Adap'Ski 2011

Bayesian optimization for adaptive MCMC

Nando de Freitas + Nimalan Mahendran + Firas Hamze
University of British Columbia
January 2011

Talk outline

1. Bayesian optimization.
   - Overview.
   - Gaussian process priors.
   - Acquisition functions.

2. Application to Adaptive MCMC
   - Overview of two-stage adaptation process.
   - Cost functions.
   - Stochastic policy.

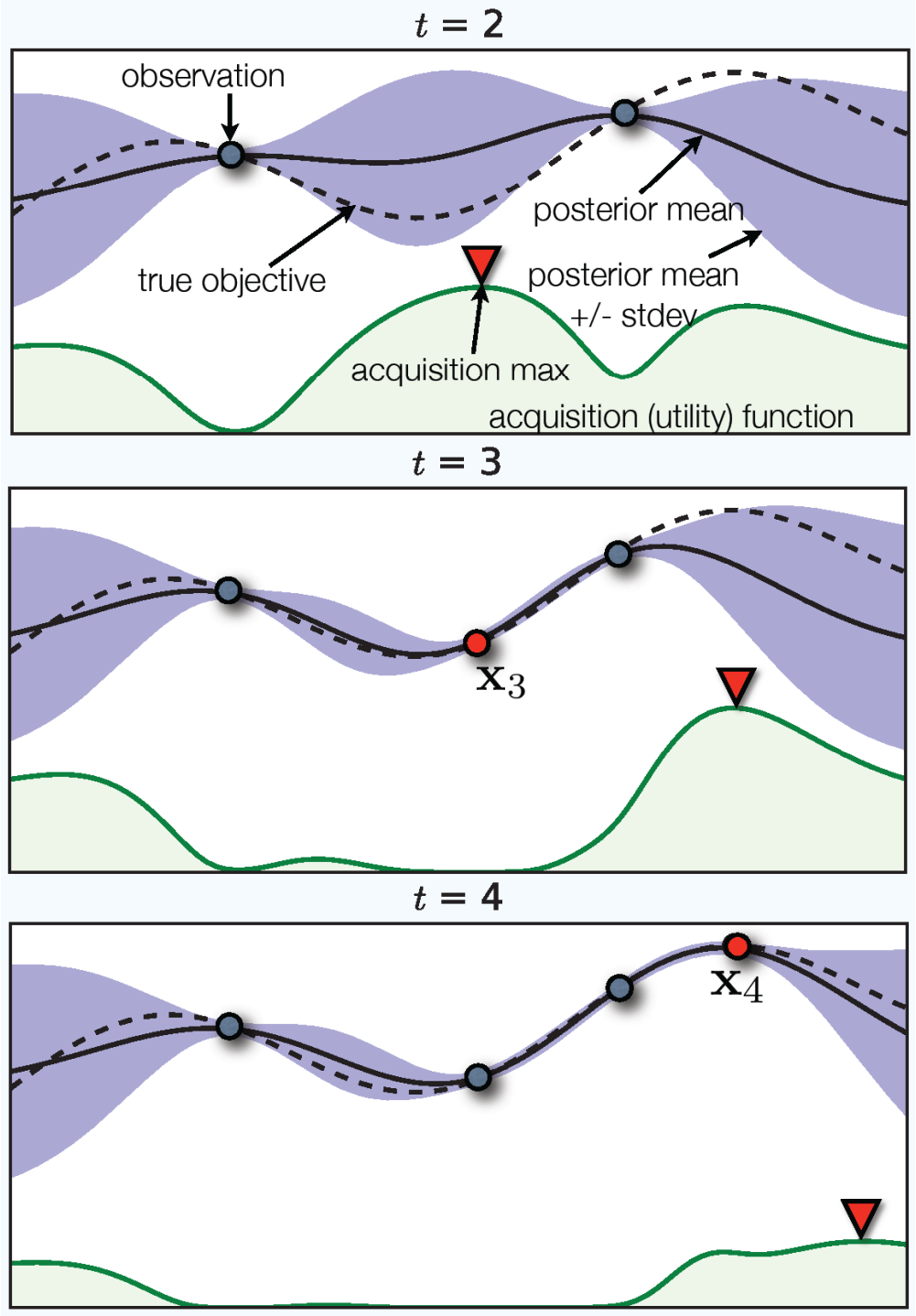
3. Application to discrete Ising models
   - 2D Ferromagnet.
   - 3D Spinglass
   - Bolzmann machine with “Gabor” filters.

