# Nonparametric Stochastic Methods for Statistical Learning and Control

Alec Koppel
University of Pennsylvania, Philadelphia, PA

Phd Committee:
Alejandro Ribeiro (Advisor), Vijay Kumar (Chair),
Brian M. Sadler, Jonathan Fink

Phd Defense

Philadelphia, PA, Jun. 30, 2017

# Statistical Learning

- Setting: random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y} \Rightarrow$ training examples $\mathbf{x}_n, y_n$
- Learn to estimate $y_n$ via $\mathbf{x}_n \Rightarrow$ find a statistical model $\hat{y}_n = f(\mathbf{x}_n)$

$\Rightarrow$ predict the price of a commodity (regression)
$\Rightarrow$ identify if a person is present in an image (classification)

# Statistical Learning

- Setting: random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y} \Rightarrow$ training examples $\mathbf{x}_n, y_n$
- Learn to estimate $y_n$ via $\mathbf{x}_n \Rightarrow$ find a statistical model $\hat{y}_n = f(\mathbf{x}_n)$
- How to quantify merit of $\hat{y}_n$? Make minimal no. of mistakes:

$$f^\star := \underset{f \in \mathscr{F}}{\text{argmin}}\, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\mathbb{I}\{f(\mathbf{x}) \neq \mathbf{y})\}]$$

$\Rightarrow$ Clear merit for choosing estimator $\hat{\mathbf{y}}$, which depends on $\mathscr{F}$

- $\mathscr{F}$ is a class of estimators

# Statistical Learning

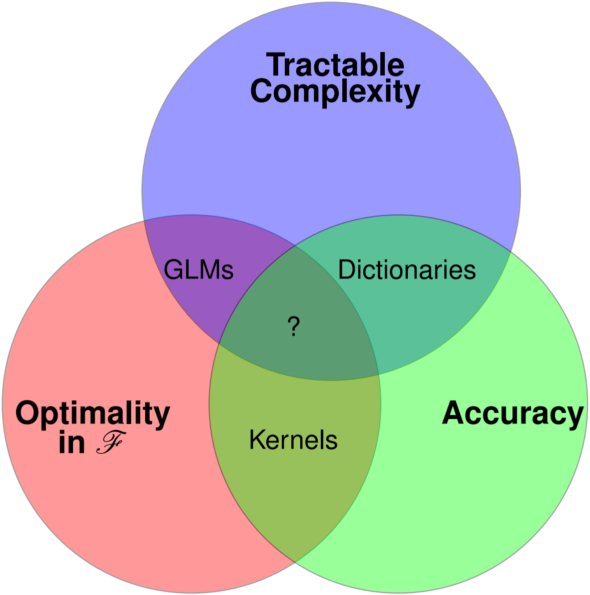
- Setting: random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ $\Rightarrow$ training examples $\mathbf{x}_n, y_n$
- Learn to estimate $y_n$ via $\mathbf{x}_n$ $\Rightarrow$ find a statistical model $\hat{y}_n = f(\mathbf{x}_n)$
- Optimizing indicator intractable $\Rightarrow$ replace by convex $\ell(f(\mathbf{x}), y)$

$$f^* := \underset{f \in \mathscr{F}}{\operatorname{argmin}} \, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), \mathbf{y})] \quad = \frac{1}{N} \sum_{n=1}^{N} \ell(f(\mathbf{x}_n), \mathbf{y}_n)$$

- Focus on instances w/ streaming data $\Rightarrow$ sample size $N$ infinite
- $\mathscr{F}$ $\Rightarrow$ balance accuracy $f^* \approx f^\star$, optimality $f_t \to f^*$, complexity

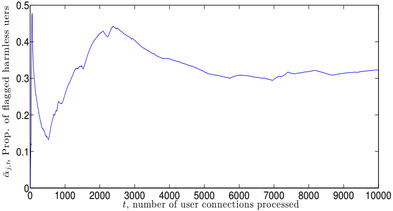
  $\Rightarrow$ Examples: web apps, comms., robotics, smart devices

# On the Choice of $\mathscr{F}$

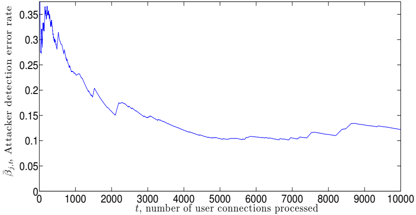


# Generalized Linear Models (GLMs)

- Linear statistical models: $\hat{\mathbf{y}} = \mathbf{w}^T\mathbf{x} \Rightarrow$ param. vector $\mathbf{w} \in \mathscr{F} = \mathbb{R}^p$
  - $\Rightarrow$ translates to vector-valued stochastic **convex** opt.
  - $\Rightarrow$ established multi-agent **optimality** via classic stoch. approx.
- Proposal E.g.: detect attackers in computer networks w/ SVM




(a) Avg. false alarm rate vs. no. of user connections $t$

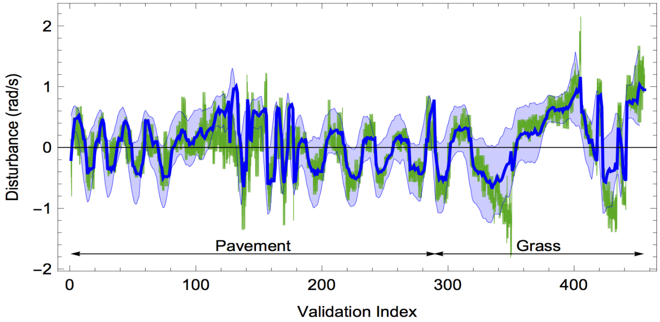



(b) Avg. error rate vs. no. of user connections $t$

- Optimality **does not** automatically translate to statistical accuracy

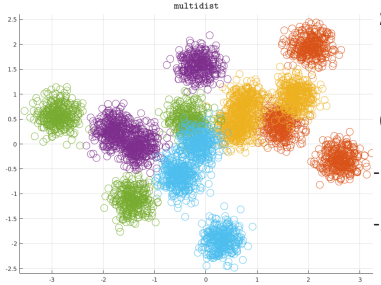
# Dictionaries

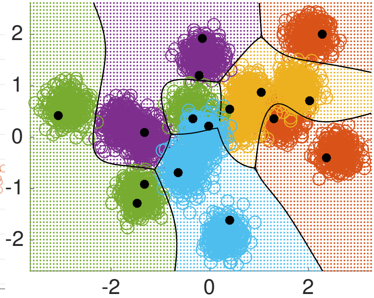
- $\hat{\mathbf{y}} = \mathbf{w}^T \alpha(\mathbf{x}, \mathbf{D})$ extension of GLM w/ learned signal encoding
  $\Rightarrow$ replace $\mathbf{x}$ w/ coding $\alpha(\mathbf{x}, \mathbf{D})$, depends on learned dictionary $\mathbf{D}$
- Actual & predicted robot control uncertainty $\Rightarrow$ closely matches




- Motivated multi-agent extension, convergence tied to stoch. err.
  $\Rightarrow$ Optimality elusive due to **nonconvexity**, "hacking" required

- $\hat{\mathbf{y}} = f(\mathbf{x}) = \sum_{n \in \mathcal{I}} w_n \kappa(\mathbf{x}_n, \mathbf{x})$ $\Rightarrow \kappa$ kernel func., $w_n$ are weights
  $\Rightarrow \mathcal{I}$ is infinite indexing set, corresponds to training examples
- Cvx. prob. in infinite space $\Rightarrow$ optimality, intractable complexity



