla^2_{d\boldsymbol{\theta}} D_{\mathrm{KL}} = \nabla^2_{d\boldsymbol{\theta}} \int_{\Theta} \pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0) \log \frac{\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)}{\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0 + d\boldsymbol{\theta})}$$
$$= \mathbb{E}_{\mathbf{a}|\mathbf{s};\theta_0}\left[\nabla^2_{d\boldsymbol{\theta}} \log \pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0 + d\boldsymbol{\theta})\right]$$
$$= F_{\mathbf{s}}$$

- is the Fisher information matrix of the policy!
- characterises information about parameters in observation

[Kakade 2002, Bagnell 2003]

NATURAL POLICY GRADIENT

$$c = \mathbb{E}_{\mathbf{s}}\left[D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)||\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right] = \mathrm{EKL}(\boldsymbol{\theta})$$
$$\approx d\boldsymbol{\theta}^T \left(\nabla^2_{d\boldsymbol{\theta}}\,\mathrm{EKL}\right)(\boldsymbol{\theta}_0)d\boldsymbol{\theta}$$

- This is the squared length with respect to matrix

$$F = \nabla^2_{d\boldsymbol{\theta}}\,\mathrm{EKL} = \mathbb{E}_s\left[\nabla^2_{d\boldsymbol{\theta}} D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)||\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right], \text{ where}$$
$$\nabla^2_{d\boldsymbol{\theta}} D_{\mathrm{KL}} = \nabla^2_{d\boldsymbol{\theta}} \int_\Theta \pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)\log\frac{\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)}{\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0+d\boldsymbol{\theta})}$$
$$= \mathbb{E}_{\mathbf{a}|\mathbf{s};\theta_0}\left[\nabla^2_{d\boldsymbol{\theta}}\log\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0+d\boldsymbol{\theta})\right]$$
$$= F_{\mathbf{s}}$$

- is the Fisher information matrix of the policy!
- characterises information about parameters in observation

[Kakade 2002, Bagnell 2003]

NATURAL POLICY GRADIENT

- Consider now the modified optimisation problem

$$\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 = \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0 + d\boldsymbol{\theta}) \qquad \text{s.t. } d\boldsymbol{\theta}^T F d\boldsymbol{\theta} = c$$

$$\approx \max_{d\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) + (\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0))^T d\boldsymbol{\theta} \qquad \text{s.t. } d\boldsymbol{\theta}^T F d\boldsymbol{\theta} = c$$

- solve constraint optimisation problem: Lagrangian

$$L(d\boldsymbol{\theta}, \lambda) = J(\boldsymbol{\theta}_0) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)^T d\boldsymbol{\theta} + \lambda(d\boldsymbol{\theta}^T \mathbf{F} d\boldsymbol{\theta} - c)$$

- At optimality, partial derivatives of L are 0

$$\frac{\partial L(d\boldsymbol{\theta}, \lambda)}{\partial d\boldsymbol{\theta}} = 0 \qquad (\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)) + \lambda F d\boldsymbol{\theta} = 0$$

$$\frac{\partial L(d\boldsymbol{\theta}, \lambda)}{\partial \lambda} = 0 \qquad d\boldsymbol{\theta}^T F d\boldsymbol{\theta} = c$$

[Kakade 2002, Bagnell 2003]

NATURAL POLICY GRADIENT

- So optimality conditions are

$$(\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)) + \lambda F d\boldsymbol{\theta} = 0$$
$$d\boldsymbol{\theta}^T F d\boldsymbol{\theta} = c$$

- From the first line, update direction

$$\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 \propto F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

- This is the **natural gradient**
  (natural gradients in ML used at least since [Amari, 1998], used in RL since [Kakade 2002])

NATURAL POLICY GRADIENT

- The policy is adapted using the **natural gradient**

$$\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 \propto F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

- We can use any known approach for the vanilla gradient
- Will this always improve J?
- For small enough step size, objective improves if

$$(\boldsymbol{\theta}^* - \boldsymbol{\theta}_0)^T \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) > 0$$

$$(\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0))^T F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) \overset{?}{>} 0$$

- Fisher information is positive definite!

NATURAL POLICY GRADIENT

- The policy is adapted using the **natural gradient**

$$\boldsymbol{\theta}^* - \boldsymbol{\theta}_0 \propto F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

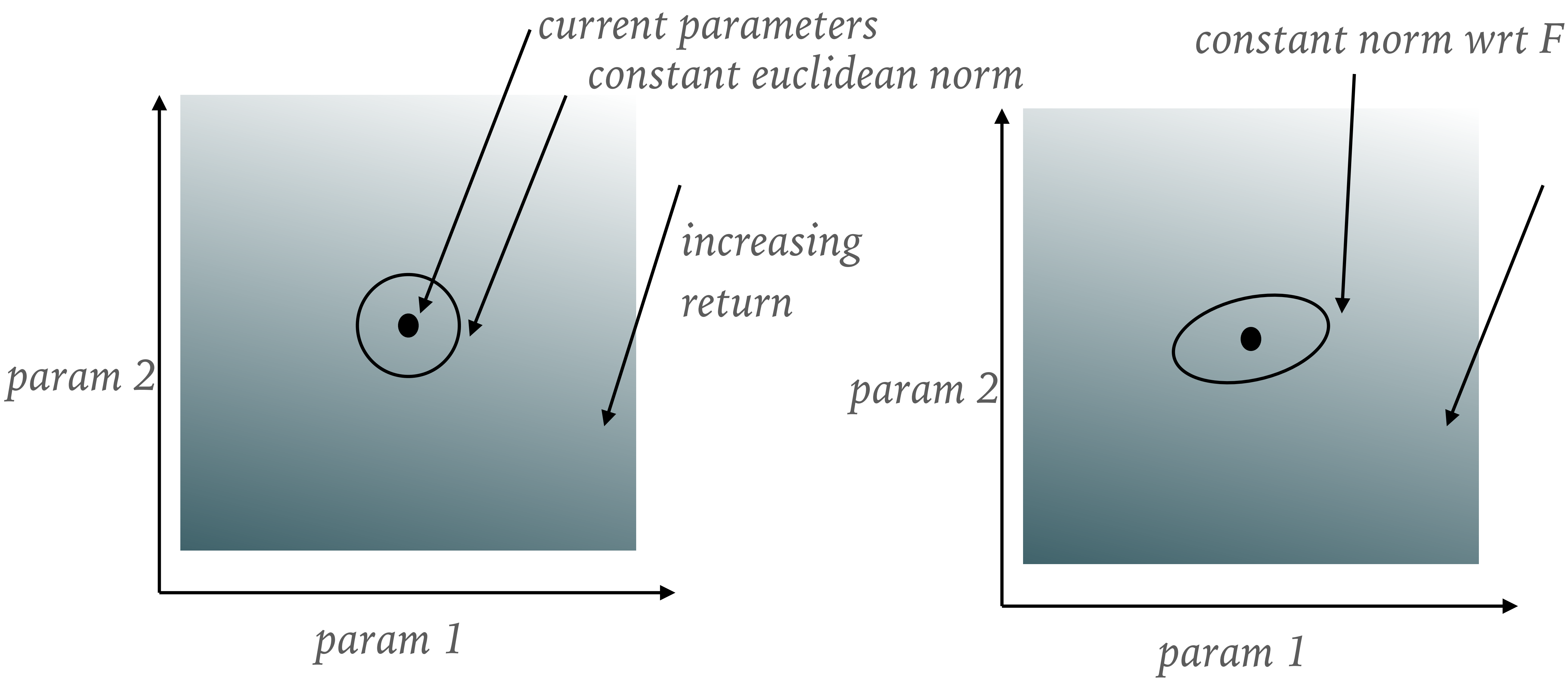
- We can use any known approach for the vanilla gradient
- Will this always improve J?
- For small enough step size, objective improves if

$$(\boldsymbol{\theta}^* - \boldsymbol{\theta}_0)^T \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) > 0$$

$$(\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0))^T F^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) \overset{?}{>} 0$$