Talk outline

1. Bayesian optimization.
   - Overview.
   - Gaussian process priors.
   - Acquisition functions.

2. Application to Adaptive MCMC
   - Overview of two-stage adaptation process.
   - Cost functions.
   - Stochastic policy.

3. Application to discrete Ising models
   - 2D Ferromagnet.
   - 3D Spinglass
   - Bolzmann machine with “Gabor” filters.



Goal: Optimize an expensive function that is only known point-wise. No need for derivatives.

Bayesian optimization

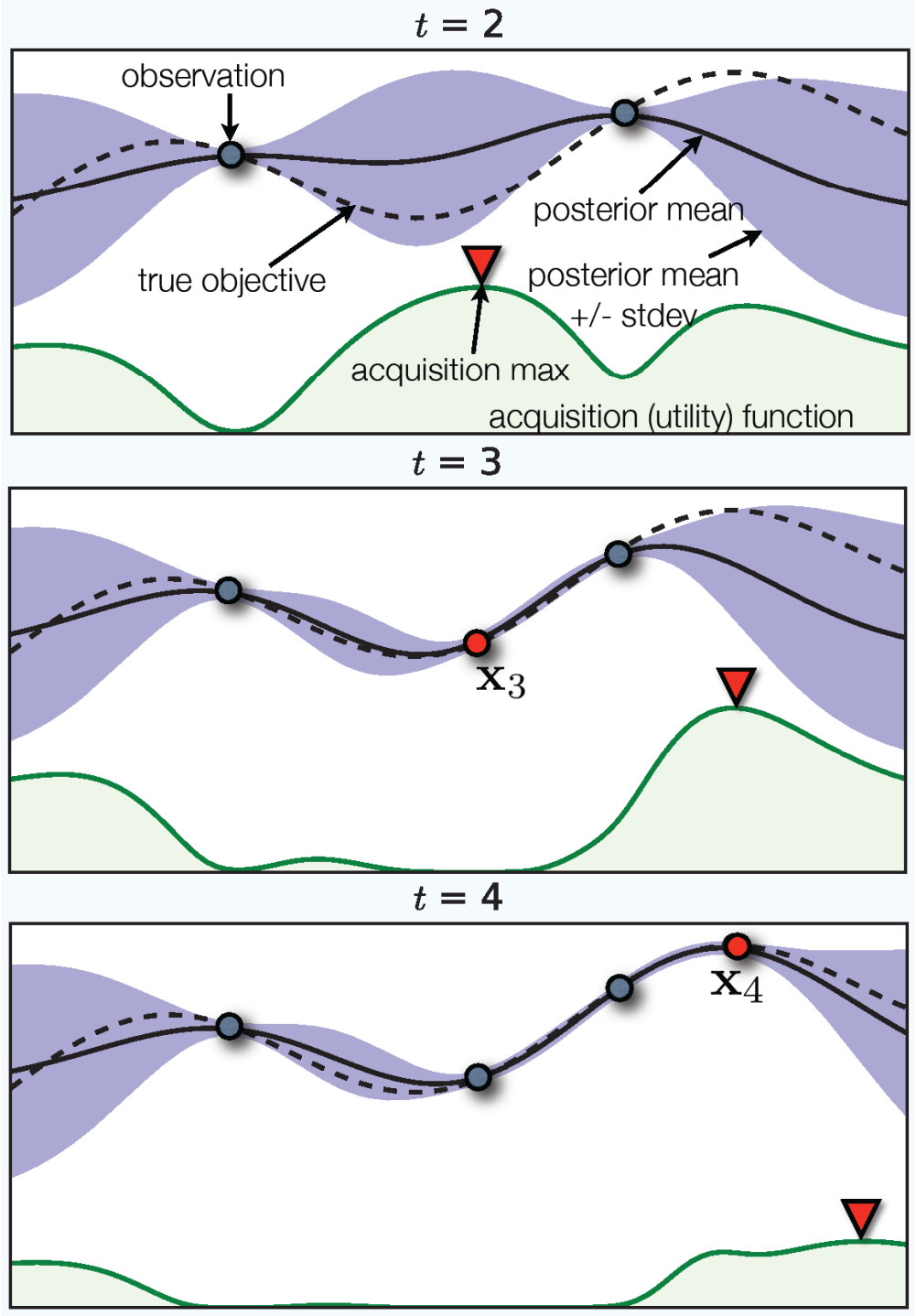
1: **for** $t = 1, 2, \ldots$ **do**

2: Find $\mathbf{x}_t$ by combining attributes of the posterior distribution in a utility function u and maximizing: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}}\, u(\mathbf{x}|\mathcal{D}_{1:t-1})$.