- This work: compressed kernel function representations
  $\Rightarrow$ Preview: online multi-class kernel SVM on Gaussian mixtures

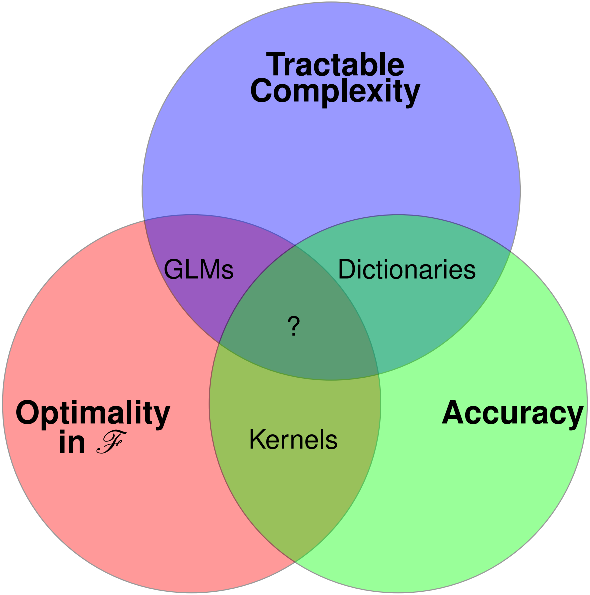
# Optimally Compressed Kernelized Estimates

- Kernel methods: $f(\mathbf{x}) = \sum_{n \in \mathcal{I}} w_n \kappa(\mathbf{x}_n, \mathbf{x}) \Rightarrow$ via Rep. Thm.
  $\Rightarrow \mathcal{I}$ is infinite indexing set $\Rightarrow$ complicated representation
- Maintain convexity, stat. inference via nonlinear interpolator
- Could train with functional stochastic gradient descent
  $\Rightarrow$ Could sparsify solution
- Problem w/ kernel setting: training complexity $\approx$ iteration index
  $\Rightarrow$ Want to sparsify training $\Rightarrow$ possibly invalid descent directions
- This work: convergent online training w/ sparsified kernels
  $\Rightarrow$ accurate, convergent, low complexity statistical learning

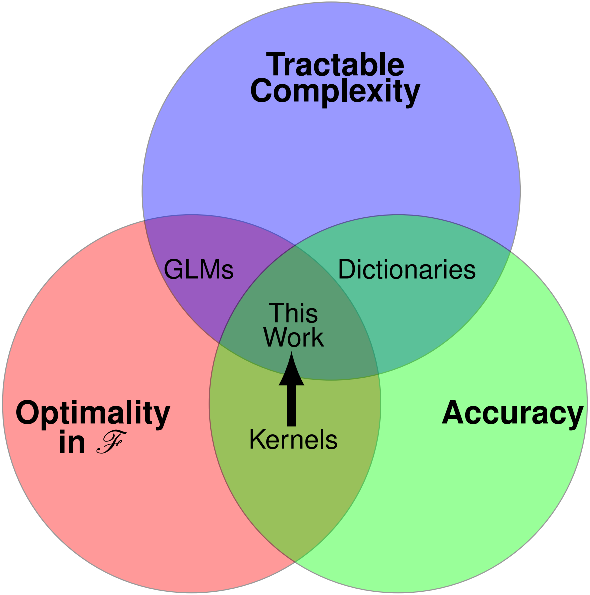
# Optimally Compressed Kernelized Estimates

- Kernel methods: $f(\mathbf{x}) = \sum_{n \in \mathcal{I}} w_n \kappa(\mathbf{x}_n, \mathbf{x}) \Rightarrow$ via Rep. Thm.
  $\Rightarrow \mathcal{I}$ is infinite indexing set $\Rightarrow$ complicated representation
- Maintain convexity, stat. inference via nonlinear interpolator
- Could train with functional stochastic gradient descent
  $\Rightarrow$ Could sparsify solution
- Problem w/ kernel setting: training complexity $\approx$ iteration index
  $\Rightarrow$ Want to sparsify training $\Rightarrow$ possibly invalid descent directions
- This work: convergent online training w/ sparsified kernels
  $\Rightarrow$ accurate, convergent, low complexity statistical learning
- Extend to probs. in reinforcement learning (RL) w/ cont. spaces
  $\Rightarrow$ used to solve Bellman's evaluation equation in full generality
  $\Rightarrow$ foundation upon which many RL methods are developed

# On the Choice of $\mathscr{F}$



# On the Choice of $\mathscr{F}$



# Kernels and Nonparametric Estimation

# Statistical Learning in Kernel Hilbert Space

- Nonlinear statistical models $\Rightarrow$ function estimation: find $f^* \in \mathscr{F}$

$$f^* = \operatorname*{argmin}_{f \in \mathscr{F}} R(f) := \operatorname*{argmin}_{f \in \mathscr{F}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)] + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$$

$\Rightarrow$ expected risk $L(f) := \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), \mathbf{y})]$

- Proposal $\Rightarrow \mathscr{F} = \mathcal{H}$, Reproducing kernel Hilbert space (RKHS)
  $\Rightarrow \mathcal{H}$ is equipped $\mathcal{H}$ w/ *kernel function*, $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that:

$$(i)\ \langle f, \kappa(\mathbf{x}, \cdot)) \rangle_{\mathcal{H}} = f(\mathbf{x})\ , \quad (ii)\ \mathcal{H} = \overline{\text{span}\{\kappa(\mathbf{x}, \cdot)\}} \quad \text{for all } \mathbf{x} \in \mathcal{X}\ .$$

- E.g., Gaussian/RBF: $\kappa(\mathbf{x}, \mathbf{x}') = \exp\{-(\|\mathbf{x} - \mathbf{x}'\|_2^2)/2c^2\}$

# Function Representation

- Consider expected risk min. $\Rightarrow$ Representer Theorem (Reisz):

$$f^* = \underset{f}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}),\mathbf{y})] \text{ takes the form } f(\mathbf{x}) = \sum_{n=1}^{\infty} w_n\, \kappa(\mathbf{x}_n, \mathbf{x})\ .$$

  $\Rightarrow$ $\mathbf{x}_n$ are feature vectors, and $w_n$ is a scalar weight.

  $\Rightarrow$ $f$ is a kernel expansion over (infinite) training set

- Unfortunately, as sample size $N \to \infty$

  $\Rightarrow$ kernel matrix $[\mathbf{K}_{\mathbf{X},\mathbf{X}}]_{m,n} := \kappa(\mathbf{x}_m, \mathbf{x}_n)$ infinite too!