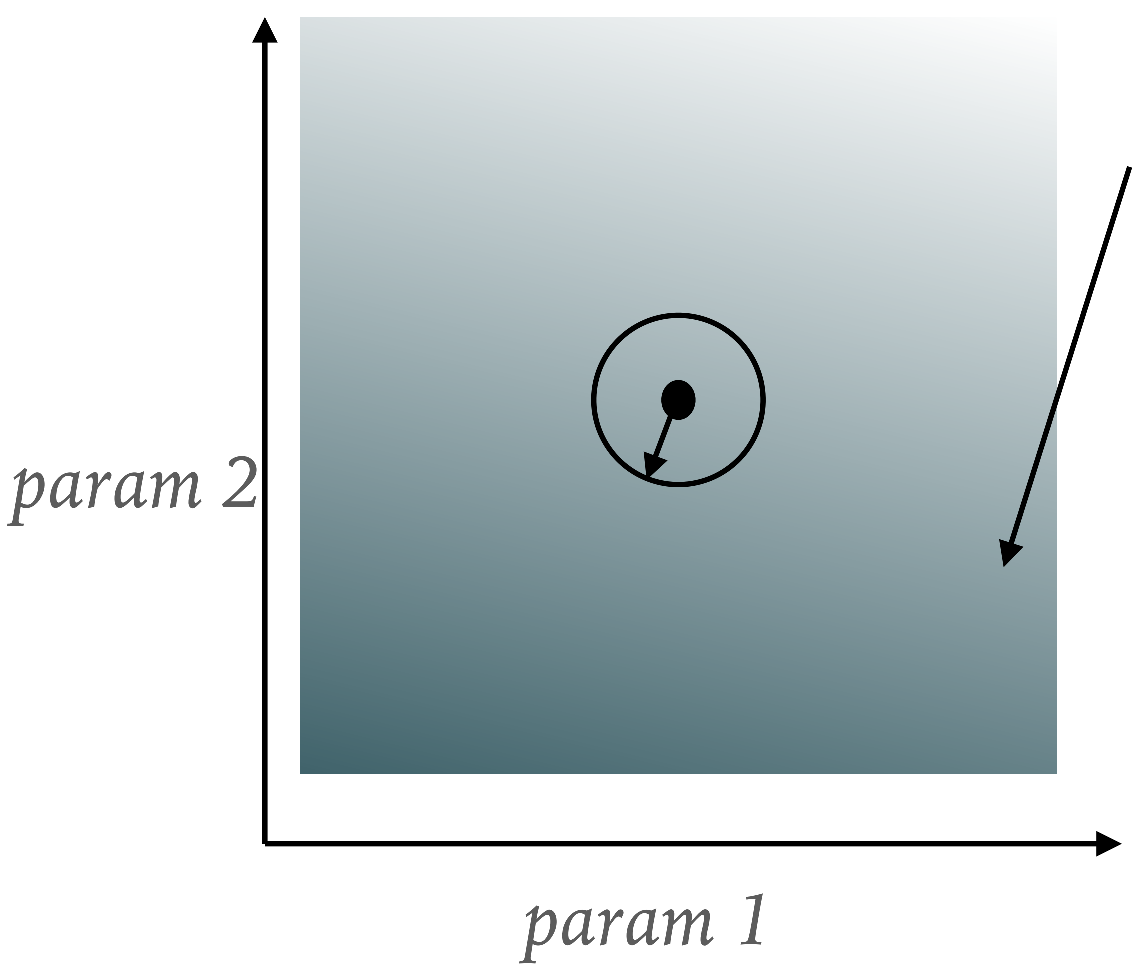
- Fisher information is positive definite! So **yes!** (geometric perspective: inner product with vanilla gradient)

NATURAL POLICY GRADIENT



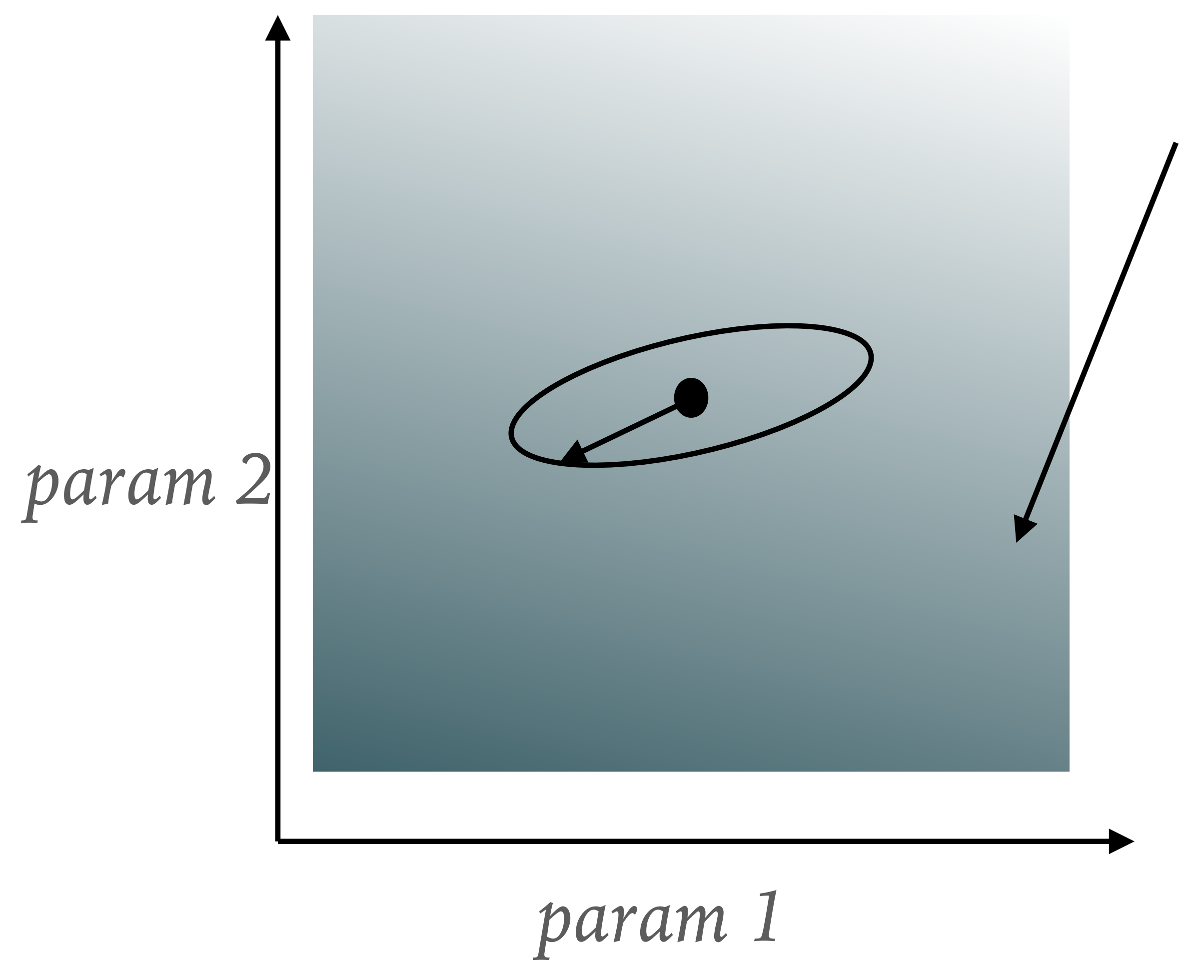
NATURAL POLICY GRADIENT

Regular gradient



Direction of expected return

Natural gradient



Within 90° of direction of return

Possibly bigger steps (depends on F)

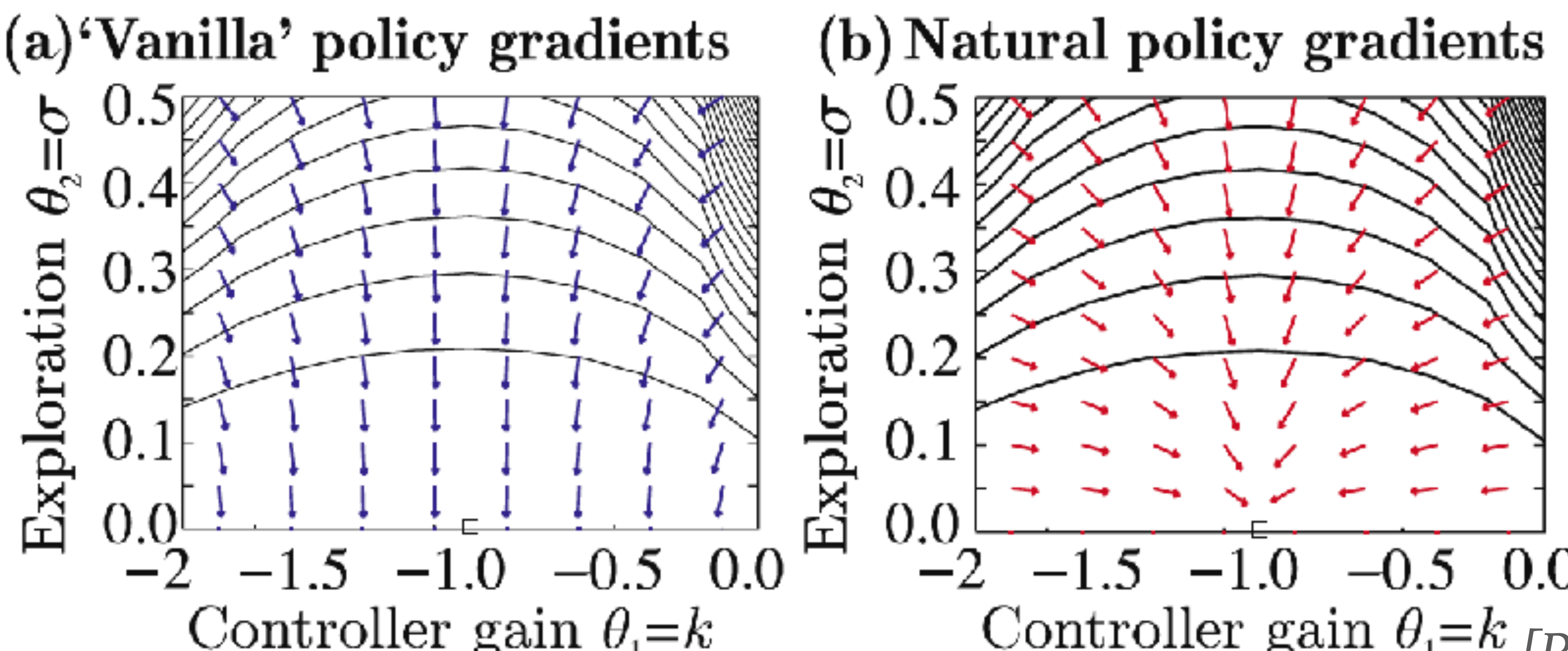
NATURAL ACTOR CRITIC

- Natural policy gradients can be used in actor-critic set-up
- Additional benefit: F cancels out!
- Natural gradients can help where the likelihood is almost flat

[Peters 2008]