3: Sample the objective function: $y_t = f(\mathbf{x}_t) + \varepsilon_t$.

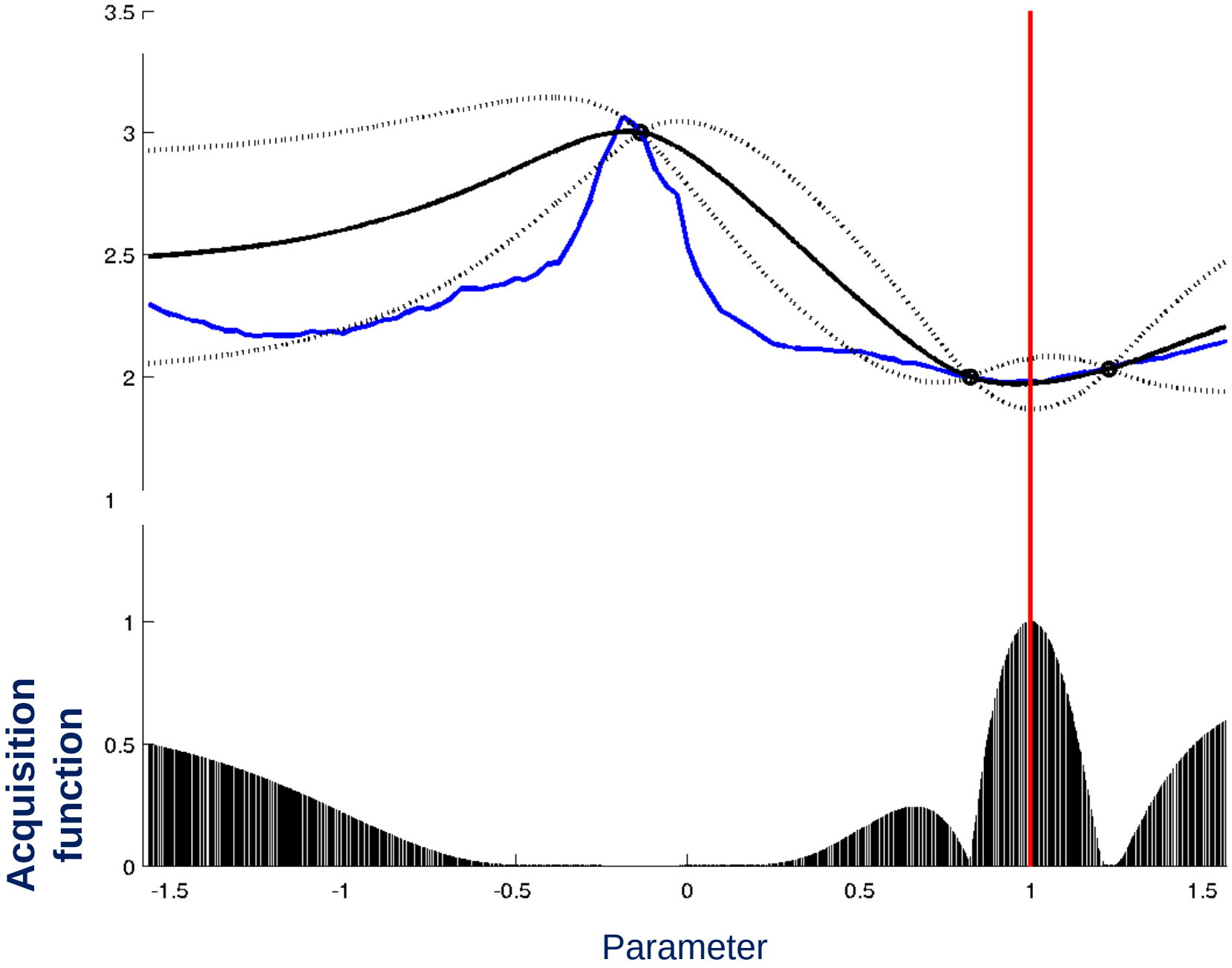
4: Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.

