Online Kernel Selection for Bayesian RL

Joseph Reisinger, Peter Stone and Risto Miikkulainen
The University of Texas at Austin
7/6/08

- **Quick summary**: In Gaussian Process (2) RL (1), the choice of kernel is important for performance.

(3) How can we choose the kernel efficiently online?

Reinforcement Learning

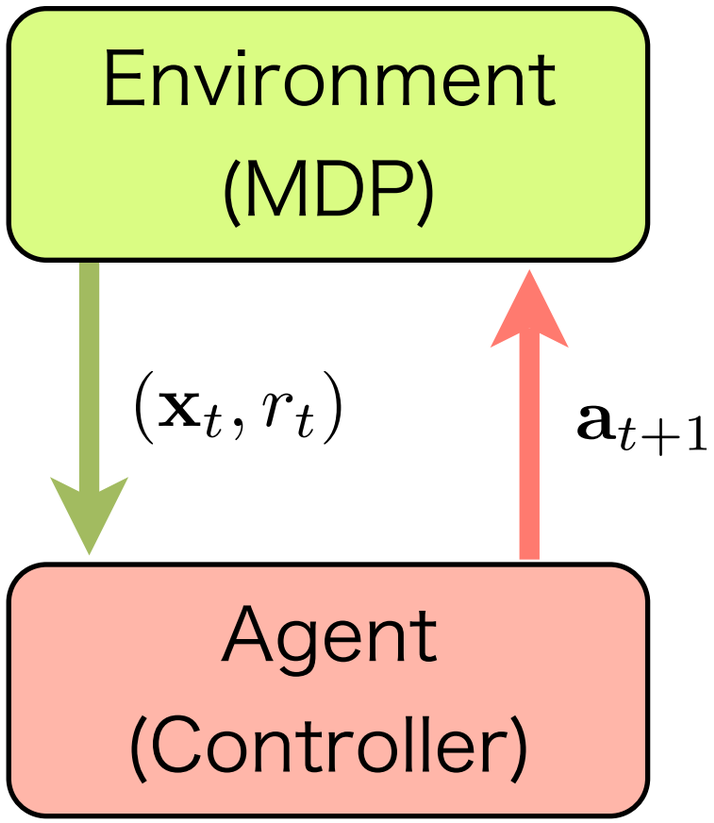
States $\mathbf{x} \in \mathcal{S}$

Actions $\mathbf{a} \in \mathcal{A}$

Reward $r \in \Re$

Transition probabilities

$\mathbf{x}_{t+1} \sim p(\cdot|\mathbf{x}_t, \mathbf{a}_t)$



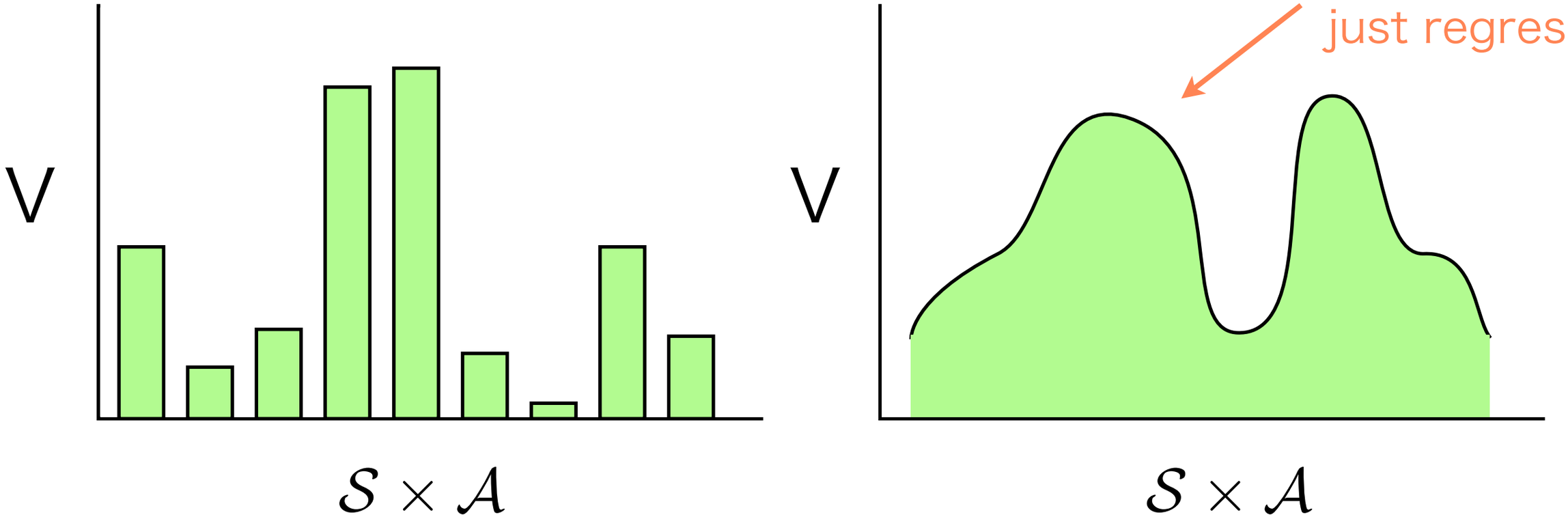
Want to find a policy $\mu : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ that maximizes

$V^{\mu}(\mathbf{x}) = \mathbf{E}_{\mu}[D(\mathbf{x})]$ the "expected discounted return"

$$D(\mathbf{x}) = \sum_{i=0}^{\infty} \gamma^i R(\mathbf{x}_i) | \mathbf{x}_0 = \mathbf{x} \qquad \mathbf{x}_{i+1} \sim p^{\mu}(\cdot|\mathbf{x}_i)$$