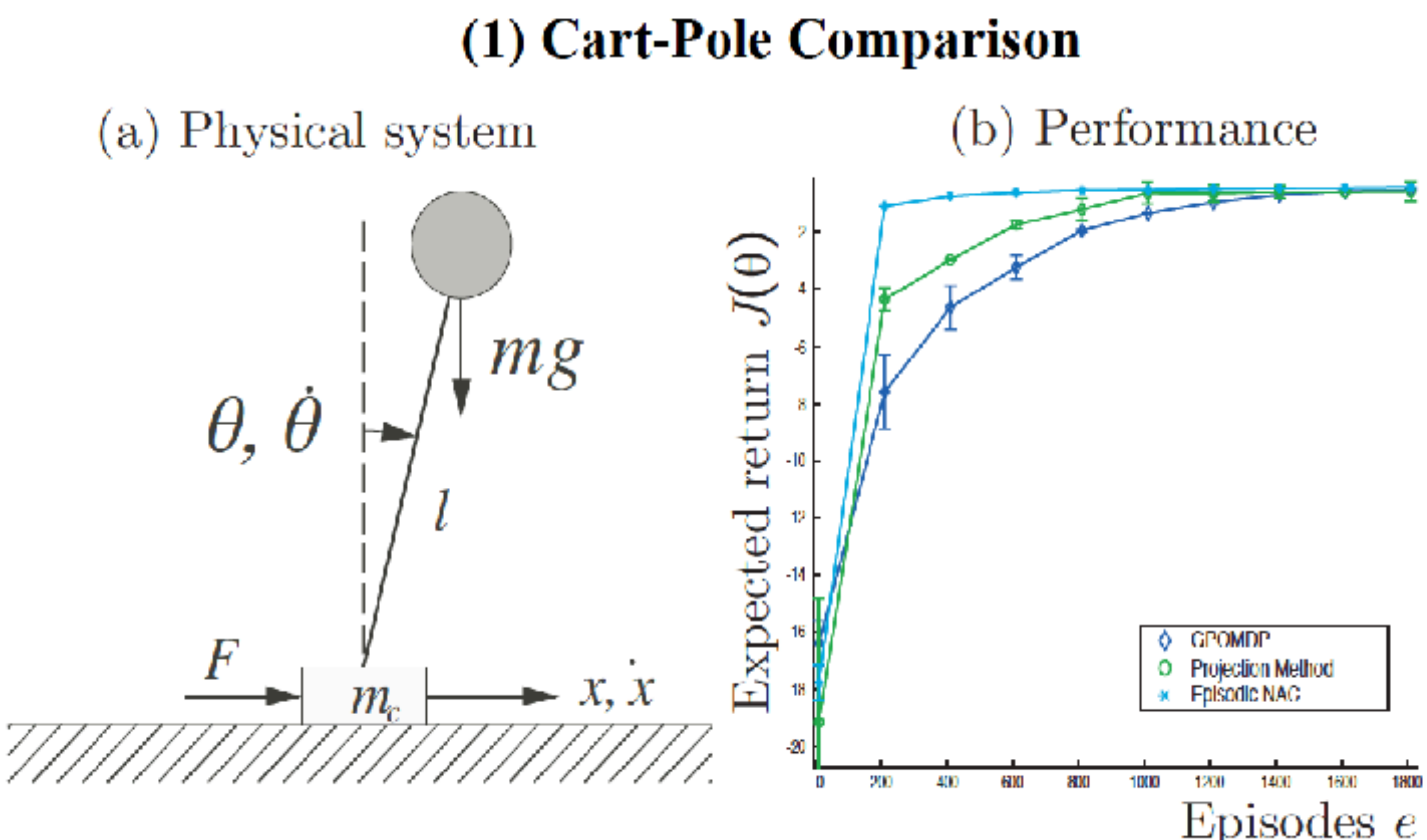
NATURAL ACTOR CRITIC

- Natural policy gradients can be used in actor-critic set-up
- Additional benefit: F cancels out!
- Natural gradients can help where the likelihood is almost flat



[Peters 2008]

NATURAL ACTOR CRITIC



[Peters 2008]

NATURAL ACTOR CRITIC EXAMPLE

[Peters 2008]

NATURAL POLICY GRADIENTS

- Advantages
  - Usually needs less training than regular policy gradients
  - Can use most tricks used for vanilla gradients
  - Inherits advantageous properties from vanilla gradients
  - Relatively easy to implement
- Limitations
  - Need Fisher information matrix
    - Known for some standard distributions, e.g. Gaussian
    - PG methods: high variance, might need many steps

STAYING CLOSE TO PREVIOUS POLICIES

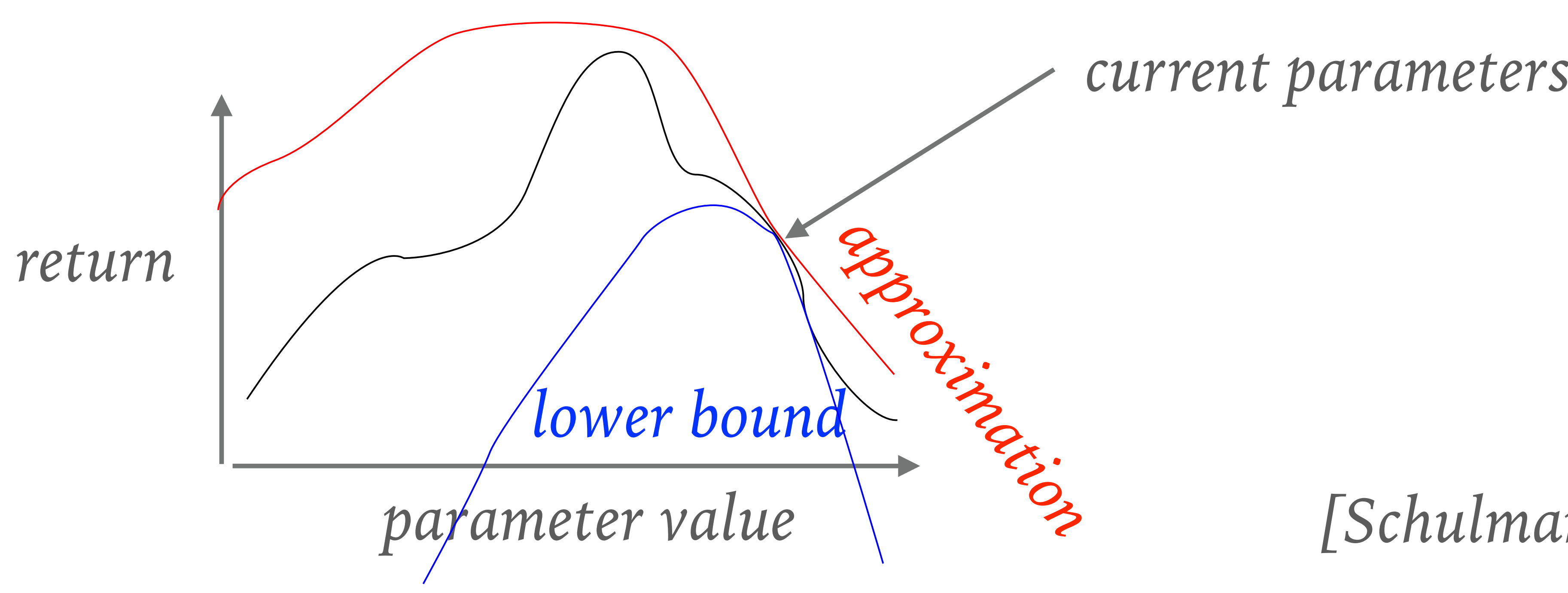
- Idea: use KL to specify how the policy can change in one step
  - Natural policy gradient
  - Trust region policy optimization (TRPO)
  - Relative entropy policy search (REPS)

TRUST REGION POLICY OPTIMISATION

- Trust region: region where approximation is valid
- Optimization step shouldn’t leave this region
- Main idea goes back long way, e.g. Levenberg (1944)
- Schulman’s “Trust Region policy optimisation” uses this notion to define a new RL algorithm
- Type of trust region motivated by theoretical bound

TRPO: THEORETICAL BOUND GUARANTEES IMPROVEMENT

- Idea: take larger steps while guaranteeing improvement
    1. approximate the return function
    2. apply a penalty term to yield lower bound
    3. maximize this lower bound



[Schulman 2016]

TRPO 1: APPROXIMATE THE RETURN FUNCTION

- Why approximate? (Simplified argument)

$$\eta(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s},\mathbf{a}}\left[r(\mathbf{s},\mathbf{a})|\boldsymbol{\theta}\right]$$

- However, samples are from previous policy.
- Know how policy changed, correct with importance sampling

$$\mathbb{E}_{\mathbf{a}\sim\pi_{\boldsymbol{\theta}}(\mathbf{s})}[r(\mathbf{s},\mathbf{a})] = \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s}) r(\mathbf{s},\mathbf{a}) d\mathbf{a}$$

$$= \int_{\mathcal{A}} \pi_{\boldsymbol{\theta}'}(\mathbf{a}|\mathbf{s}) \frac{\pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})}{\pi_{\boldsymbol{\theta}'}(\mathbf{a}|\mathbf{s})} r(\mathbf{s},\mathbf{a}) d\mathbf{a} = \mathbb{E}_{\mathbf{a}\sim\pi_{\boldsymbol{\theta}'}(\mathbf{s})}\left[\frac{\pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})}{\pi_{\boldsymbol{\theta}'}(\mathbf{a}|\mathbf{s})} r(\mathbf{s},\mathbf{a})\right]$$