  $\Rightarrow$ $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \cdots]$ $\Rightarrow$ kernel dictionary

  $\Rightarrow$ $\boldsymbol{\kappa}_{\mathbf{X}}(\cdot) = [\kappa(\mathbf{x}_1, \cdot) \ldots \kappa(\mathbf{x}_N, \cdot)]^T$ $\Rightarrow$ empirical kernel map

  $\Rightarrow$ model order $M(= N) \to \infty$ $\Rightarrow$ number of dictionary columns

- We want to learn **close approx. to $f^*$ with low memory**

# Functional Stochastic Gradient Descent

- SGD applied to R(f), given independent training example $(\mathbf{x}_t, \mathbf{y}_t)$:

$$f_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)$$

# Functional Stochastic Gradient Descent

- SGD applied to R(f), given independent training example $(\mathbf{x}_t, \mathbf{y}_t)$:
- Apply chain rule:

$$f_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \frac{\partial \ell(f_t(\mathbf{x}_t), y_t)}{\partial f_t(\mathbf{x}_t)} \frac{\partial f_t(\mathbf{x}_t)}{\partial f_t}(\cdot)$$

- Now, differentiate both sides of reproducing property of kernel:

$$\frac{\partial f_t(\mathbf{x}_t)}{\partial f_t} = \frac{\partial \langle f_t, \kappa(\mathbf{x}_t, \cdot)) \rangle_{\mathcal{H}}}{\partial f_t} = \kappa(\mathbf{x}_t, \cdot)$$

# Functional Stochastic Gradient Descent

- SGD applied to R(f), given independent training example $(\mathbf{x}_t, \mathbf{y}_t)$:

$$f_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \ell'(f(\mathbf{x}_t), y_t) \kappa(\mathbf{x}_t, \cdot)$$

- Newest feature vector $\mathbf{x}_t$ enters kernel dictionary $\mathbf{X}_t$
  $\Rightarrow$ with associated weight $\ell'(f(\mathbf{x}_t), y_t) := \partial \ell(f_t(\mathbf{x}_t), y_t) / \partial f_t(\mathbf{x}_t)$

- FSGD $\Rightarrow$ updates on weights, dictionary (Kivinen & Smola '04)

$$\mathbf{X}_{t+1} = [\mathbf{X}_t, \;\; \mathbf{x}_t], \quad \mathbf{w}_{t+1} = [(1 - \eta_t \lambda)\mathbf{w}_t, \;\; -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)] \,,$$

  $\Rightarrow$ Model order $M_t = t - 1$ grows per step $\Rightarrow$ prohibitively costly

- Induction + Rep. Thm. $\Rightarrow f_t(\mathbf{x}) = \sum_{n=1}^{t-1} w_n \kappa(\mathbf{x}_n, \mathbf{x}) = \mathbf{w}_t^T \boldsymbol{\kappa}_{\mathbf{X}_t}(\mathbf{x})$ .

# Controlling Model Order

- Define vanilla FSGD iterate at step $t+1$

$$\tilde{f}_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t; \mathbf{x}_t, \mathbf{y}_t).$$

$\Rightarrow$ parameterized by dictionary and coefficients

$$\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \;\; \mathbf{x}_t], \qquad \tilde{\mathbf{w}}_{t+1} = [(1 - \eta_t \lambda)\mathbf{w}_t, \;\; -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)] \, .$$

# Controlling Model Order

- Propose compressing $\tilde{f}_{t+1} \Rightarrow$ replace FSGD w/ projected variant:

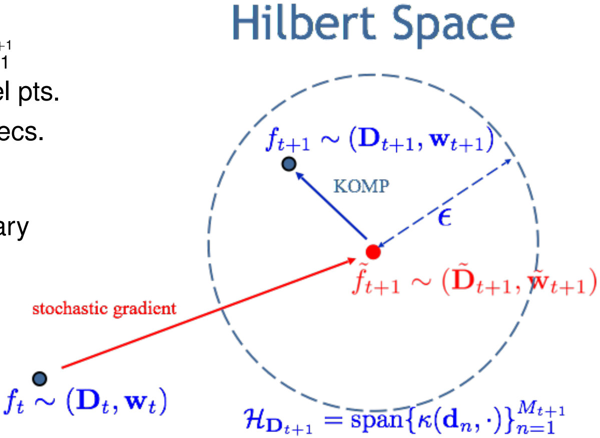
$$f_{t+1} = \underset{f \in \mathcal{H}_{\mathbf{D}_{t+1}}}{\operatorname{argmin}} \left\| f - \Big( (1-\eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big) \right\|_{\mathcal{H}}^2$$
$$:= \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}} \Big[ (1-\eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big].$$

- Define Hilbert subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$
  $\Rightarrow$ $\mathbf{d}_n$ are model points $\Rightarrow$ subset of past feature vectors $\{\mathbf{x}_u\}_{u \leq t}$
- Select $\mathcal{H}_{\mathbf{D}_{t+1}}$ greedily $\Rightarrow$ matching pursuit (Mallat, '93)
  $\Rightarrow$ find dict. pt. w/o which causes minimal Hilbert-norm error
  $\Rightarrow$ remove this model pt., repeat while $\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}} \leq \epsilon_t$ true
- Convex methods impractically assume isometry/incoherence

# Selecting $\mathcal{H}_{\mathbf{D}_{t+1}}$ via Matching Pursuit

$$(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$$

- Fix approx. error $\epsilon_t$
- $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$
- $\{\mathbf{d}_n\} \subset \{\mathbf{x}_u\}_{u \leq t}$ $\Rightarrow$ model pts.
  $\Rightarrow$ subset of past feat. vecs.
- Remove model pts. $\mathbf{d}_n$
  $\Rightarrow$ until hit nbhd. boundary
- Stopping criterion:
  $\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}} \leq \epsilon_t$
- New model order:
  $M_{t+1} \leq M_t + 1$



# Parsimonious Online Learning with Kernels

**Require:** $\{\mathbf{x}_t, \mathbf{y}_t, \eta_t, \epsilon_t\}_{t=0,1,2,\dots}$

**initialize** $f_0(\cdot) = 0, \mathbf{D}_0 = [], \mathbf{w}_0 = []$, i.e. initial dict., coeffs. empty

**for** $t = 0, 1, 2, \dots$ **do**

Obtain independent training pair realization $(\mathbf{x}_t, y_t)$

Compute unconstrained functional stochastic gradient step

$$\tilde{f}_{t+1}(\cdot) = (1 - \eta_t\lambda)f_t - \eta_t\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$$

Revise dictionary $\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \mathbf{x}_t]$,

Revise weights $\tilde{\mathbf{w}}_{t+1} \leftarrow [(1 - \eta_t\lambda)\mathbf{w}_t, \; -\eta_t\ell'(f_t(\mathbf{x}_t), y_t)]$

Compute sparse function approximation via KOMP

$$(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \mathbf{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$$

**end for**

# Convergence Results for POLK

**Theorem**

*The POLK sequence* $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{\textit{KOMP}}\,(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$*, with regularizer* $\eta_t < 1/\lambda$*, initialization* $f_0 = 0$*, and diminishing step-sizes/compression budget*

$$\sum_{t=1}^{\infty} \eta_t = \infty \;, \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty \;, \quad {\color{red}\epsilon_t = \eta_t^2} \;,$$

*achieves null sub-optimality in limit infimum:*

$$\liminf_{t \to \infty} R(f_t) - R(f^*) = 0 \qquad a.s.$$

*Also,* $\{f_t\}$ *converges almost surely to the optimizer* $f^* = \operatorname{argmin}_f R(f)$*:*

$$\lim_{t \to \infty} \|f_t - f^*\|_{\mathcal{H}}^2 = 0 \qquad a.s.$$

- Requires approx. budget $\epsilon_t = \eta_t^2 \;\Rightarrow$ **model grows arbitrarily**

# Convergence Results for POLK

**Theorem**

*The POLK sequence* $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{\textit{KOMP}}\,(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1})$ *run with* $f_0 = 0$*, regularizer* $\lambda > 0$*, constant step-size, compression budget*

$$\eta_t = \eta\ , \quad \epsilon = K\eta^{3/2} = \mathcal{O}(\eta^{3/2})\ , \quad \eta < 1/\lambda\ ,$$