5: **end for**



Bayesian Optimization

- shown to be extremely efficient way to optimize (in terms of number of samples)
- works when objective is nonconvex, has unknown derivatives, etc.
- two parts: model of the objective and the acquisition function


3.5
3
2.5
2
1
1
0.5
0
-1.5
-1
-0.5
0
0.5
1
1.5
Acquisition function
Parameter

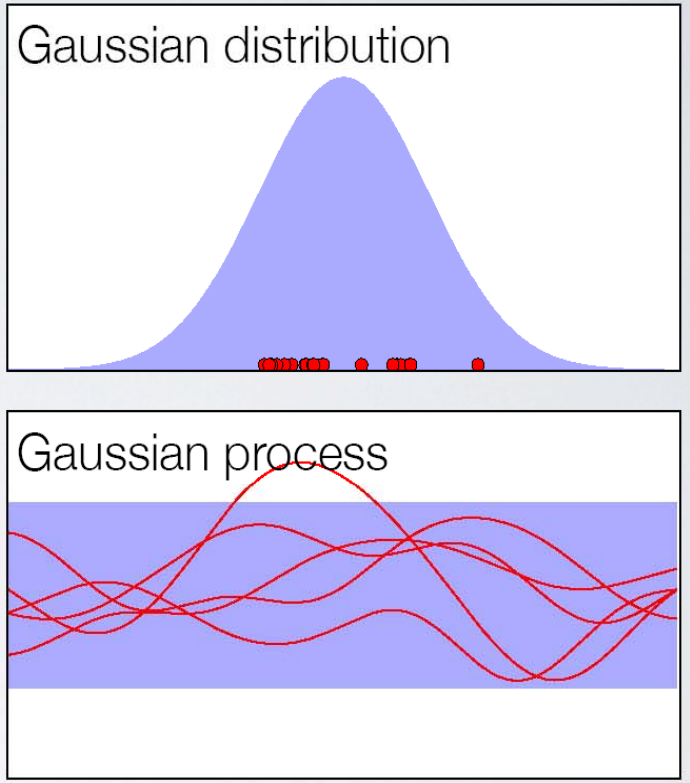
Place a Gaussian process on "unknown" cost function

- GP is generalization of multivariate Gaussian to infinitely many dimensions
- Gaussian distribution is fully specified by mean vector and covariance matrix:

$$\mathbf{f}_{1:n} \sim \mathcal{N}(\mu_{1:n}, \Sigma_{1:n,1:n})$$

- Gaussian process is fully specified by mean and covariance functions:

$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot,\cdot))$$



Gaussian process regression

- given a GP prior $f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot,\cdot))$ $\quad k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\theta}\right]$

- and a set of data $\mathcal{D}_{1:t} = \mathbf{x}_{1:t}, y_{1:t}$

$$y_i = f(\mathbf{x}_i) + \epsilon$$
$$\epsilon \sim \mathcal{N}(0, \sigma_n^2)$$

$$\begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & \dots & k(\mathbf{x}_1,\mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t,\mathbf{x}_1) & \dots & k(\mathbf{x}_t,\mathbf{x}_t) \end{bmatrix}$$

- we get the posterior distribution over functions:

$$f(\mathbf{x}')|\mathcal{D}_{1:t} \sim \mathcal{GP}(\mathbf{k}^T\mathbf{K}^{-1}y_{1:t}, k(\mathbf{x}',\mathbf{x}') - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k})$$