- But we don't know state distribution changed! Approximate:

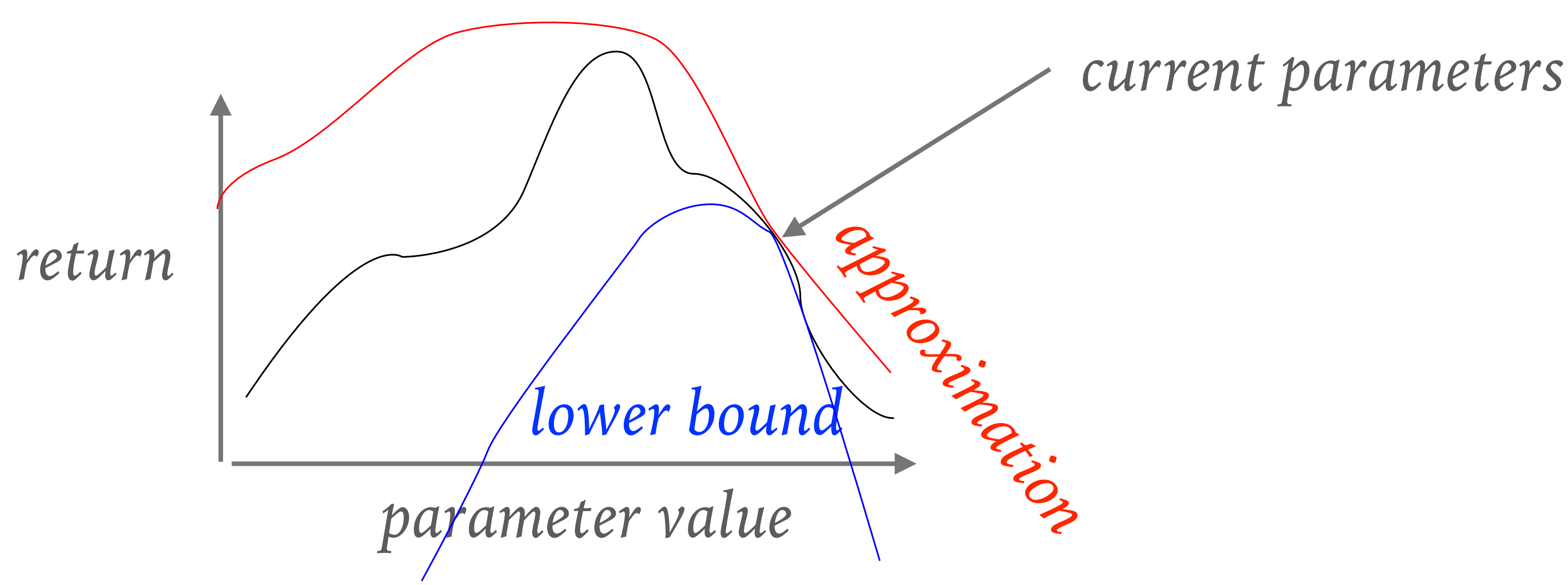
$$\eta(\boldsymbol{\theta}) \approx L_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s},\mathbf{a}}\left[\frac{\pi_{\boldsymbol{\theta}}(\mathbf{a}|\mathbf{s})}{\pi_{\boldsymbol{\theta}'}(\mathbf{a}|\mathbf{s})} r(\mathbf{s},\mathbf{a})|\boldsymbol{\theta}'\right]$$

[Schulman 2016]

TRPO 2: GET LOWER BOUND

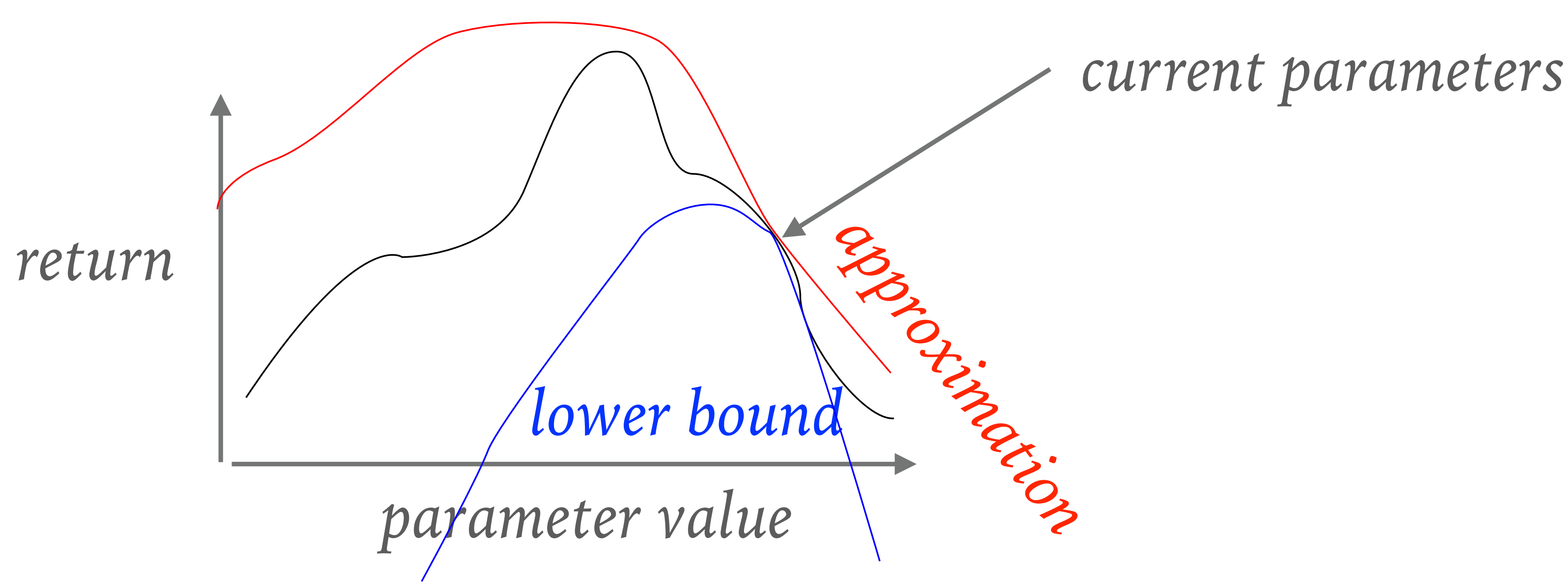
➤ [Schulman, 2016] shows the following holds:

$$\eta(\boldsymbol{\theta}) \geq \underbrace{\underbrace{L_{\boldsymbol{\theta}'}(\boldsymbol{\theta})}_{\text{approximation}} - \underbrace{\frac{2\epsilon\gamma}{(1-\gamma)^2}}_{\text{depends on problem, old policy}} \underbrace{\max_{\mathbf{s}} D_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}'}(\cdot|\mathbf{s})||\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{s}))}_{\text{maximum KL}}}_{\text{lower bound}}$$



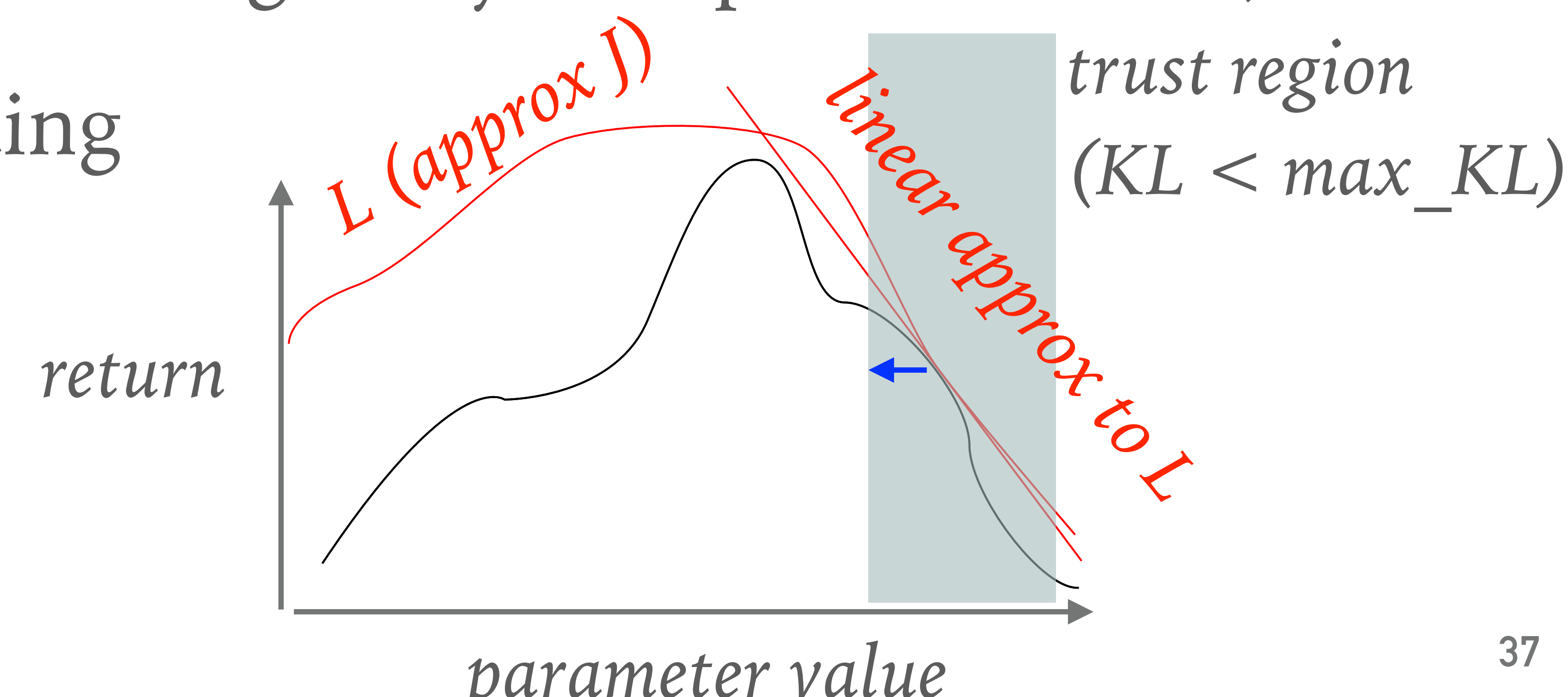
TRPO 3: FIND NEW POLICY

- Policy maximising lower-bound has guaranteed improvement
- In practice, need to approximate:
  - average KL instead of max, constraint instead of penalty
  - step in direction of natural gradient, size determined by KL