*where* $K > 0$ *is a positive scaler, converges to a nbhd. w.p.1:*

$$\liminf_{t\to\infty} \|f_t - f^*\|_{\mathcal{H}} \leq \frac{\sqrt{\eta}}{\lambda}\left(K + \sqrt{K^2 + \lambda\sigma^2}\right) = \mathcal{O}(\sqrt{\eta}) \qquad a.s.$$

- Bias induced by sparsification asymptotically doesn't hurt too bad
  - $\Rightarrow$ even when **approx. budget doesn't go to null**

# Convergence Results for POLK

**Theorem**

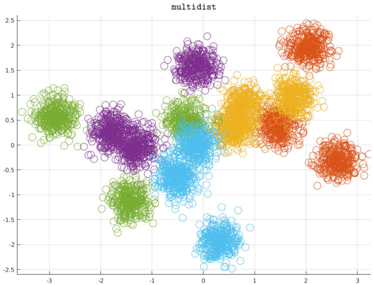
*The POLK sequence $f_t$ with constant step-size $\eta_t = \eta < 1/\lambda$ and approximation budget $\epsilon = K\eta^{3/2}$ where $K > 0$ is a scalar, has finite model order:* $\max M_t \leq M^\infty < \infty$

- Model order of limiting function $f^\infty = \lim_t f_t$ is always finite
- $M^\infty$ depends on $(K\sqrt{\eta})/(C)$

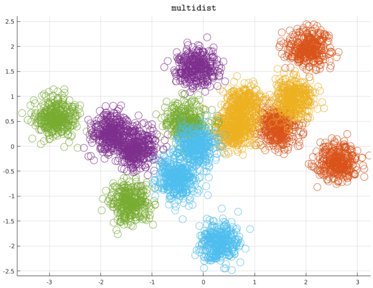
  $\Rightarrow$ KOMP criterion, step-size $\eta$, constant $K$, Lipschitz mod. of $\ell$

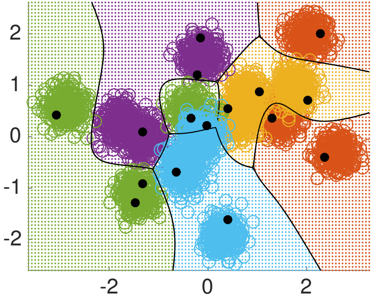
# Gaussian Mixtures Experiments

- Case where training examples for a fixed class
  $\Rightarrow$ drawn from a distinct Gaussian mixture
- 3 Gaussians per mixture, $C = 5$ classes total for this experiment
  $\Rightarrow$ 15 total Gaussians generate data



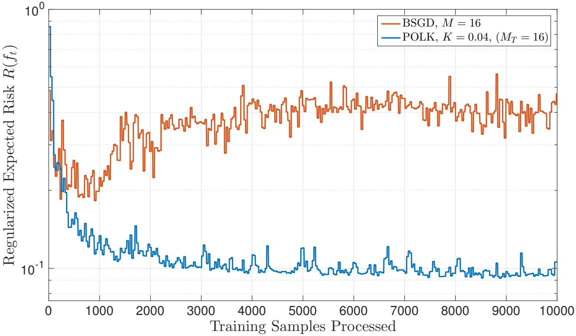
- Case where training examples for a fixed class
  $\Rightarrow$ drawn from a distinct Gaussian mixture
- 3 Gaussians per mixture, $C = 5$ classes total for this experiment
  $\Rightarrow$ 15 total Gaussians generate data



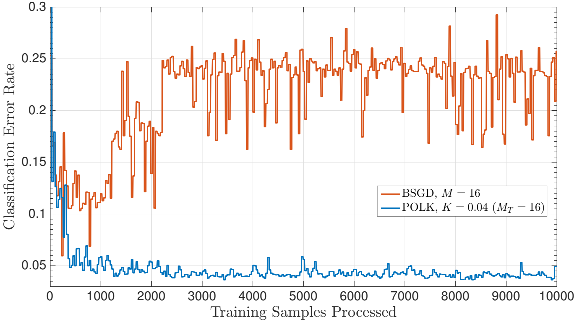

- Grid colors $\Rightarrow$ decision, bold black dots $\Rightarrow$ kernel dict. elements
- Online multi-class kernel SVM achieves $\sim 96\%$ accuracy

# Gaussian Mixtures Experiments



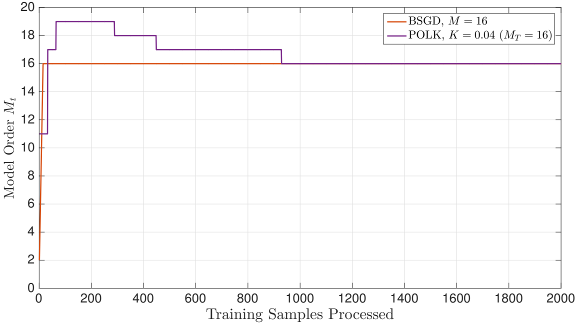

- Comparison with SVM-only competitor

  $\Rightarrow$ fixes model order, not approx. error $\Rightarrow$ set to $M = 16$
- POLK outperforms in terms of regularized risk




- POLK also outperforms in terms of accuracy

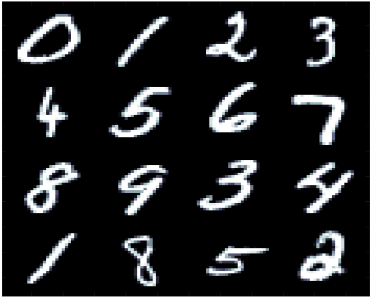
# Gaussian Mixtures Experiments




- POLK *learns* correct model order $M_T = 16$
  $\Rightarrow$ true data domain has 15 modes

# Benchmark Data

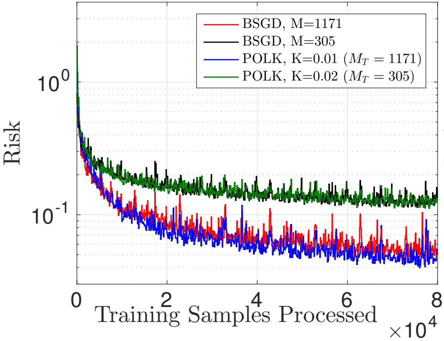
Brodatz textures



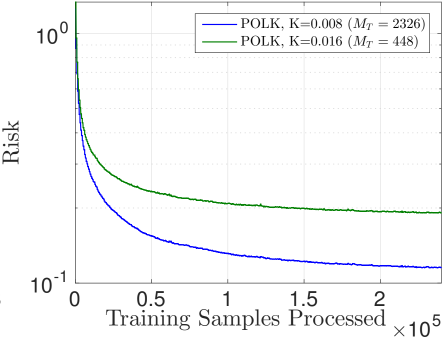
MNIST Digits

- Brodatz: classify texture $\{$roof, grass, etc.$\}$ (13 classes)
- MNIST Digits: classify if digit is $\{0, \ldots, 9\}$ (10 total classes)