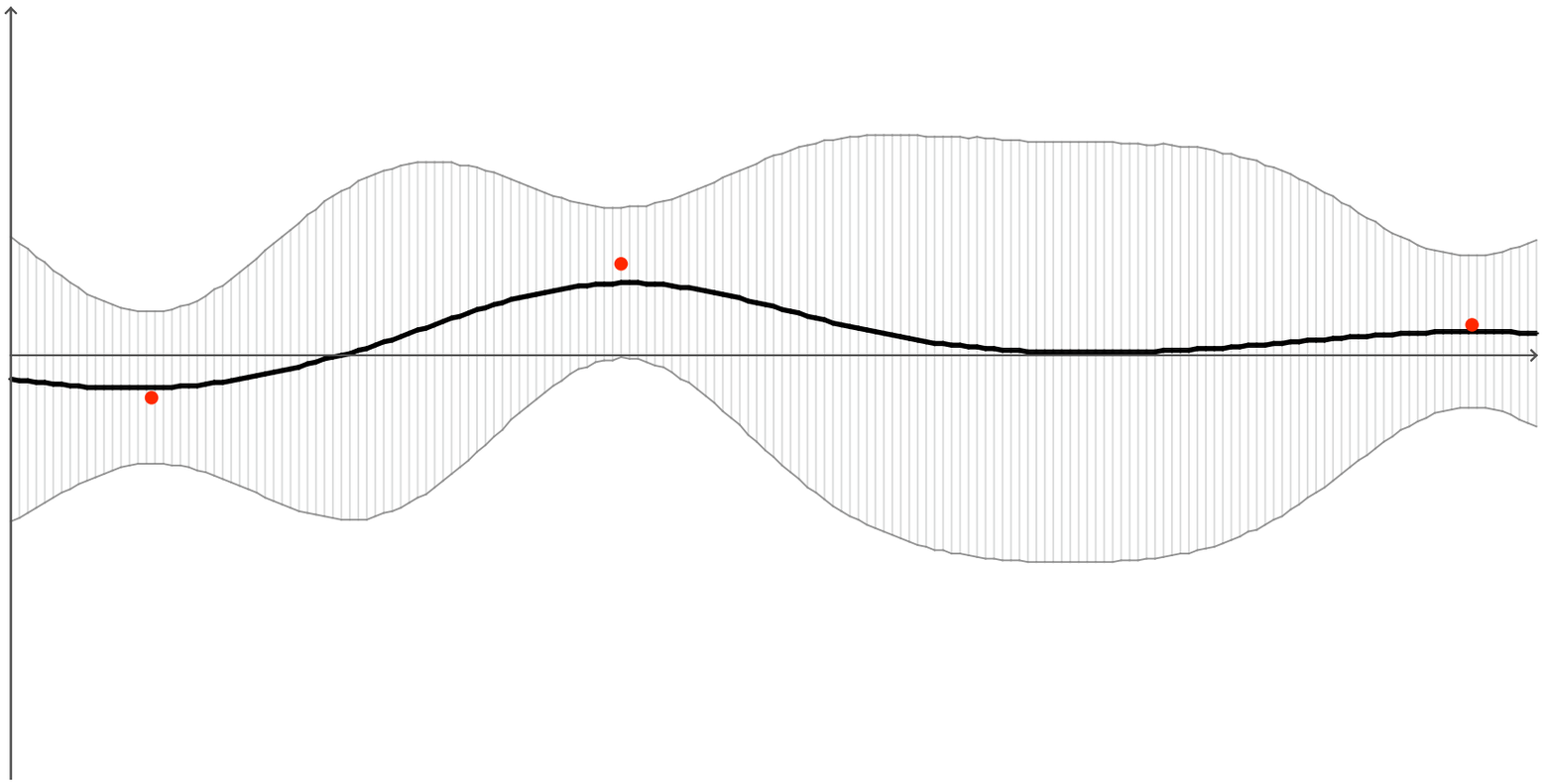
(Sutton and Barto 1998)

Value Function



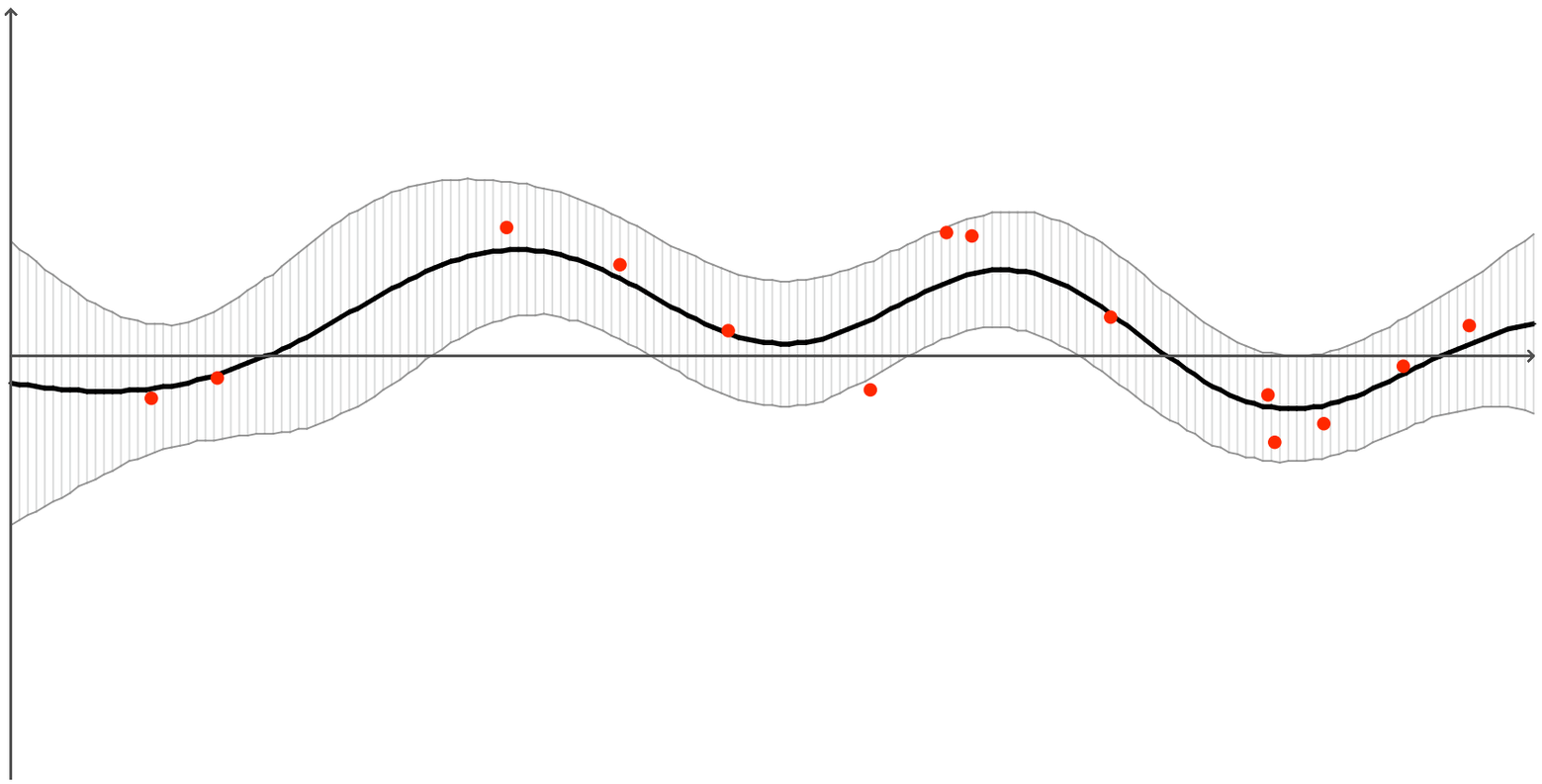
- Can compute a policy from a value function
- How is the value function represented?
- Generalization without “approximation” is possible!
  ↳ e.g. with Gaussian processes