$[k(\mathbf{x}',\mathbf{x}_1),\dots,k(\mathbf{x}',\mathbf{x}_t)]$

- gives us predictive posterior distribution

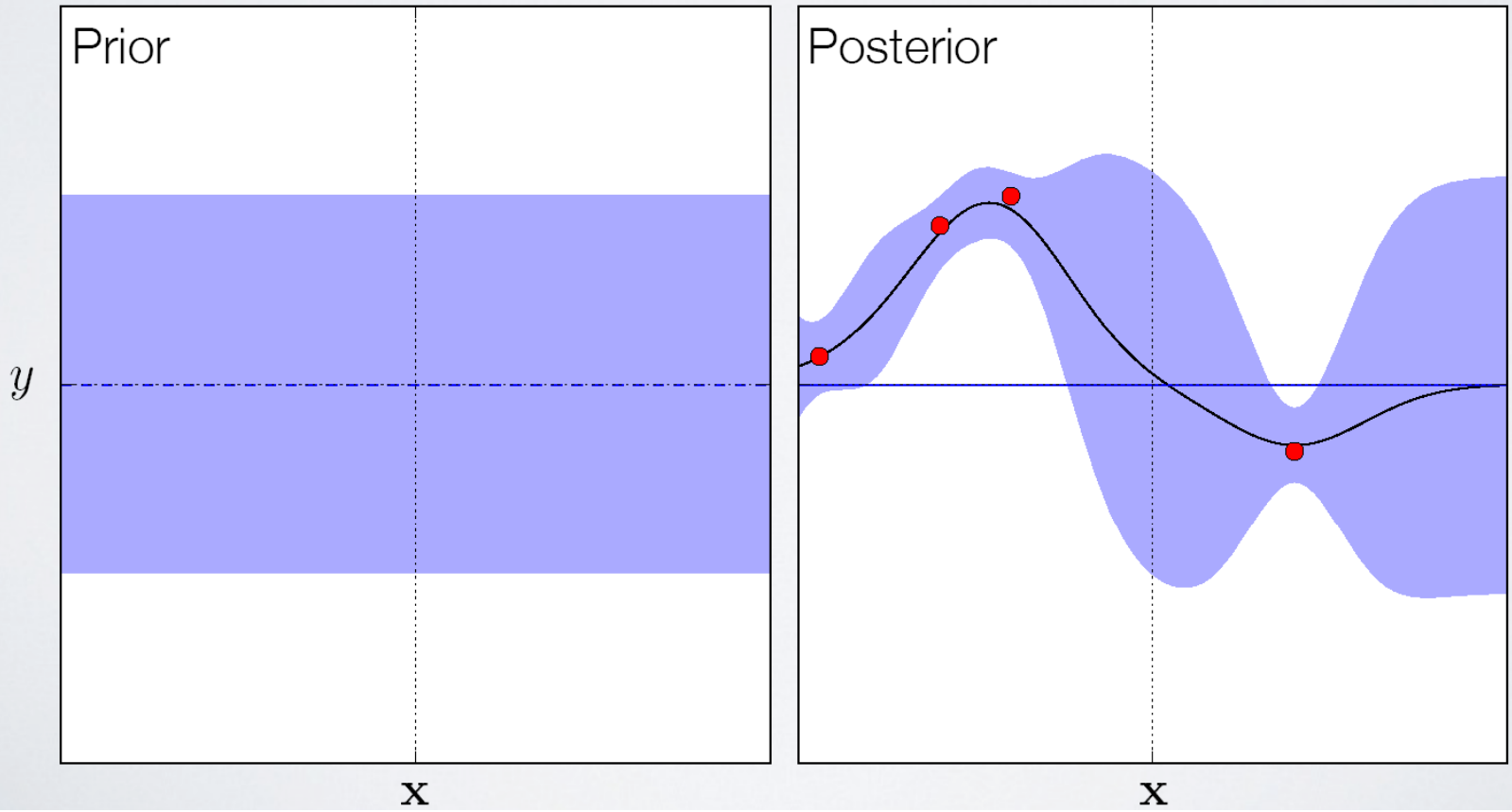
$$y_{t+1}|\mathcal{D}_{1:t}, \mathbf{x}_{t+1} \sim \mathcal{N}(\underbrace{\mathbf{k}^T[\mathbf{K} + I\sigma_n^2]^{-1}y_{1:t}}_{\mu(\mathbf{x}_{t+1})}, \underbrace{k(\mathbf{x}_{t+1},\mathbf{x}_{t+1}) - \mathbf{k}^T[\mathbf{K} + I\sigma_n^2]^{-1}\mathbf{k} + \sigma_n^2}_{\sigma^2(\mathbf{x}_{t+1}) + \sigma_n^2})$$