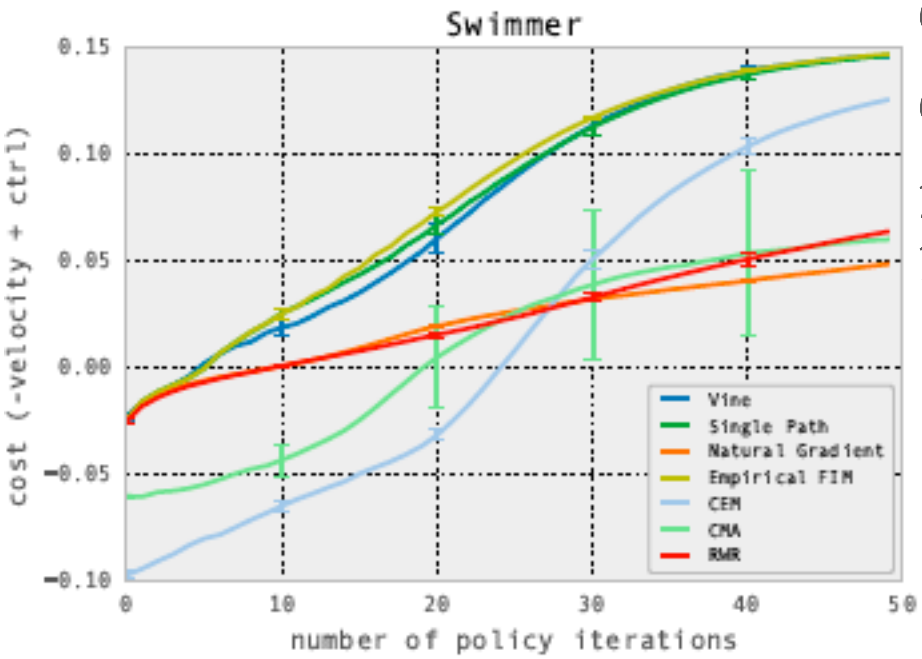


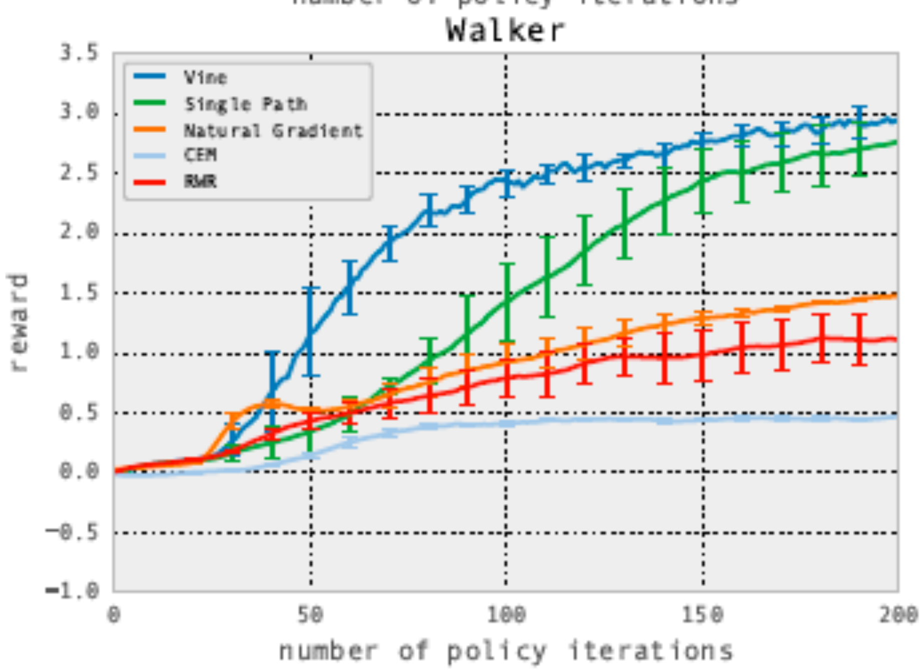
CONNECTION TO NATURAL GRADIENTS

- Natural gradients find direction that improves most s.t. KL
- Step size is manually set
- Easier: set max KL
- TRPO: solve for step size β in $D_{\mathrm{KL}} \approx \beta^2 \mathbf{s}^T F_{\mathbf{s}} \mathbf{s}/2$
- This is based on approximation (linear L, quadratic KL):
  - follow by line search using analytic expressions of L, KL
  - prevents overshooting



TRPO EVALUATION



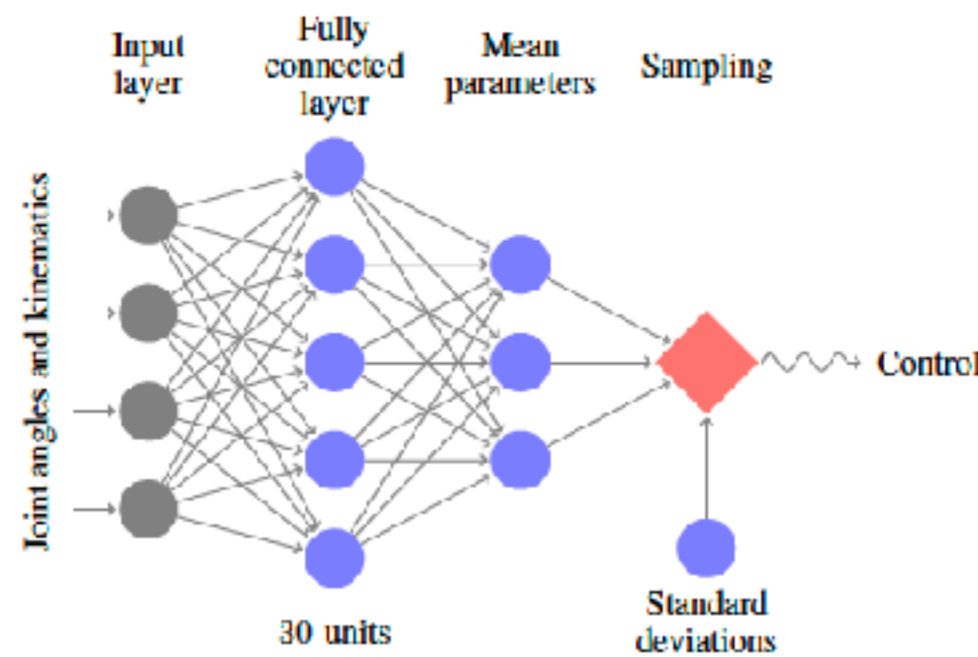


different TRPO variants

direct policy search

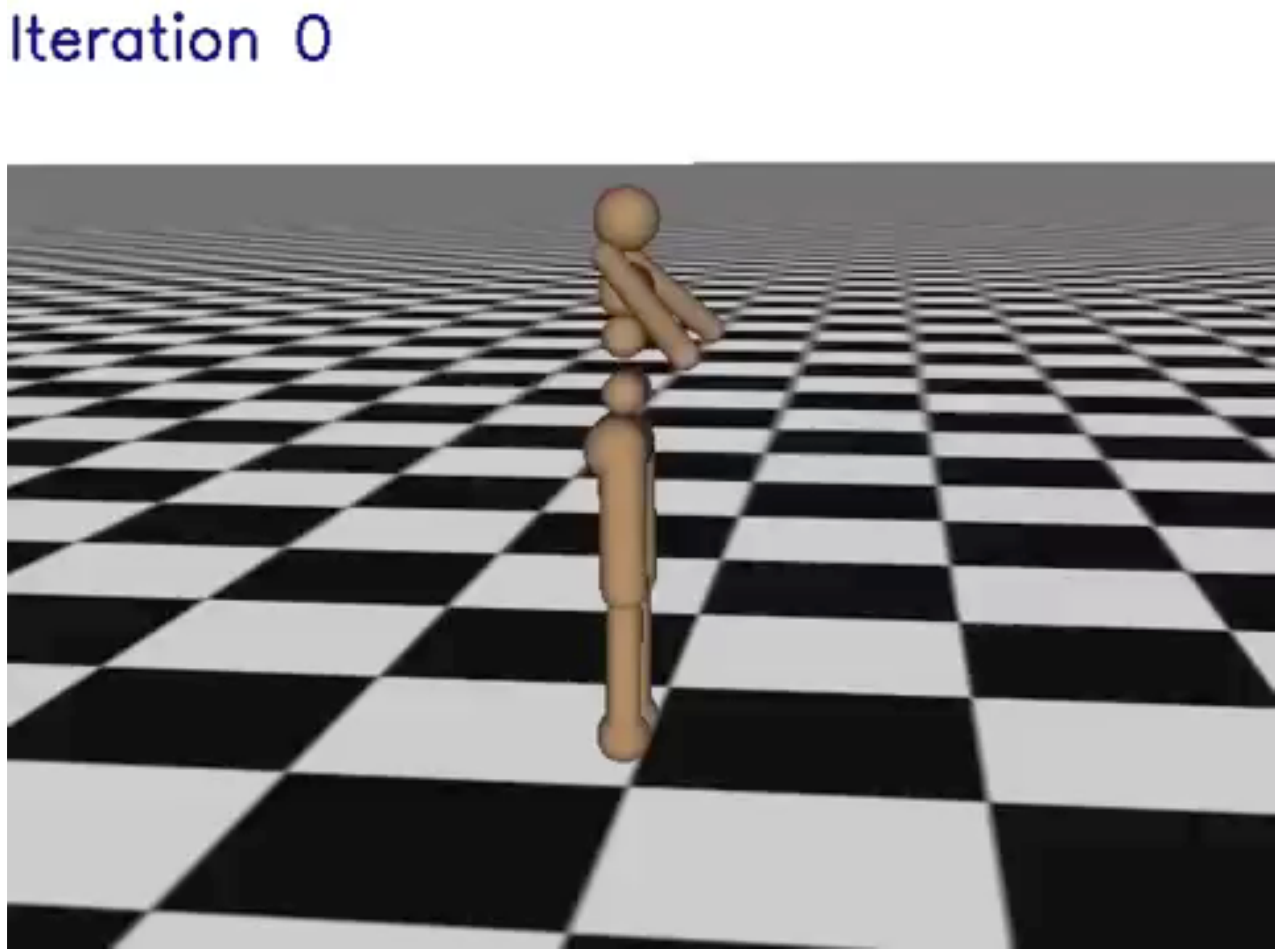
natural gradients,
reward-weighted regression