Gaussian Processes



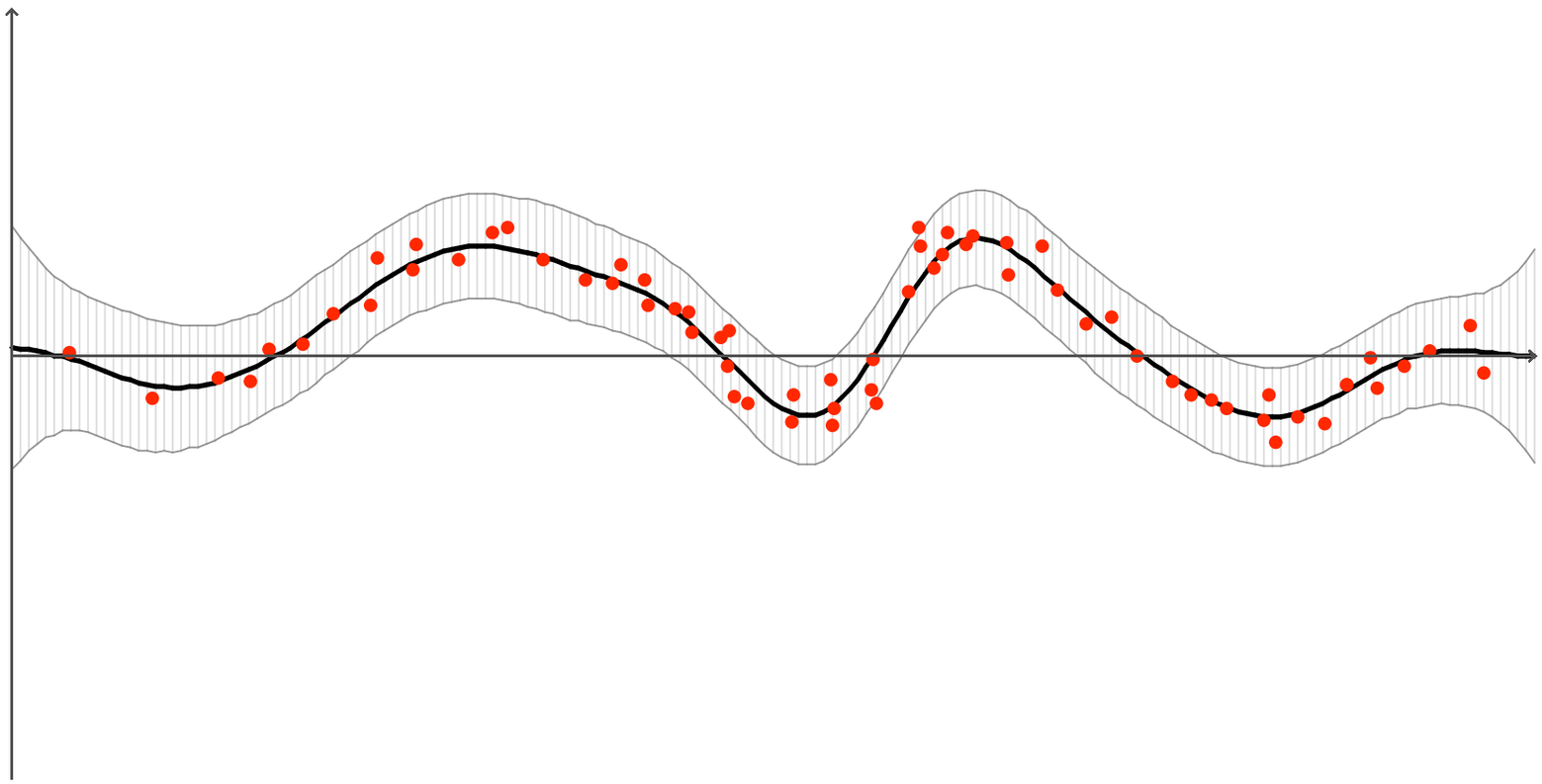
- Don’t make complexity assumptions without seeing the data. Nonparametric inference allows the number of parameters to scale with the size of the data set
- But what about overfitting? Use a Bayesian method: i.e. regularization through the prior

Gaussian Processes



- Don’t make complexity assumptions without seeing the data. Nonparametric inference allows the number of parameters to scale with the size of the data set
- But what about overfitting? Use a Bayesian method: i.e. regularization through the prior

Gaussian Processes



- Don’t make complexity assumptions without seeing the data. Nonparametric inference allows the number of parameters to scale with the size of the data set
- But what about overfitting? Use a Bayesian method: i.e. regularization through the prior

Gaussian Processes

Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=0}^{N}$ be the observed (labeled) data.

Assume that the random variables $\mathbf{y}$ are distributed

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \quad \text{where } [\mathbf{K}]_{ij} \stackrel{\text{def}}{=} k(\mathbf{x}_i, \mathbf{x}_j)$$

prior covariance function (kernel)

Using the data, we can infer an unknown value y* at a test pt x*

$$\begin{bmatrix} \mathbf{y} \\ y^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^\top & \sigma^* \end{bmatrix}\right)$$

where, $\mathbf{k} \stackrel{\text{def}}{=} (k(x_0, x^*), \ldots, k(x_n, x^*))^\top$. This has posterior moments

$$\mathbf{E}[y^*|\mathbf{y}] = \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{y}$$

$$\mathbf{Var}[y^*|\mathbf{y}] = \sigma^* - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}$$

i.e. we can do prediction given the covariance.

(Rasmussen and Williams 2006)

Gaussian Processes

Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=0}^{N}$ be the observed (labeled) data.

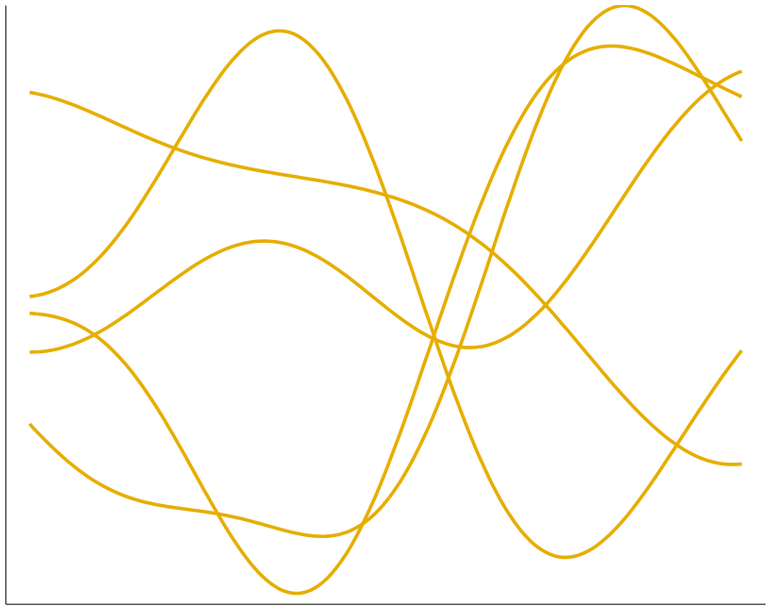
$$[\mathbf{K}]_{ij} \overset{\text{def}}{=} k(\mathbf{x}_i, \mathbf{x}_j)$$