# Benchmark Data Experiments



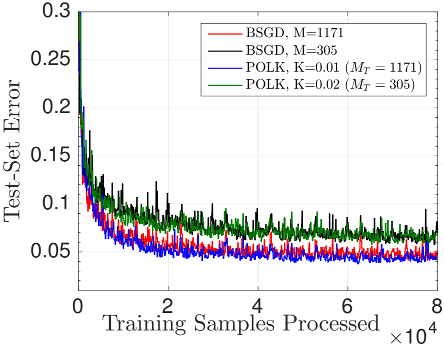

KSVM on Brodatz Textures



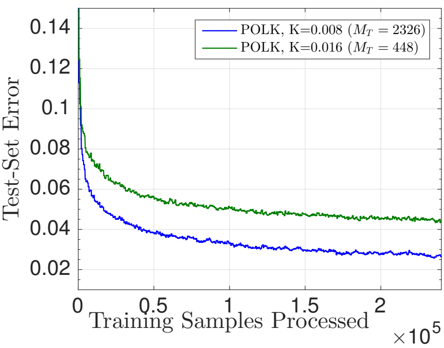

KLR on MNIST Digits

- Objective is stable on real data

# Benchmark Data Experiments



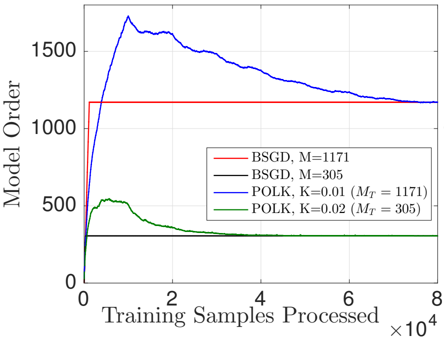

KSVM on Brodatz Textures



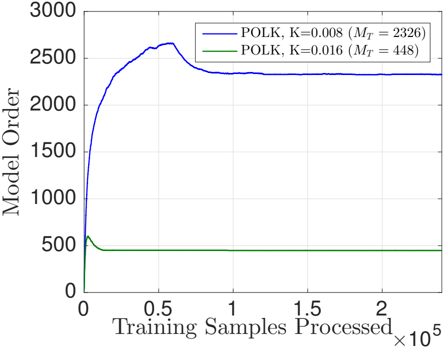

KLR on MNIST Digits

- Stability over descriptive function class $\mathscr{F} = \mathcal{H}$
  - $\Rightarrow$ translates to small error rates: 4.53%, 2.68%
  - $\Rightarrow$ better than SVM-only competitor

# Benchmark Data Experiments



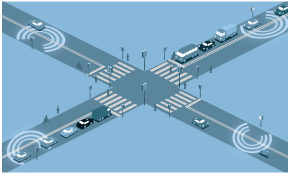

KSVM on Brodatz Textures




KLR on MNIST Digits

- POLK *learns* model order needed for stability
  - $\Rightarrow$ driven by complexity of class-conditional probability density

- Network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
- Node $i$ observes $\{\mathbf{x}_{i,t}, y_{i,t}\}_{t\geq 0}$
  - $\Rightarrow$ wants to learn estimate $\hat{y}_{i,t}$
  - $\Rightarrow$ as good as one w/ global info



- Decentralized nonparametric stochastic program:

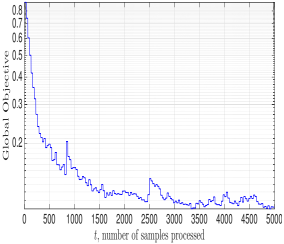
$$f^* = \underset{\{f_i\}_{i\in\mathcal{V}}\subset\mathcal{H}}{\text{argmin}} \sum_{i\in\mathcal{V}} \mathbb{E}_{\mathbf{x}_i, y_i}[\ell_i(f_i(\mathbf{x}_i), y_i)] \quad \text{s.t. } f_i = f_j \text{ for all } (i,j) \in \mathcal{E}\ .$$

- Penalty functional $\psi_c(f)$ $\Rightarrow$ each node applies POLK to $\psi_{i,c}(f_i)$

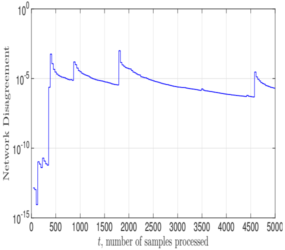
$$\psi_c(f) = \sum_{i\in\mathcal{V}} \left(\mathbb{E}_{\mathbf{x}_i,\mathbf{y}_i}\left[\ell_i(f_i(\mathbf{x}_i), y_i)\right] + \frac{\lambda}{2}\|f_i\|^2_{\mathcal{H}} + \frac{c}{2}\sum_{j\in n_i}\mathbb{E}_{\mathbf{x}_i}\left\{[f_i(\mathbf{x}_i) - f_j(\mathbf{x}_i)]^2\right\}\right)$$

# Results for KSVM on Gaussian Mixtures

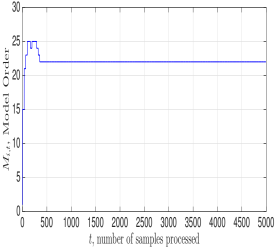
- Penalty initialized $c = 0.01$, doubles every two hundred samples




Global Objective




Network Disagreement




Model Order

- Network Disagreement is $\sum_{(i,j)\in\mathcal{E}} \|f_{i,t} - f_{j,t}\|_{\mathcal{H}}^2$
- POLK in multi-agent setting $\Rightarrow$ 95.7% multi-class accuracy

# Markov Decision Processes

- Agent wants to augment behavior via temporal incentives
  - $\Rightarrow$ starting at state $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^p$, selects action $\mathbf{a}_t \in \mathcal{A} \subset \mathbb{R}^q$
  - $\Rightarrow$ choosing $\mathbf{a}_t$ influences next state $\mathbf{x}_{t+1} \sim \mathbb{P}(\cdot \mid \mathbf{x}_t, \mathbf{a}_t)$
  - $\Rightarrow$ denote $\mathbf{x}_{t+1}$ as $\mathbf{y}_t$ for disambiguation.
- When transitioning to state $\mathbf{y}_t$, a reward $r(\mathbf{x}_t, \mathbf{a}_t, \mathbf{y}_t)$ is assigned
  - $\Rightarrow$ e.g., portfolio revenue, platform stability

# Markov Decision Processes

- Agent wants to augment behavior via temporal incentives
  - $\Rightarrow$ starting at state $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^p$, selects action $\mathbf{a}_t \in \mathcal{A} \subset \mathbb{R}^q$
  - $\Rightarrow$ choosing $\mathbf{a}_t$ influences next state $\mathbf{x}_{t+1} \sim \mathbb{P}(\cdot \mid \mathbf{x}_t, \mathbf{a}_t)$
  - $\Rightarrow$ denote $\mathbf{x}_{t+1}$ as $\mathbf{y}_t$ for disambiguation.
- When transitioning to state $\mathbf{y}_t$, a reward $r(\mathbf{x}_t, \mathbf{a}_t, \mathbf{y}_t)$ is assigned
- This setting is defined by a Markov Decision Process
  - $\Rightarrow$ a quintuple $(\mathcal{X}, \mathcal{A}, \mathbb{P}, r, \gamma)$
  - $\Rightarrow$ $r$ is the reward function, $\gamma \in (0, 1)$ is discount factor
  - $\Rightarrow$ continuous state & action spaces