neural network policy used



[Schulman, 2015]

TRPO EXAMPLE



TRPO with generalised advantage estimate, [Schulman 2016]

TRPO

- Advantages
  - Can take larger steps than natural gradients
  - In principle, guaranteed to converge
  - Works well with neural network controllers
- Disadvantages
  - Approximations break guarantee
  - Typically, still need quite many trials
  - Need Q-estimates, can be high-variance or need simulator

STAYING CLOSE TO PREVIOUS POLICIES

- Idea: use KL to specify how the policy can change in one step
  - Natural policy gradient
  - Trust region policy optimization (TRPO)
  - Relative entropy policy search (REPS)

REPS

- Relative Entropy Policy search also uses KL divergence
- Again: stay close to previous data
  - Knowledge most reliable in frequently visited states
  - Don't forget what was earlier learned
- Small change in policy can have large impact on state distribution - limiting expected policy divergence not enough!
- Think of policy that can go one step left or right in any state

REPS

- Small change in policy can have large impact on state distribution - limiting expected policy divergence not enough!

$$\mathbb{E}_{\mathbf{s}}\left[D_{\mathrm{KL}}(\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}_0)||\pi(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})\right]$$

- So, limit KL from reference to next state-action distribution

$$\mathrm{KL}(\pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})||q(\mathbf{s},\mathbf{a}))$$

- Could allow even larger steps in policy space

REPS

➤ So, limit KL from reference to next state-action distribution

new policy and induced state distribution

$$\begin{aligned}
\max_{\pi,\mu_\pi} \quad & \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{R}_{\mathbf{s}}^{\mathbf{a}} d\mathbf{a} d\mathbf{s}, \\
\text{s. t.} \quad & \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s}) d\mathbf{a} d\mathbf{s} = 1, \\
\forall s'. \quad & \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{P}_{\mathbf{ss}'}^{\mathbf{a}} d\mathbf{a} d\mathbf{s} = \mu_\pi(\mathbf{s}'), \\
& \mathrm{KL}(\pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})||q(\mathbf{s},\mathbf{a})) \leq \epsilon,
\end{aligned}$$

maximize expected reward

normalised distribution

Bellman flow constraint

KL constraint

reference or sampling distribution

[Peters 2010]

REPS

- So, limit KL from reference to next state-action distribution

new policy and induced state distribution

$$\max_{\pi,\mu_\pi} \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{R}_{\mathbf{s}}^{\mathbf{a}}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s},$$

maximize expected reward

$$\text{s. t.} \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} = 1,$$

normalised distribution

$$\forall s'. \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{P}_{\mathbf{s}\mathbf{s}'}^{\mathbf{a}}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} = \mu_\pi(\mathbf{s}'),$$

Bellman flow constraint

$$\mathrm{KL}(\pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})||q(\mathbf{s},\mathbf{a})) \leq \epsilon,$$

KL constraint

reference or sampling distribution

- Solve using Lagrangian optimisation

$$L(p,\eta,V,\lambda) = \iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathrm{s},\mathrm{a})\mathcal{R}_{\mathrm{s}}^{\mathrm{a}}\mathrm{d}\mathrm{a}\mathrm{d}\mathrm{s} + \int_{\mathcal{S}} V(\mathrm{s}')\left(\iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathrm{s},\mathrm{a})\mathcal{P}_{\mathrm{s}\mathrm{s}'}^{\mathrm{a}}\mathrm{d}\mathrm{a}\mathrm{d}\mathrm{s} - \mu_\pi(\mathrm{s}')\right)\mathrm{d}\mathrm{s}'$$
$$+ \lambda\left(1 - \iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathrm{s},\mathrm{a})\mathrm{d}\mathrm{a}\mathrm{d}\mathrm{s}\right) + \eta\left(\epsilon - \iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathrm{s},\mathrm{a})\log\frac{p_\pi(\mathrm{s},\mathrm{a})}{q(\mathrm{s},\mathrm{a})}\mathrm{d}\mathrm{a}\mathrm{d}\mathrm{s}\right).$$

[Peters 2010]

REPS

$$p_\pi(\mathbf{s},\mathbf{a}) = q(\mathbf{s},\mathbf{a}) \exp\left(\frac{\mathcal{R}_\mathbf{s}^\mathbf{a} + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} \mathrm{d}\mathbf{s}' - V(\mathbf{s})}{\eta}\right) \exp\left(\frac{-\lambda - \eta}{\eta}\right)$$

- Lagrangian V looks like a value function! Policy like softmax!
- Now know form of p, but dependent on unknown parameters

[Peters 2010]

REPS

$$p_\pi(\mathbf{s},\mathbf{a}) = q(\mathbf{s},\mathbf{a}) \exp\left(\frac{\mathcal{R}_{\mathrm{s}}^{\mathrm{a}} + \int_{\mathcal{S}} V(\mathrm{s}')\mathcal{P}_{\mathrm{ss}'}^{\mathrm{a}} d\mathrm{s}' - V(\mathrm{s})}{\eta}\right) \exp\left(\frac{-\lambda - \eta}{\eta}\right)$$

- Lagrangian V looks like a value function! Policy like softmax!
- Now know form of p, but dependent on unknown parameters
- Define search space for V, e.g. linear $V(\mathbf{s}) = \boldsymbol{\phi}(\mathbf{s})^T \boldsymbol{\theta}$
- Re-insert in Lagrangian

$$g(\eta, V, \lambda) = \lambda + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[\delta(\mathbf{s},\mathbf{a},V) - \lambda - \eta \log \frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})}\right]$$
$$= \eta\epsilon + \eta \log\left(\mathbb{E}_{q(\mathbf{s},\mathbf{a})} \exp\left(\delta(\mathbf{s},\mathbf{a},V)/\eta\right)\right),$$

- Expectation wrt q can be approximated using samples

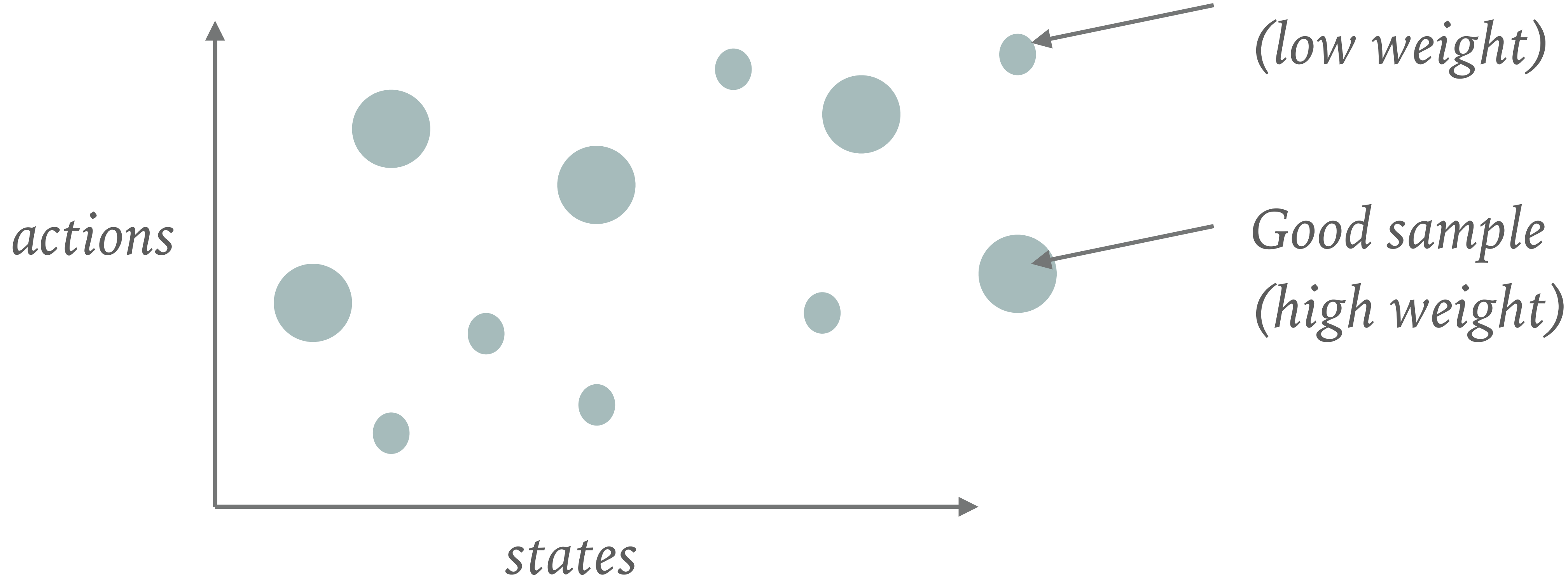
[Peters 2010]