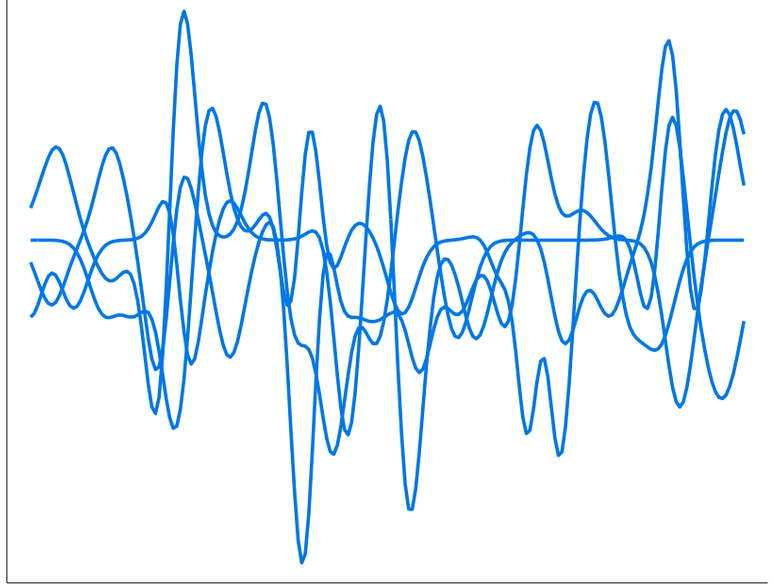
A Gaussian process is completely defined by the data and the kernel

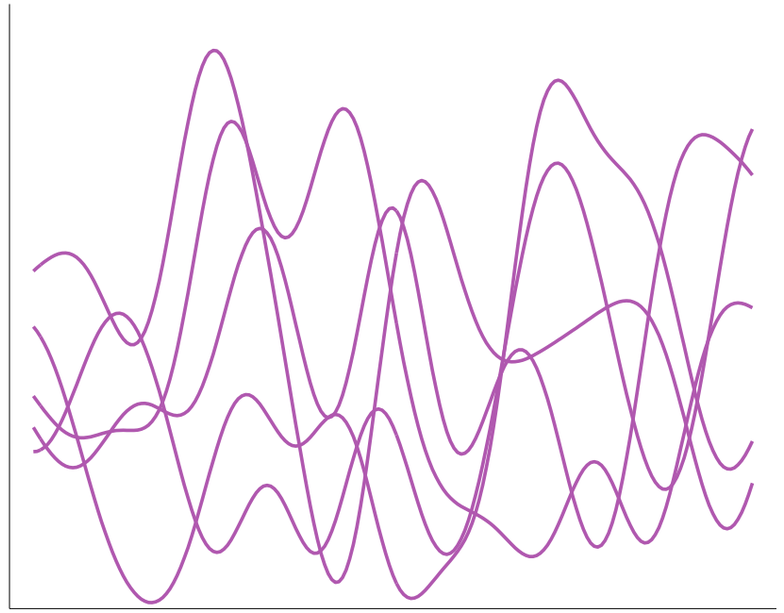
- Straightforward adaptation to RL:
  - value function model
  - sparsification: don't save all of the data
  - O(n) online updates

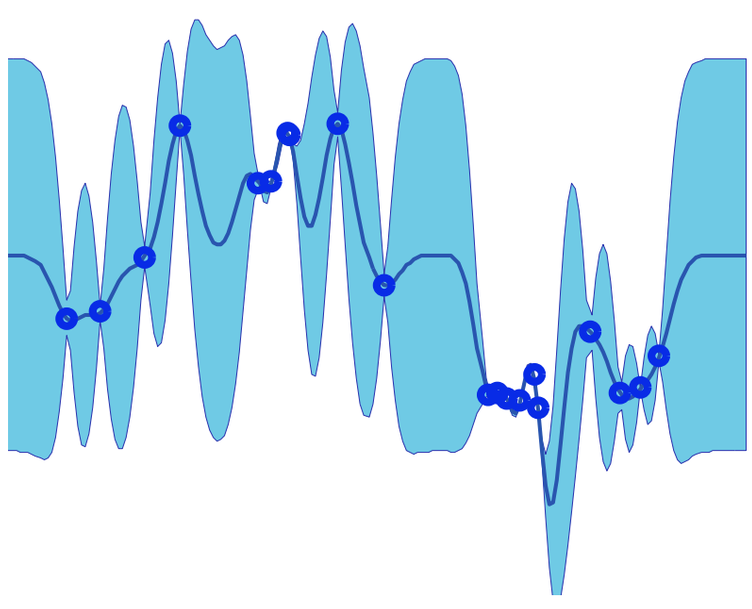
(Engel 2005, Rasmussen and Williams 2006)