# The Value Function

- General goal in an MDP $\Rightarrow$ choose actions $\{\mathbf{a}_t\}_{t=1}^{\infty}$
  $\Rightarrow$ maximize reward accumulation when starting at $\mathbf{x}_0 = \mathbf{x}$

$$V(\mathbf{x}, \{\mathbf{a}_t\}_{t=0}^{\infty}) = \mathbb{E}_{\mathbf{y}}\Big[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{x}_t, \mathbf{a}_t, \mathbf{y}_t) \,\Big|\, \mathbf{x}_0 = \mathbf{x}, \{\mathbf{a}_t\}_{t=0}^{\infty}\Big].$$

  $\Rightarrow$ **value function**; $\mathbb{E}$ taken w.r.t. Markov transition density

- Determining sequence $\{\mathbf{a}_t\}$ for continuous $\mathcal{X}$, $\mathcal{A}$
  $\Rightarrow$ has been open for decades (Bellman in 1950s)
- Step towards solution $\Rightarrow$ **evaluate action seq.** $\Rightarrow$ **policy eval.**
  $\Rightarrow$ foundation of determining optimal action sequence

# Policy Evaluation and Bellman's Equation

- Control decisions $\mathbf{a}_t$ $\Rightarrow$ chosen according to a fixed distribution
  $\Rightarrow$ distribution is called a policy $\pi : \mathcal{X} \to \rho(\mathcal{A})$
- Seek to compute *value* of a policy starting from state $\mathbf{x}$,
  $\Rightarrow$ quantified by discounted expected sum of rewards

$$V^{\pi}(\mathbf{x}) = \mathbb{E}_{\mathbf{y}}\Big[\sum_{t=0}^{\infty}\gamma^t r(\mathbf{x}_t, \mathbf{a}_t, \mathbf{y}_t) \,\Big|\, \mathbf{x}_0 = \mathbf{x}, \{\mathbf{a}_t = \pi(\mathbf{x}_t)\}_{t=0}^{\infty}\Big].$$

# Policy Evaluation and Bellman's Equation

- Decomposing value function into its first & subsequent terms
  $\Rightarrow$ yields the **Bellman evaluation equation** (Bellman 1957)

$$V^{\pi}(\mathbf{x}) = \int_{\mathcal{X}} [r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V^{\pi}(\mathbf{y})] \mathbb{P}(d\mathbf{y} \,|\, \mathbf{x}, \pi(\mathbf{x})) \;\text{ for all } \mathbf{x} \in \mathcal{X},$$

- Bellman eval. eqn. defines Bellman operator $\mathscr{B}^{\pi} : \mathcal{B}(\mathcal{X}) \to \mathcal{B}(\mathcal{X})$

$$(\mathscr{B}^{\pi} V)(\mathbf{x}) = \int_{\mathcal{X}} [r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V(\mathbf{y})] \mathbb{P}(d\mathbf{y} \,|\, \mathbf{x}, \pi(\mathbf{x})) \;\text{ for all } \mathbf{x} \in \mathcal{X},$$

- $V^{\pi}(\mathbf{x})$ is fixed pt. of $\mathscr{B}^{\pi}$: $(\mathscr{B}^{\pi} V^{\pi})(\mathbf{x}) = V^{\pi}(\mathbf{x})$ (Bertsekas, '78)
  $\Rightarrow$ our goal is to find $V^{\pi}$ $\Rightarrow$ solve fixed point prob.

# Compositional Stochastic Programming

- Reformulate **Bellman eval. eqn.** as comp. stochastic prog.

  $\Rightarrow$ Subtract $V^\pi(\mathbf{x})$ from both sides, pull inside expectation:

$$0 = \mathbb{E}_{\mathbf{y}}[r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V^\pi(\mathbf{y}) - V^\pi(\mathbf{x}) \,\big|\, \mathbf{x}, \pi(\mathbf{x})] \quad \text{for all } \mathbf{x} \in \mathcal{X} \;.$$

- Square above eqn., then integrate out $\mathbf{x}$, policy $\pi(\mathbf{x})$:

$$V^\pi = \underset{V \in \mathcal{B}(\mathcal{X})}{\operatorname{argmin}} \, \mathbb{E}_{\mathbf{x}, \pi(\mathbf{x})} \Big\{ \frac{1}{2} (\mathbb{E}_{\mathbf{y}}[r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x}) \,\big|\, \mathbf{x}, \pi(\mathbf{x})])^2 \Big\} \;,$$

# Compositional Stochastic Programming

- Reformulate **Bellman eval. eqn.** as comp. stochastic prog.
  $\Rightarrow$ Subtract $V^\pi(\mathbf{x})$ from both sides, pull inside expectation:

$$0 = \mathbb{E}_{\mathbf{y}}[r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V^\pi(\mathbf{y}) - V^\pi(\mathbf{x}) \,\big|\, \mathbf{x}, \pi(\mathbf{x})] \quad \text{for all } \mathbf{x} \in \mathcal{X} \; .$$

- Square above eqn., then integrate out $\mathbf{x}$, policy $\pi(\mathbf{x})$:

$$V^\pi = \underset{V \in \mathcal{B}(\mathcal{X})}{\text{argmin}}\, \mathbb{E}_{\mathbf{x},\pi(\mathbf{x})}\Big\{\frac{1}{2}(\mathbb{E}_{\mathbf{y}}[r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x}) \,\big|\, \mathbf{x}, \pi(\mathbf{x})])^2\Big\} \; ,$$

- Can't search over $\mathcal{B}(\mathcal{X})$ $\Rightarrow$ Hypothesize $\mathcal{B}(\mathcal{X}) \approx \mathcal{H}$, a RKHS
  $\Rightarrow$ Unrestrictive for universal kernel (Micchelli '06), (Gaussian)

$$V^* = \underset{V \in \mathcal{H}}{\text{argmin}}\, \mathbb{E}_{\mathbf{x},\pi(\mathbf{x})}\Big\{\frac{1}{2}(\mathbb{E}_{\mathbf{y}}[r(\mathbf{x}, \pi(\mathbf{x}), \mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x}) \,\big|\, \mathbf{x}, \pi(\mathbf{x})])^2\Big\} + \frac{\lambda}{2}\|V\|_{\mathcal{H}}^2$$

$\Rightarrow$ Define $J(V) = L(V) + (\lambda/2)\|V\|_{\mathcal{H}}^2$, $L(V)$ is compositional term

# Stochastic Gradient Bias Problem

- Differentiate $L(V)$ w.r.t. $V$:

$$\nabla_V L(V) = \mathbb{E}_{\mathbf{x},\pi(\mathbf{x})}\{\mathbb{E}_{\mathbf{y}}[\gamma\kappa(\mathbf{y},\cdot) - \kappa(\mathbf{x},\cdot) \,|\, \mathbf{x},\pi(\mathbf{x})]\mathbb{E}_{\mathbf{y}}[r(\mathbf{x},\pi(\mathbf{x}),\mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x}) \,|\, \mathbf{x},\pi(\mathbf{x})]\}$$

$\Rightarrow$ derivative inside $\mathbb{E}$ + chain rule + reproducing property

- Stochastic descent in $\mathcal{H}$ requires stoch. estimate of above grad.

$$\nabla_V J(V,\delta;\mathbf{x},\pi(\mathbf{x}),\mathbf{y}) = [\gamma\kappa(\mathbf{y},\cdot) - \kappa(\mathbf{x},\cdot)][r(\mathbf{x},\pi(\mathbf{x}),\mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x})] + \lambda V$$

$\Rightarrow \delta := r(\mathbf{x},\pi(\mathbf{x}),\mathbf{y}) + \gamma V(\mathbf{y}) - V(\mathbf{x})$ $\Rightarrow$ temporal difference

- Stoch. grad. biased w.r.t. $\nabla_V J(V)$ due to **correlated** terms
- *Coupled* descent: estimate both terms in product-of-expectations
- Construct total mean of $[\gamma\kappa(\mathbf{y},\cdot) - \kappa(\mathbf{x},\cdot)]$? $\Rightarrow$ infinite complexity

$\Rightarrow$ Build up expectation of scalar temporal difference $\delta$

# Functional Stochastic Quasi-Gradient Method

- Define a scalar fixed pt. recursion $z_t$ to estimate average TD $\bar{\delta}$

$$\delta_t = r(\mathbf{x}_t, \pi(\mathbf{x}_t), \mathbf{y}_t) + \gamma V_t(\mathbf{y}_t) - V_t(\mathbf{x}_t)\ , \quad z_{t+1} = (1-\beta_t) z_t + \beta_t \delta_t$$

$\Rightarrow \delta_t \ \Rightarrow$ temporal difference; $\beta_t \in (0,1) \ \Rightarrow$ step-size.