REPRESENTING THE POLICY

- Generally, can't represent the policy in simple form

$$p_\pi(\mathbf{s},\mathbf{a}) = \underbrace{q(\mathbf{s},\mathbf{a})}_{\text{samples from q}} \underbrace{\exp\left(\frac{\mathcal{R}_\mathbf{s}^\mathbf{a} + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} d\mathbf{s}' - V(\mathbf{s})}{\eta}\right)}_{\text{re-weighting factors}} \exp\left(\frac{-\lambda-\eta}{\eta}\right)$$

- Use weighted samples to represent it
- Then, we can fit a stochastic controller

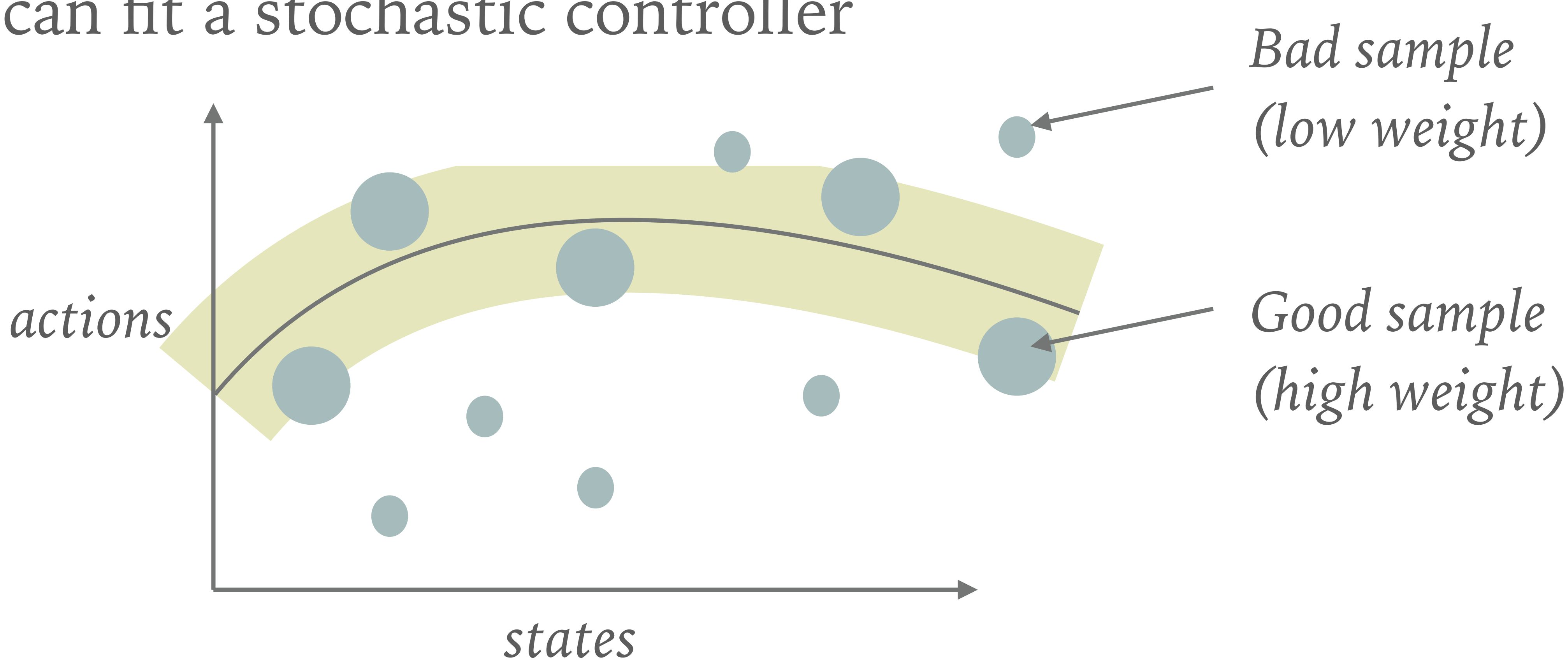


REPRESENTING THE POLICY

- Generally, can't represent the policy in simple form

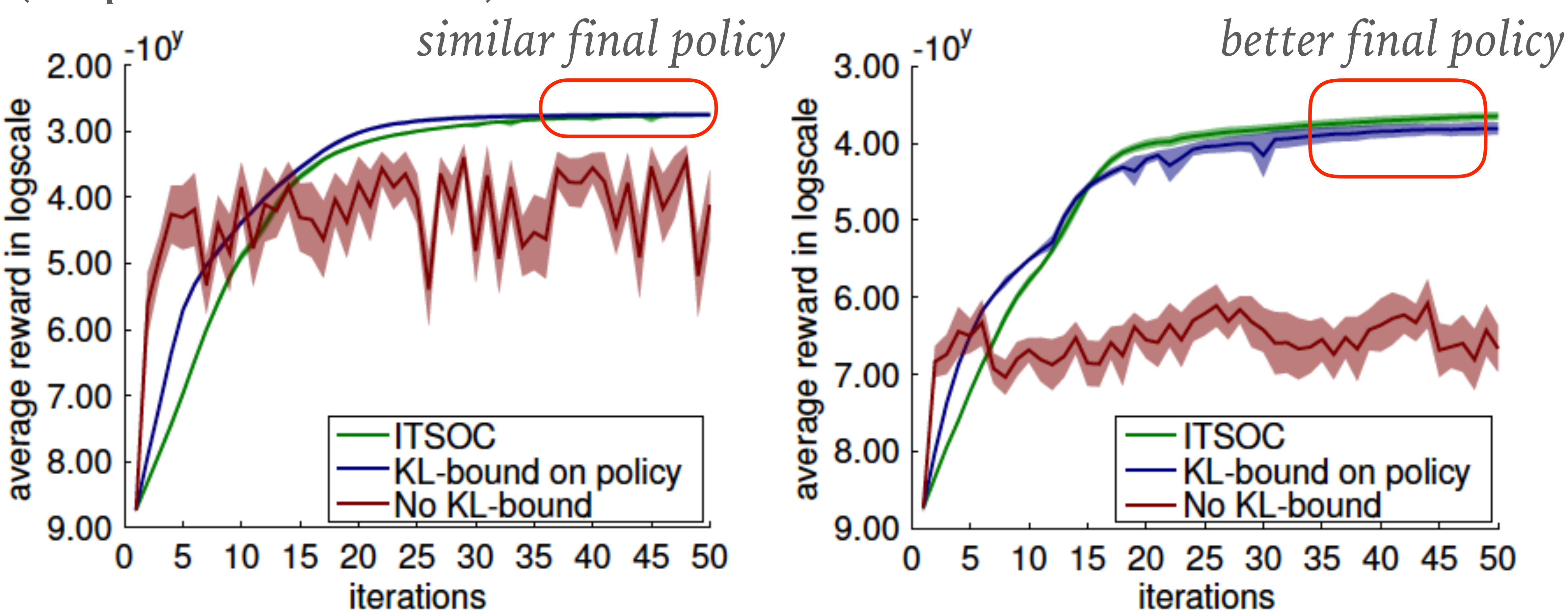
$$p_\pi(\mathbf{s},\mathbf{a}) = \underbrace{q(\mathbf{s},\mathbf{a})}_{\text{samples from q}} \underbrace{\exp\left(\frac{\mathcal{R}_\mathbf{s}^\mathbf{a} + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} d\mathbf{s}' - V(\mathbf{s})}{\eta}\right)}_{\text{re-weighting factors}} \exp\left(\frac{-\lambda-\eta}{\eta}\right)$$

- Use weighted samples to represent it
- Then, we can fit a stochastic controller



REPS

- REPS-style KL bound or TRPO style KL bound (step-based variant)