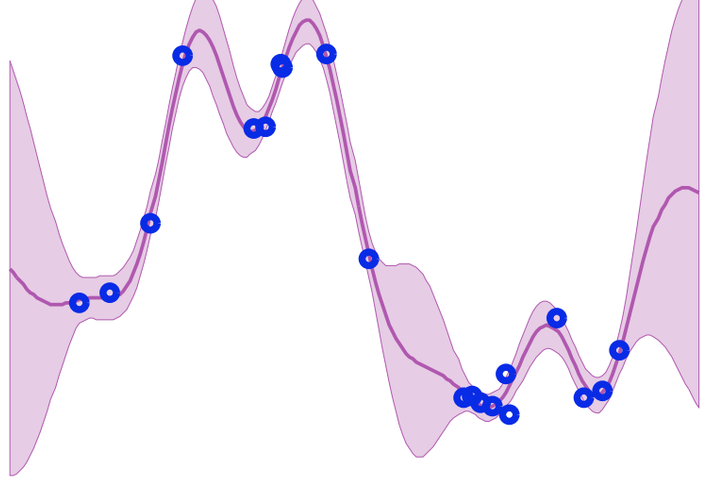
Gaussian Processes





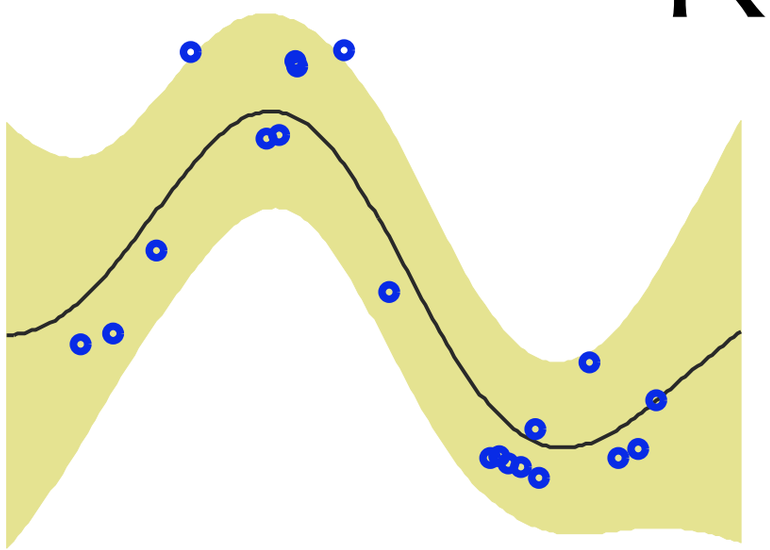


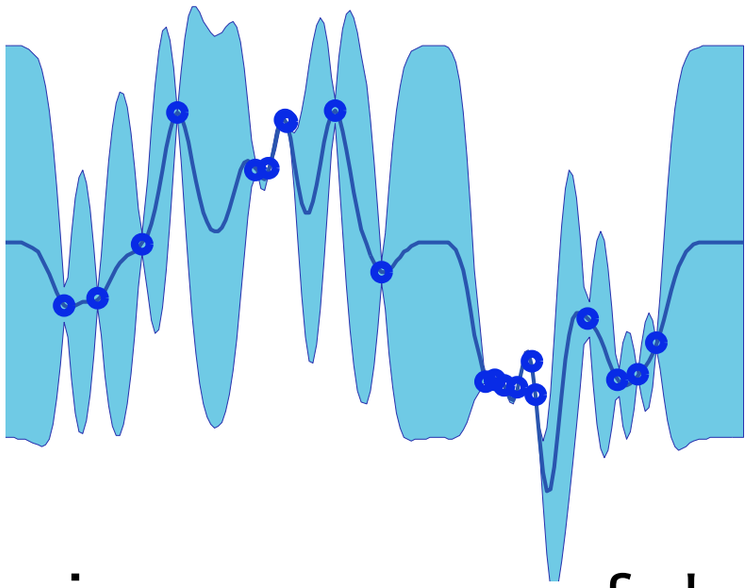


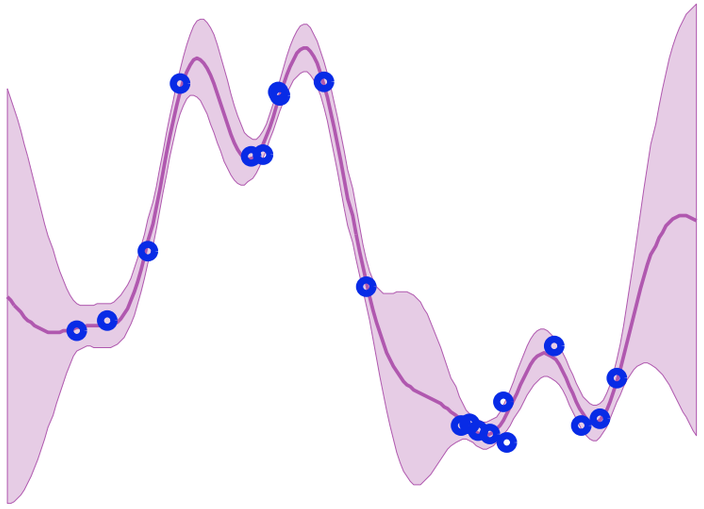


Properties of learned function depends on the choice of kernel (e.g. smoothness)

Kernel Selection





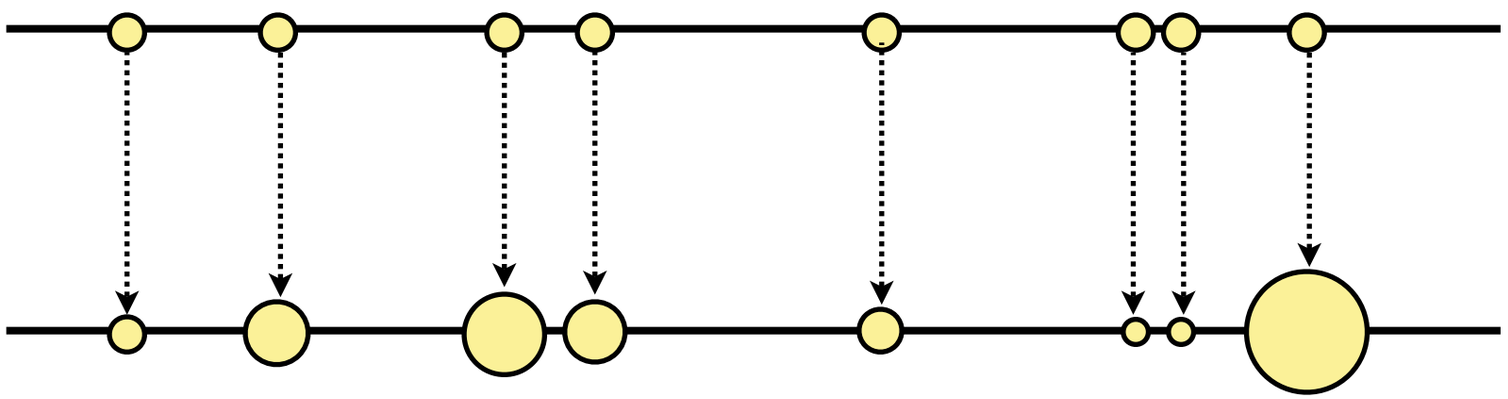


- Kernel notion is very powerful, defining a metric on $\mathcal{S} \times \mathcal{A}$
- How can we choose kernels?
- Typically some model selection step, e.g. cross-validation... also Bayesian model-averaging
- ... but these methods weren't designed to work online

Online Kernel Selection

no restrictions on this

$\boldsymbol{\theta}_i$ kernel instantiation $\boldsymbol{\theta}_i \in \Theta$ the model space



$\{\boldsymbol{\theta}_i^{(0)}\}_{i=1}^n \sim p(\boldsymbol{\theta})$

Calculate $\{w_i^{(t)}\}_{i=1}^n$

$$w_i \stackrel{\text{def}}{=} \frac{p(\mathcal{D}|\boldsymbol{\theta}_i^{(t)})p(\boldsymbol{\theta}_i^{(t)})}{\sum_m p(\mathcal{D}|\boldsymbol{\theta}_m^{(t)})p(\boldsymbol{\theta}_m^{(t)})}$$

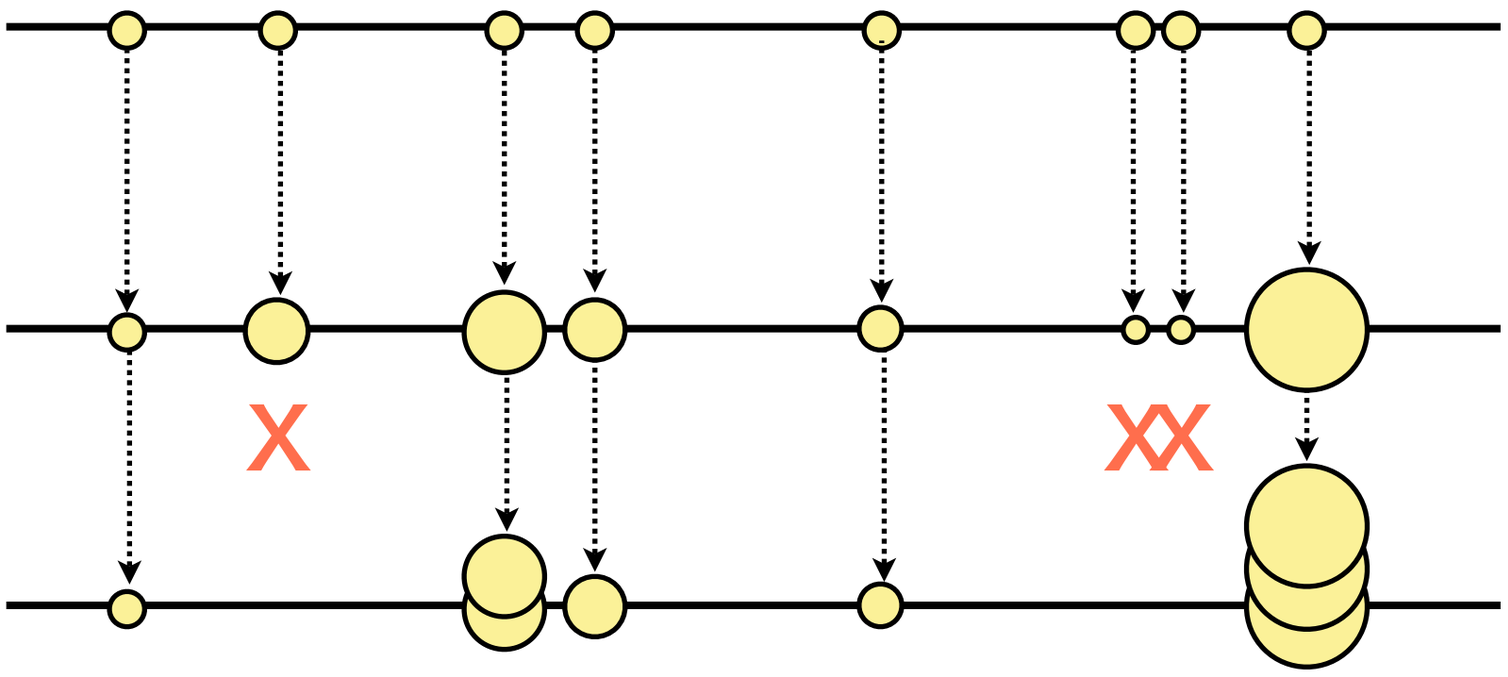
$p(\mathcal{D}|\boldsymbol{\theta}_i^{(t)})$ likelihood of the data given the kernel