- Stoch. descent step: replace 1st term in expectation w/ estimate
  $\Rightarrow [\gamma\kappa(\mathbf{y}_t, \cdot) - \kappa(\mathbf{x}_t, \cdot)]$, evaluated at triple $(\mathbf{x}_t, \pi(\mathbf{x}_t), \mathbf{y}_t)$
  $\Rightarrow$ replace $\delta_t$ by $z_{t+1} \ \Rightarrow$ stoch. quasi-gradient (Ermoliev '83)

$$\hat{V}_{t+1} = (1-\alpha_t\lambda)\hat{V}_t - \alpha_t(\gamma\kappa(\mathbf{y}_t, \cdot) - \kappa(\mathbf{x}_t, \cdot)) z_{t+1}$$

  $\Rightarrow \alpha_t$ is a second step-size
- Extends gradient temporal diff. (Sutton '09) to infinite MDPs

# RKHS Parameterization

- If $V_0 = 0 \in \mathcal{H}$, inductively applying Representer Thm. yields

$$\hat{V}_t(\mathbf{x}) = \sum_{n=1}^{2(t-1)} w_n \kappa(\mathbf{v}_n, \mathbf{x}) = \mathbf{w}_t^T \boldsymbol{\kappa}_{\mathbf{X}_t}(\mathbf{x}) \ .$$

$\Rightarrow$ define $\mathbf{v}_n = \mathbf{x}_n$ for $n$ even, $\mathbf{v}_n = \mathbf{y}_n$ for $n$ odd

$$\mathbf{w}_t = [w_1, \cdots, w_{2(t-1)}] \in \mathbb{R}^{2(t-1)} \ ,$$
$$\mathbf{X}_t = [\mathbf{x}_1, \mathbf{y}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{y}_{t-1}] \in \mathbb{R}^{p \times 2(t-1)} \ .$$

- Kernel expansion + together with FSQG $\Rightarrow$ parametric updates:

$$\mathbf{X}_{t+1} = [\mathbf{X}_t, \ \ \mathbf{x}_t, \mathbf{y}_t], \quad \mathbf{w}_{t+1} = [(1 - \alpha_t \lambda)\mathbf{w}_t, \quad \alpha_t z_{t+1}, -\alpha_t \gamma z_{t+1}] \ ,$$

- Of course, same complexity issue as FSGD in RKHS: $M_t = \mathcal{O}(t)$

  $\Rightarrow$ but can solve this w/ sparse projections of POLK!

**Require:** $\{\mathbf{x}_t, \pi(\mathbf{x}_t), \mathbf{y}_t, \alpha_t, \beta_t, \epsilon_t\}_{t=0,1,2,\ldots}$
**initialize** $V_0(\cdot) = 0, \mathbf{D}_0 = [], \mathbf{w}_0 = [], z_0 = 0$
**for** $t = 0, 1, 2, \ldots$ **do**
Obtain trajectory realization $(\mathbf{x}_t, \pi(\mathbf{x}_t), \mathbf{y}_t)$
Compute temporal difference and update auxiliary sequence $z_{t+1}$

$$\delta_t = r(\mathbf{x}_t, \pi(\mathbf{x}_t), \mathbf{y}_t) + \gamma V_t(\mathbf{y}_t) - V_t(\mathbf{x}_t)\ , \quad z_{t+1} = (1 - \beta_t) z_t + \beta_t \delta_t$$

Compute functional stochastic quasi-gradient step

$$\tilde{V}_{t+1}(\cdot) = (1 - \alpha_t \lambda)\tilde{V}_t(\cdot) - \alpha_t(\gamma\kappa(\mathbf{y}_t, \cdot) - \kappa(\mathbf{x}_t, \cdot)) z_{t+1}$$

Revise dictionary $\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \mathbf{x}_t\ , \mathbf{y}_t]$,
and weights $\tilde{\mathbf{w}}_{t+1} = [(1 - \alpha_t\lambda)\mathbf{w}_t,\ \ \alpha_t z_{t+1}, -\alpha_t \gamma z_{t+1}]$

Project function $(V_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{KOMP}(\tilde{V}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$

**end for**

# Convergence of PKGTD ("Pike")

### Theorem

*PKGTD sequences* $\{z_t, V_t\}$ *w/ regularizer* $\lambda > 0$, *step-sizes satisfying:*

$$\sum_{t=1}^{\infty} \alpha_t = \infty \,, \quad \sum_{t=1}^{\infty} \beta_t = \infty \,, \quad \sum_{t=1}^{\infty} \alpha_t^2 + \beta_t^2 + \frac{\alpha_t^2}{\beta_t} < \infty \,, \quad \epsilon_t = \alpha_t^2$$

*converges:* $V_t \to V^*$ *defined w.p.* 1, *achieving RKHS Bellman fixed pt.*

- Generally, step-sizes have to satisfy: $\alpha_t = \mathcal{O}(t^{-p_\alpha})$, $\beta_t = \mathcal{O}(t^{-p_\beta})$, $\Rightarrow p_\alpha \in (3/4, 1)$, $p_\beta \in (1/2, 2p_\alpha - 1)$.
- Increase $V_t$ accuracy w.r.t. $\mathscr{B}^\pi$ fixed pt. $\Rightarrow$ reduce regularizer $\lambda$

# Convergence of PKGTD ("Pike")

**Theorem**

*When PKGTD is run w/ constant learning rates $\alpha_t = \alpha$ and $\beta_t = \beta$, compression budget $\epsilon_t = \epsilon$ and large enough regularizer, i.e.*

$$0 < \beta < 1 \, , \alpha = \beta, \epsilon = C\alpha^2, \lambda = G_V^2 \frac{\alpha}{\beta} + \lambda_0$$

*where $C > 0$ is a scalar, $0 < \lambda_0 < 1$. Then the sub-optimality $\| V_t - V^* \|_{\mathcal{H}}^2$ converges in mean to nbhd.:*

$$\limsup_{t \to \infty} \mathbb{E} \| V_t - V^* \|_{\mathcal{H}}^2 = \mathcal{O}\left(\alpha + \alpha^2 + \alpha^3\right) \, .$$

- Larger step-sizes require $0 < \beta < 1$ but arbitrary $\alpha > 0$

$$\limsup_{t \to \infty} \mathbb{E} \| V_t - V^* \|_{\mathcal{H}}^2 = \mathcal{O}\left(\alpha^2 + \beta^2 + \frac{\alpha^2}{\beta}\left[1 + \alpha^2 + \frac{\alpha}{\beta} + \frac{\alpha^2}{\beta^2}\right]\right) \, .$$