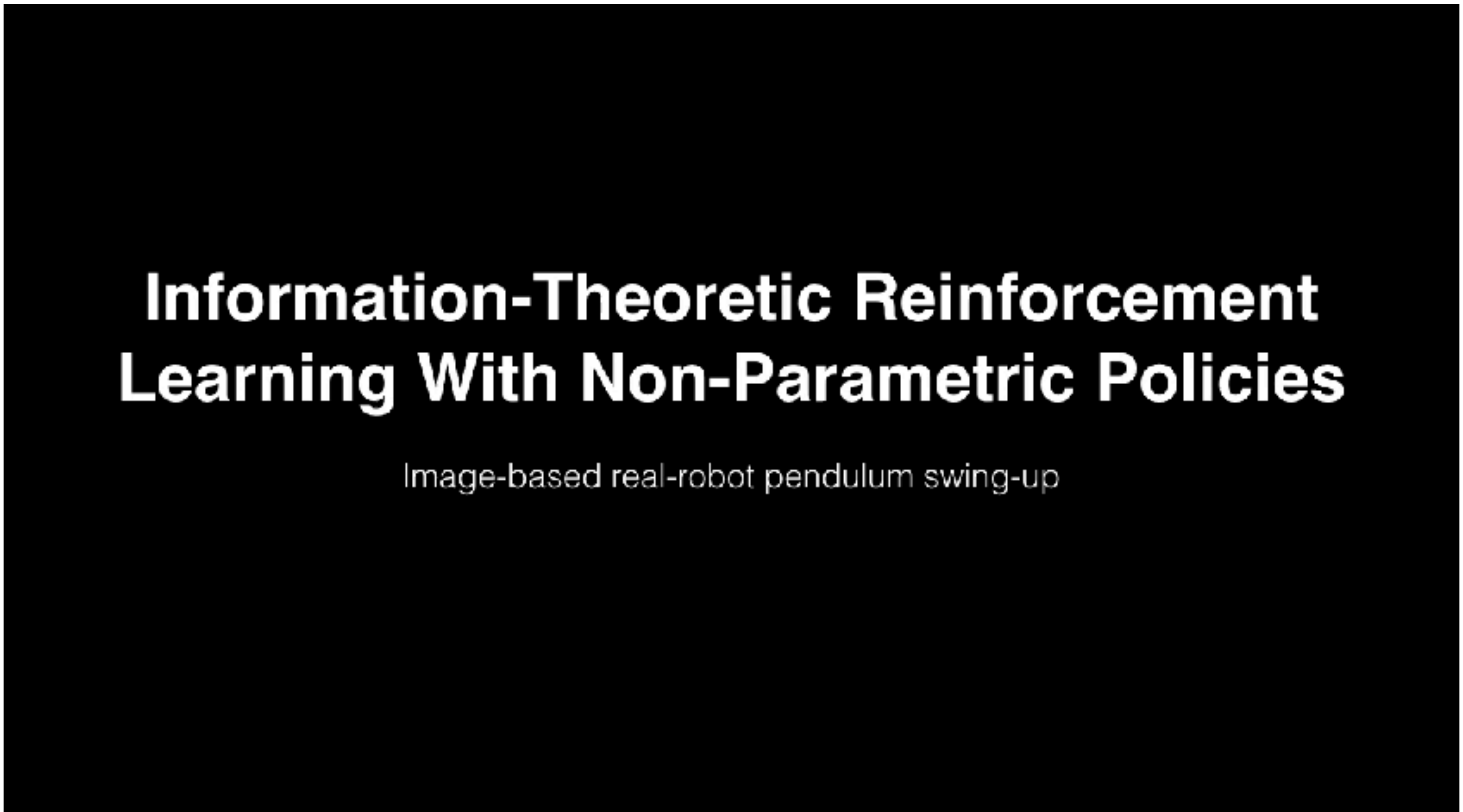


(a) 4-link reaching.No Gravity. $\epsilon = 0.3$ (b) 4-link reaching.With Gravity. $\epsilon = 0.3$

[Lioutikov, 2014]

REPS

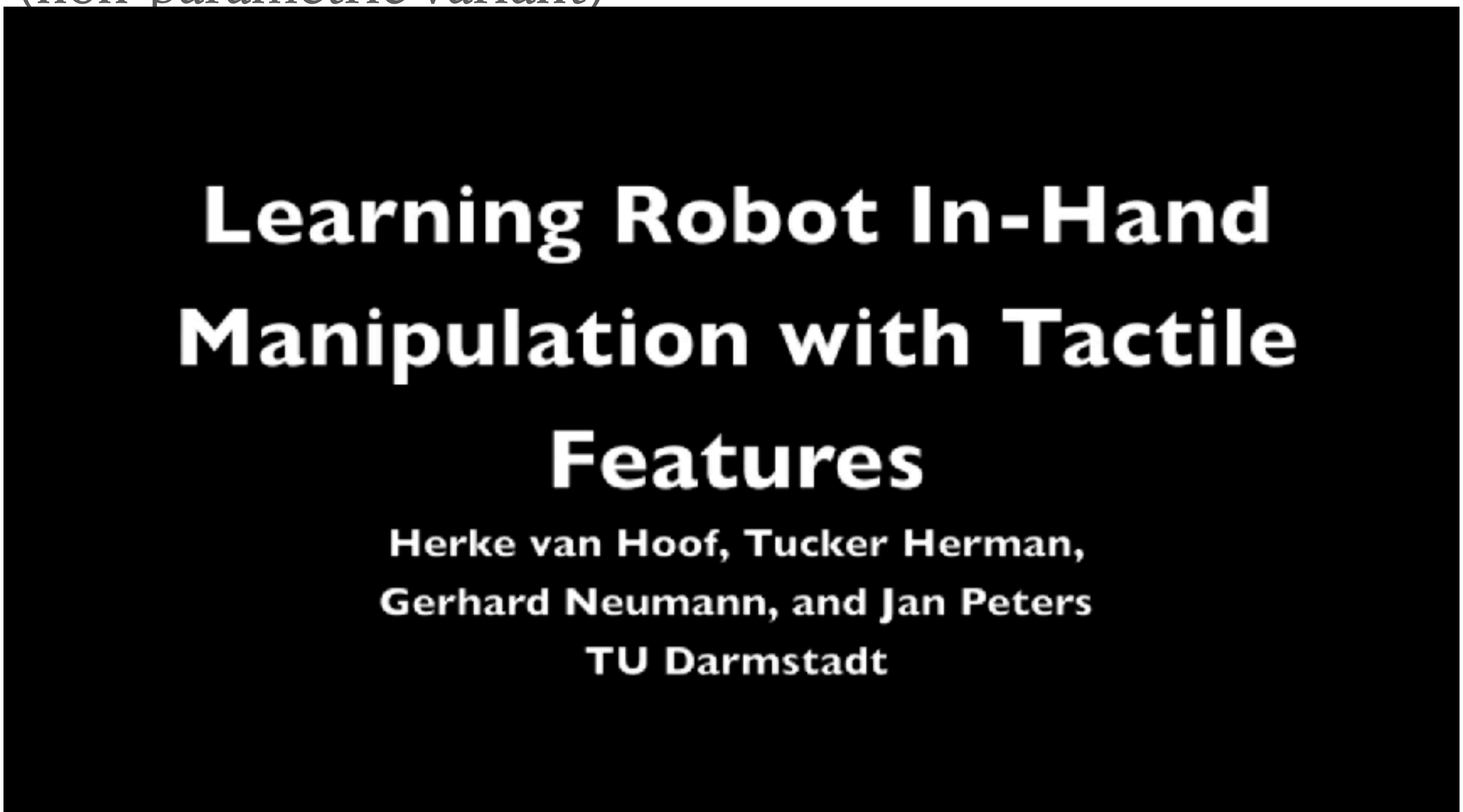
- Learning pendulum swing-up from vision (non-parametric variant)



[van Hoof, 2015]

REPS

- Learning to manipulate (non-parametric variant)



[van Hoof, 2015]

REPS

- Advantages
  - Should be able to take larger steps than TRPO, NPG
  - Consequentially, is relatively data-efficient
  - Variant has optimal regret in adversarial MDPs *[Zimin 2013]*
- Disadvantages
  - Tricky to implement
  - Requires policy approximation step
  - Usually with linear or Gaussian process policies
  - Optimization problem computation intensive

CONCLUSIONS

- Better metric for policy updates: use structure of parameters
- Allows taking larger steps in policy space than e.g. PGT
- NPG, NAC: easy to implement
- TRPO: larger steps (faster), use with neural network
- REPS: even larger steps (?), tricky to implement, linear or Gaussian controllers (for now)

REFERENCES

- [Amari 1998] Amari S.-i., 1998, Natural Gradient Works Efficiently in Learning. Neural Computation, 10, pp. 251–276
- [Bagnell 2003] Bagnell, J.A. and Schneider, J., 2003. Covariant policy search. IJCAI.
- [Kakade 2002] Kakade, S., 2002. A natural policy gradient. Advances in neural information processing systems, 2, pp.1531-1538.
- [Lioutikov 2014] Lioutikov, R., Paraschos, A., Peters, J. and Neumann, G., 2014. Generalizing Movements with Information-Theoretic Stochastic Optimal Control. Journal of Aerospace Information Systems, 11(9), pp.579-595
- [Peters 2008] Peters, J. and Schaal, S., 2008. Natural actor-critic. Neurocomputing, 71(7), pp.1180-1190
- [Peters 2010] Peters, J., Mülling, K. and Altun, Y., 2010, July. Relative Entropy Policy Search. In AAAI (pp. 1607-1612).
- [Schulman 2015] Schulman, J., Levine, S., Abbeel, P., Jordan, M.I. and Moritz, P., 2015. Trust Region Policy Optimization. In ICML (pp. 1889-1897).
- [Schulman 2016] Schulman, J., Moritz, P., Levine, S., Jordan, M.I. and Abbeel, P., 2016. High-Dimensional Control using Generalized Advantage Estimation. In ICLR.
- [Van Hoof 2015] Van Hoof, H., Peters, J. and Neumann, G., 2015. Learning of Non-Parametric Control Policies with High-Dimensional State Features. In AIStats.
- [Zimin 2013] Zimin, A. and Neu, G., 2013. Online learning in episodic Markovian decision processes by relative entropy policy search. In NIPS (pp. 1583-1591).