simplification: use average reward as a surrogate

Online Kernel Selection

no restrictions on this

$\boldsymbol{\theta}_i$ kernel instantiation $\boldsymbol{\theta}_i \in \Theta$ the model space



$\{\boldsymbol{\theta}_i^{(0)}\}_{i=1}^n \sim p(\boldsymbol{\theta})$

Calculate $\{w_i^{(t)}\}_{i=1}^n$

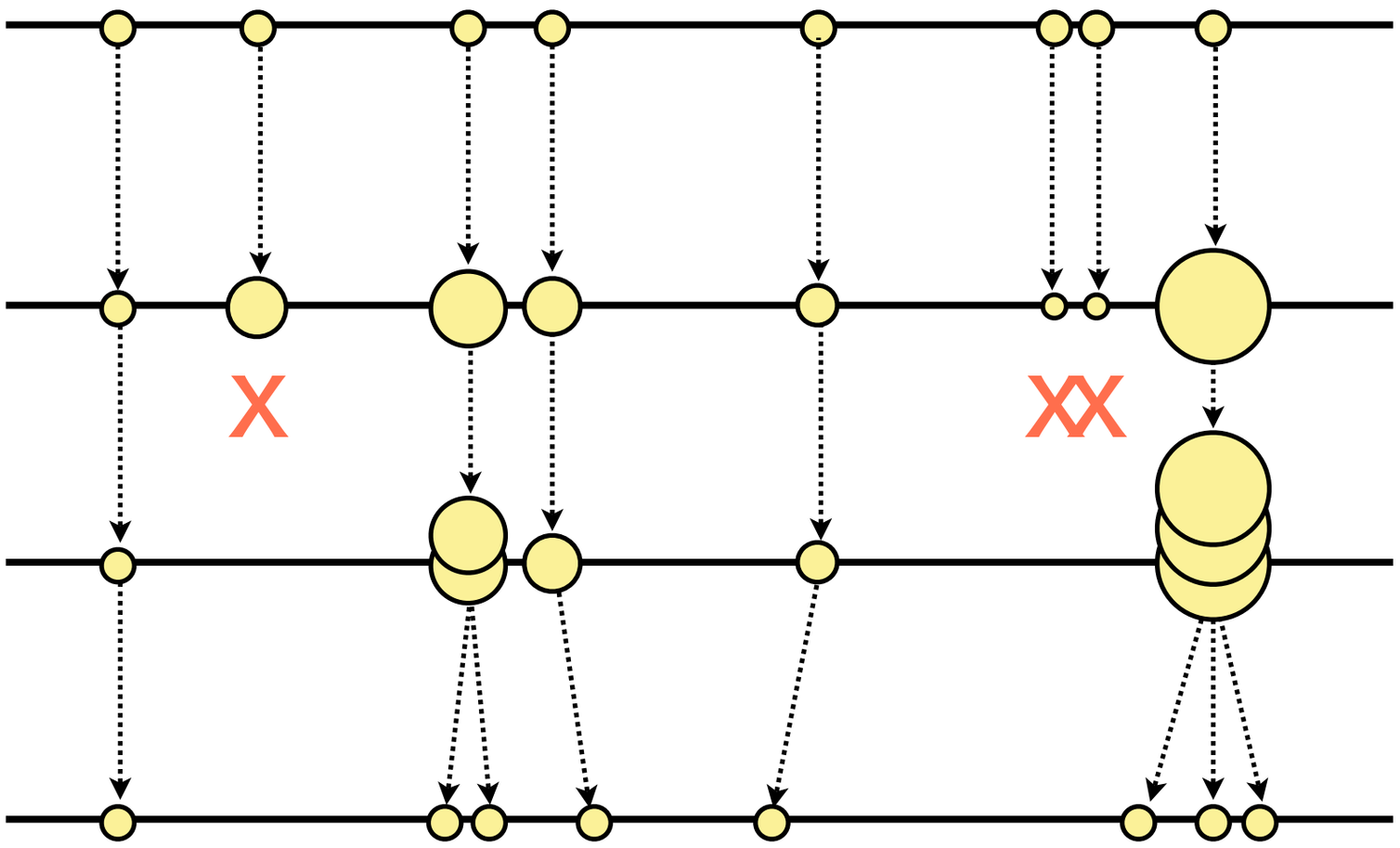
Resample $\{\tilde{\boldsymbol{\theta}}_i^{(t+1)}\}_{i=1}^n$

Dictionary of observations $\mathcal{D}$ can be “inherited” here, unlike e.g., NEAT+Q

Online Kernel Selection

no restrictions on this

$\boldsymbol{\theta}_i$ kernel instantiation $\boldsymbol{\theta}_i \in \Theta$ the model space



$\{\boldsymbol{\theta}_i^{(0)}\}_{i=1}^n \sim p(\boldsymbol{\theta})$

Calculate $\{w_i^{(t)}\}_{i=1}^n$

Resample $\{\tilde{\boldsymbol{\theta}}_i^{(t+1)}\}_{i=1}^n$

Transition kernel

Experimental Setup

- Three methods:
  - GPSARSA - standard model-free GPRL + grid search over kernel parameters
  - RKRL - SMC kernel selection
  - EP-RKRL - RKRL + dictionary of training points is inherited