$\Rightarrow$ dominated by ratios $\alpha^2/\beta$ and $\alpha^2/\beta^2$

# Convergence of PKGTD ("Pike")

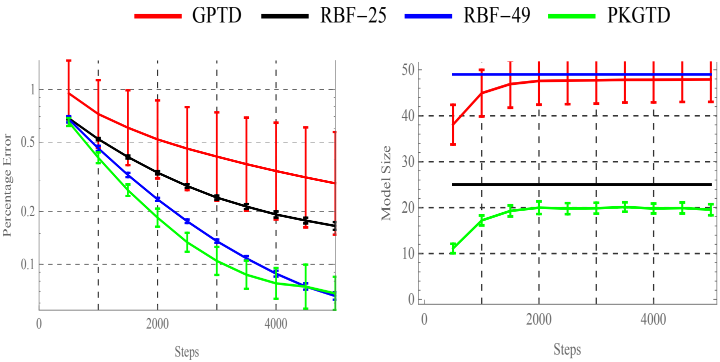
**Corollary**

*The PKGTD sequence $V_t$ run with constant step-sizes $\alpha_t = \alpha$ and $\beta_t = \beta \in (0, 1)$, compression budget $\epsilon_t = \epsilon = C\alpha^2$, and regularizer $\lambda = (\alpha/\beta)G_V^2 + \lambda_0 = \mathcal{O}(\alpha\beta^{-1} + 1)$ has finite model order for all $t$, i.e., $M_t \leq M^\infty < \infty$ for some $M^\infty$, as does its limit $V^\infty = \lim_t V_t$.*

# The Mountain Car Problem

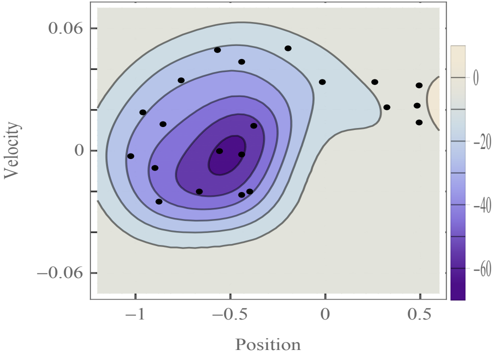
- Mountain Car (Sutton, '98): agent at bottom of valley
  - $\Rightarrow$ attempts to climb up to top of mountain side
  - $\Rightarrow$ actions $\mathcal{A} = \{\texttt{reverse}, \texttt{coast}, \texttt{forward}\}$
  - $\Rightarrow$ continuous state: scalar position & velocity: $\mathcal{X} = \mathbb{R}^2$.
- Reward function $r(\mathbf{x}_t, \mathbf{a}_t, \mathbf{y}_t)$ is $-1$
  - $\Rightarrow$ unless $\mathbf{y}_t$ is goal state at mountain top, in which case it's 0
- Benchmark policy $\Rightarrow$ trust region policy opt. (Schulman '15)
- Training set of states & rewards $\Rightarrow$ run policy for 5000 steps
- Ground truth via "Monte Carlo:" generate 10000 step trajectory
  - $\Rightarrow$ sample 2000 states: from each, apply policy until termination
  - $\Rightarrow$ use observed discounted return as $\hat{V}_\pi(\mathbf{x})$.

# Mountain Car Value Function




- Percentage Error$(V) = (1/2000)\sum_{i=1}^{2000} |(V(\mathbf{x}_i) - \hat{V}_\pi(\mathbf{x}_i))/\hat{V}_\pi(\mathbf{x}_i)|$
- PKGTD w/ Gaussian kernel to alternatives:
  - $\Rightarrow$ Gaussian process temporal difference (GPTD) (Engel '03)
  - $\Rightarrow$ Gradient TD (GTD) (Sutton '09) w/ Gaussian features.
- PKGTD $\Rightarrow$ lowest percentage error *and* memory

# Mountain Car Value Function




- Contour plot of value function, bold dots $\Rightarrow$ kernel dict. elements
  $\Rightarrow$ plateau at mountain top is goal $\Rightarrow$ has highest value of null
- Value function tells us value we obtain in any state
  $\Rightarrow$ and where in the state space is good for achieving goal

# Conclusion

- Greedily compressed RKHS-valued stochastic approx. algs.
  $\Rightarrow$ allow us to stably reduce memory of kernelized regressors
- Accurate, stable, low complexity stat. learning w/ streaming data
  $\Rightarrow$ Extendable to multi-agent networks using dist. opt. methods
- Policy eval. in infinite MDPs $\Rightarrow$ RKHS-valued comp. stoch. prog.
  $\Rightarrow$ solved with sparse projected stochastic quasi-gradient
  $\Rightarrow$ favorable trade-off in memory vs. accuracy
- Compressed kernels $\Rightarrow$ stable, low-memory, highly accurate

# Future Directions

**Near Term:**

- General compositional stochastic prog. in RKHS
  $\Rightarrow$ minimizing estimator variance, Bellman optimality eqn.
- Adaptive kernels $\Rightarrow$ Optimize kernel parameters & model points

# Future Directions

**Near Term:**

- General compositional stochastic prog. in RKHS
  $\Rightarrow$ minimizing estimator variance, Bellman optimality eqn.
- Adaptive kernels $\Rightarrow$ Optimize kernel parameters & model points

**Longer Term:**

- Exact decentralized statistical learning via primal-dual method
  $\Rightarrow$ requires Rep. Thm. for stoch. saddle pt. prob. in RKHS
- Multi-scale kernels $\Rightarrow$ composition/linear combo of kernels
  $\Rightarrow$ benefits of multi-layer networks + stability theory in RKHS
- Reinforcement learning $\Rightarrow$ POLK for policy search & actor-critic

# References

- Parts I-II of Dissertation/Proposal Presentation:

⇒ A. Koppel, F. Jakubeic, and A. Ribeiro, “A saddle point algorithm for networked online convex optimization,” IEEE Trans. Signal Process., vol.PP, no.99, June. 2015.

⇒ A. Koppel, B. Sadler, and A. Ribeiro, ”Proximity without Consensus in Online Multi-Agent Optimization,” in IEEE Trans. Signal Proc. (submitted), Mar. 2017.

⇒ A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro, ”Online learning for characterizing unknown environments in ground robot vehicle models,” in 2016 IEEE International Conference in Intelligent Robots and Systems (IROS). IEEE, 2016.

⇒ A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, “D4L: Decentralized Dynamic Discriminative Dictionary Learning,” in IEEE Trans. Signal Info. Process over Networks., June. 2016.

- Part III of Dissertation/Defense Presentation:

⇒ A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, ”Parsimonious online learning with kernels via sparse projections in function space,” The Journal of Machine Learning Research (under review), 2017 [arXiv preprint arXiv:1612.04111, 2016].

⇒ A. Koppel, S. Paternain, C. Richard, and A. Ribeiro, ”Decentralized efficient nonparametric stochastic optimization,” in IEEE Trans. Signal Process (under preparation), 2017. [Preliminary version submitted to GlobalSIP 2017]

⇒ A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, ”Breaking bellman’s curse of dimensionality: Efficient kernel gradient temporal difference,” in Advances in Neural Information Processing Systems (under review), 2017.