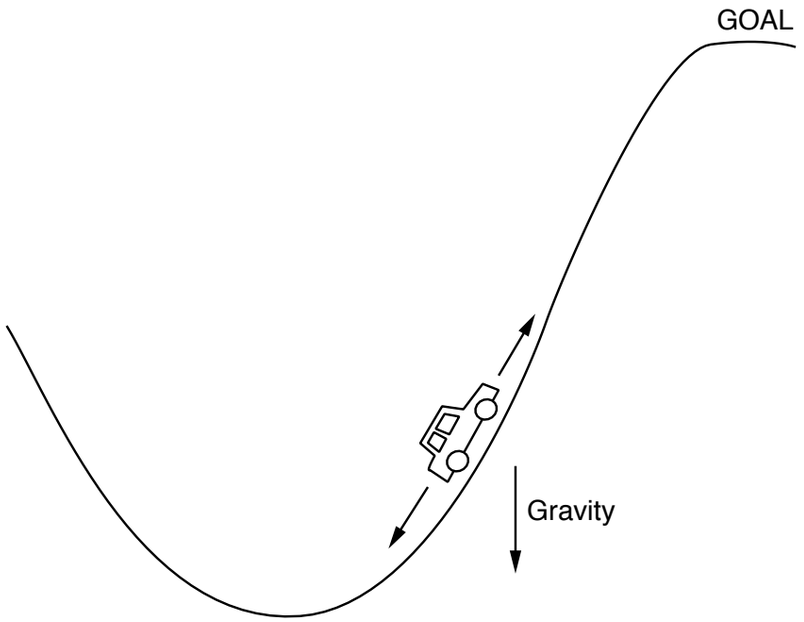
Basic Kernel Example

$$k(\mathbf{x}, \mathbf{x}') = \exp\left[\frac{-||\mathbf{x} - \mathbf{x}'||^2}{\sigma^2}\right]$$

Expanded Kernel Example

$$k(\mathbf{x}, \mathbf{x}') = \exp\left[-\sum_i w_i (x_i - x_i')^2\right]$$

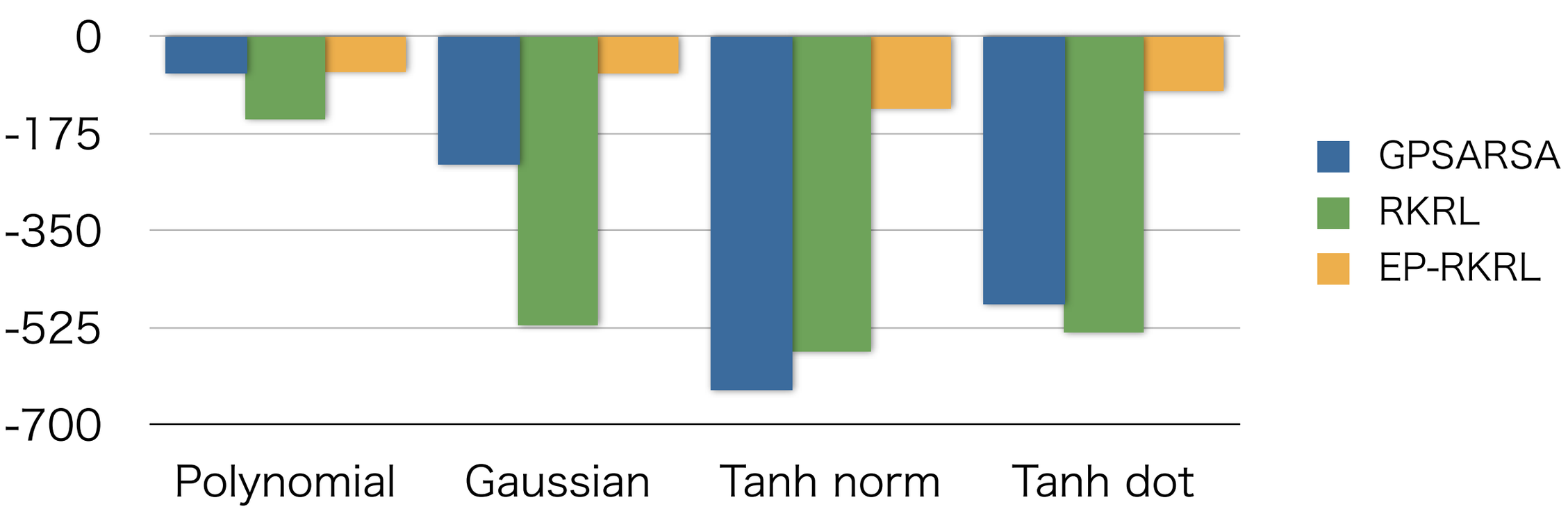
Results: Mountain Car



$$\mathbf{x} = (\dot{x}, x) \in \Re^2$$

$$a \in \{-1, 0, 1\}$$

100 eps/eval

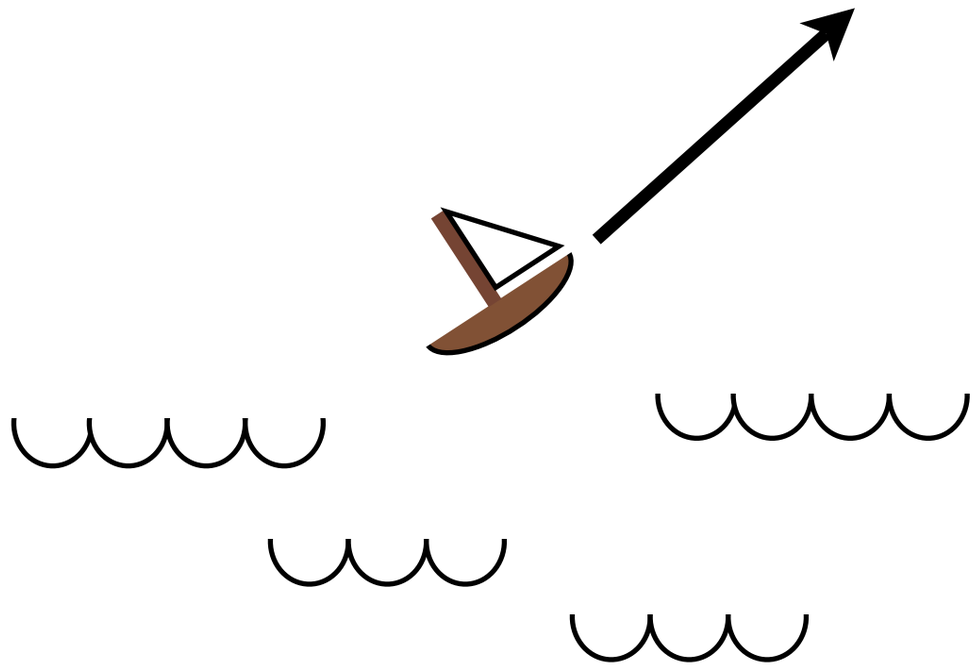


Whiteson (2007): -52

White (2007): -53.92 (± 0.37)

(Figure from Singh and Sutton, 1996)

Results: Sailboat Steering

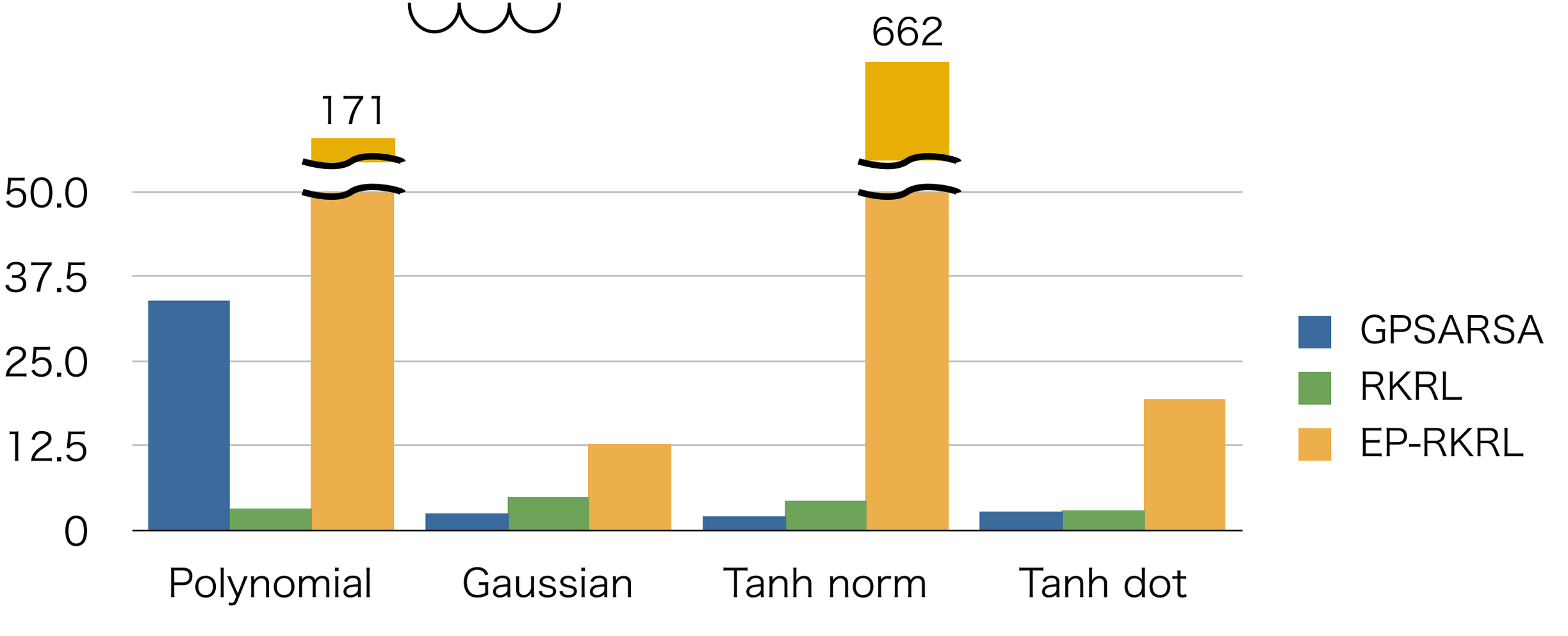


$$\mathbf{x} = (\theta, \dot{\theta}, \dot{x}) \in \Re^3$$

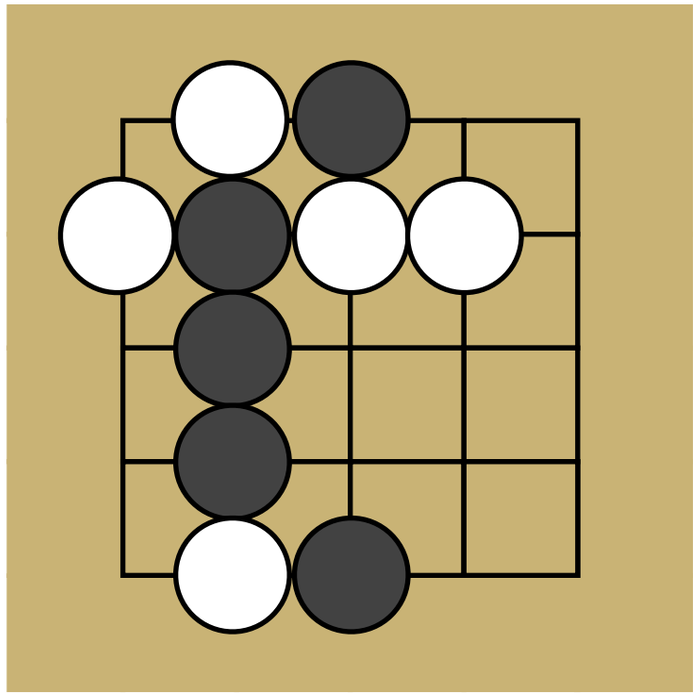
$$a \in [-90, 90] \times [-1, 2] \subset \Re^2$$

discretized actions
(3 degrees, 0.5 thrust)

1000 steps/episode

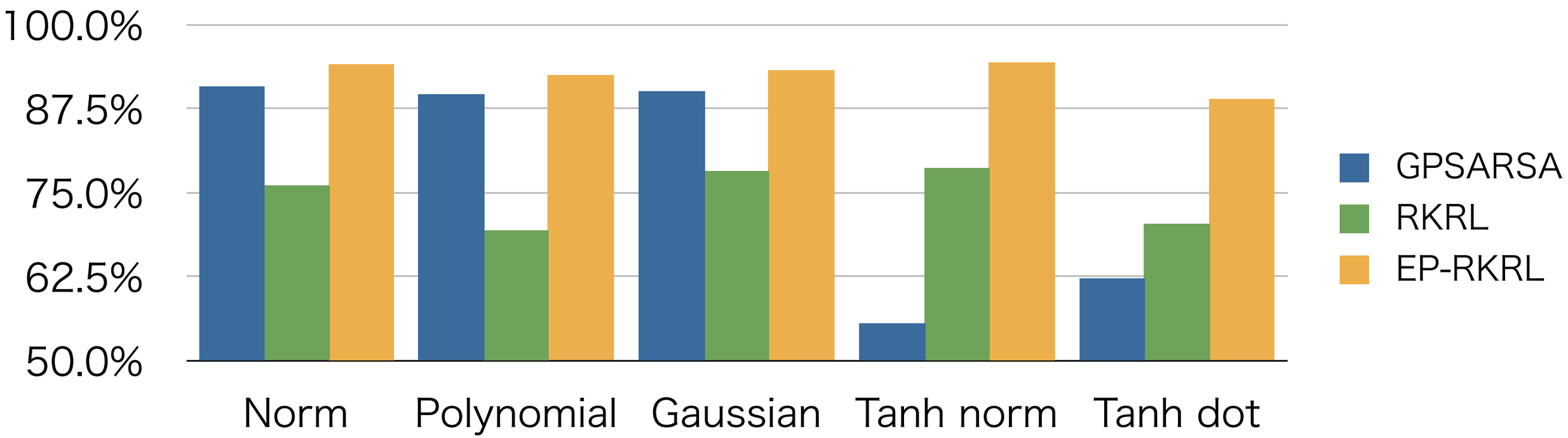


Results: Capture Go



$$\mathbf{x} \in \{-1, 0, 1\}^{25}$$
$$a \in [0, 25]$$

afterstates



Conclusion

- Introduced an online kernel selection procedure: RKRL
- RKRL significantly improves the performance of Gaussian process reinforcement learning
- RKRL is practical online even with many parameters

Thanks!

Questions?

Acknowledgements

- Thanks to Yaak atthew Taylor, Bryan Silvertho l discussions.

- This work was supported by an NSF Graduate Research Fellowship and NSF CAREER award IIS-0237699

Parameters

Basic RL Parameters	GPRL Parameters	RKRL Parameters
$\epsilon = 0.01$	$\sigma = 1.0$	$N = 25$
$\gamma = 1.0$	$\nu = 0.0$	$\mu = 0.01$
		$\tau = 0.5$