Prior

$$y \sim \mathcal{N}(0, 1 + \sigma_n^2)$$

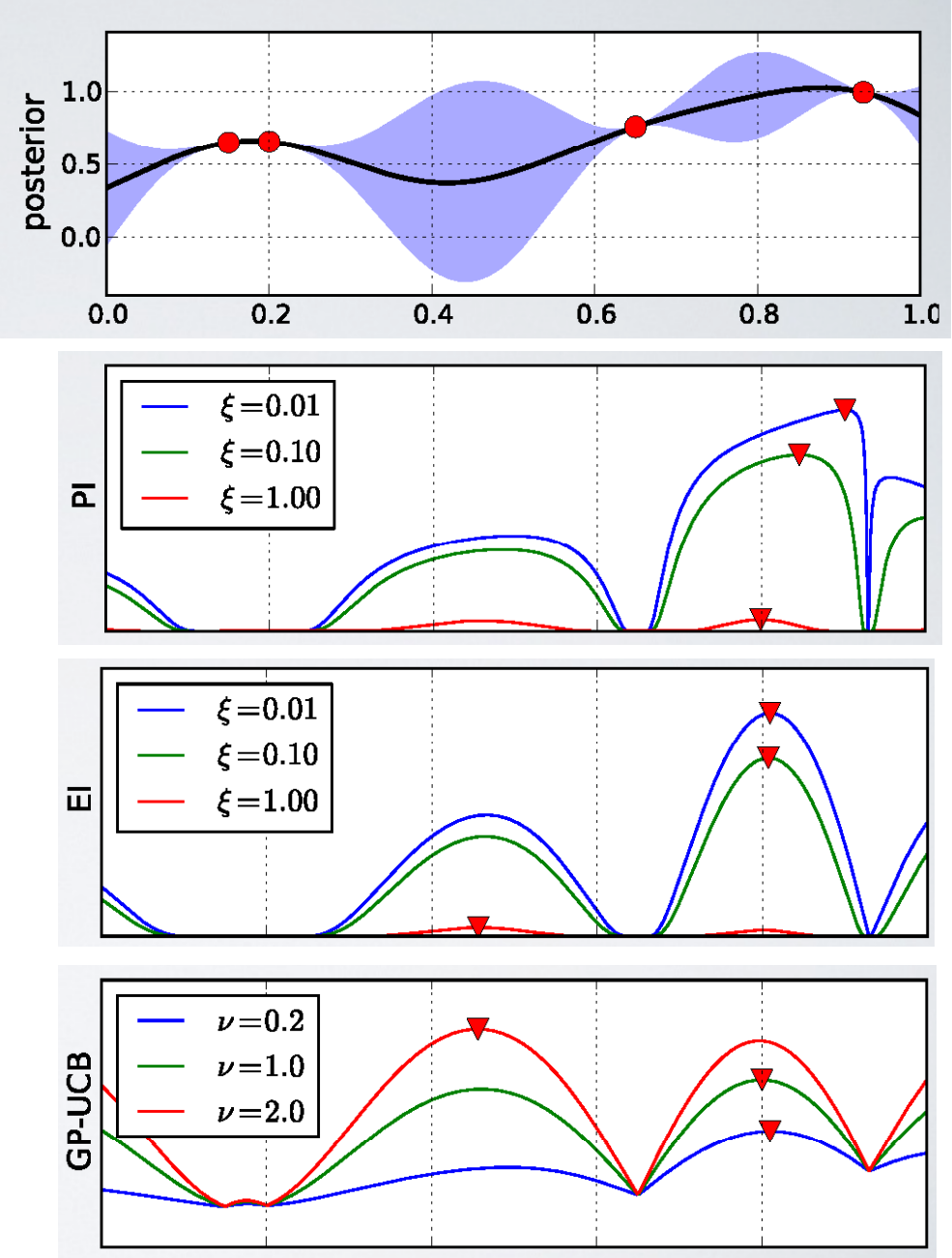
Posterior

$$y_{t+1}|\mathcal{D}_{1:t}, \mathbf{x}_{t+1} \sim \mathcal{N}(\underbrace{\mathbf{k}^T[\mathbf{K} + I\sigma_n^2]^{-1} y_{1:t}}_{\mu(\mathbf{x}_{t+1})}, \underbrace{k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) - \mathbf{k}^T[\mathbf{K} + I\sigma_n^2]^{-1}\mathbf{k} + \sigma_n^2}_{\sigma^2(\mathbf{x}_{t+1}) + \sigma_n^2})$$



Acquisition functions

- aka infill, figure of merit
- acquisition function guides the optimization by determining which $\mathbf{x}_{t+1}$ to observe next
- uses predictive posterior to combine exploration (high-variance regions) and exploitation (high-mean regions)
- optimize to find sample point (can be done cheaply/approximately)



Acquisition functions

$$\mu^+ = \text{argmax}_{\mathbf{x}_i \in \mathbf{x}_{1:t}} \mu(\mathbf{x}_i)$$

- Probability of Improvement

$$\text{PI}(\mathbf{x}) = \Phi\left(\frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}\right)$$

Kushner 1964

- Expected Improvement

$$\text{EI}(\mathbf{x}) = (\mu(\mathbf{x}) - \mu^+ - \xi)\Phi(Z) + \sigma(\mathbf{x})\phi(Z)$$

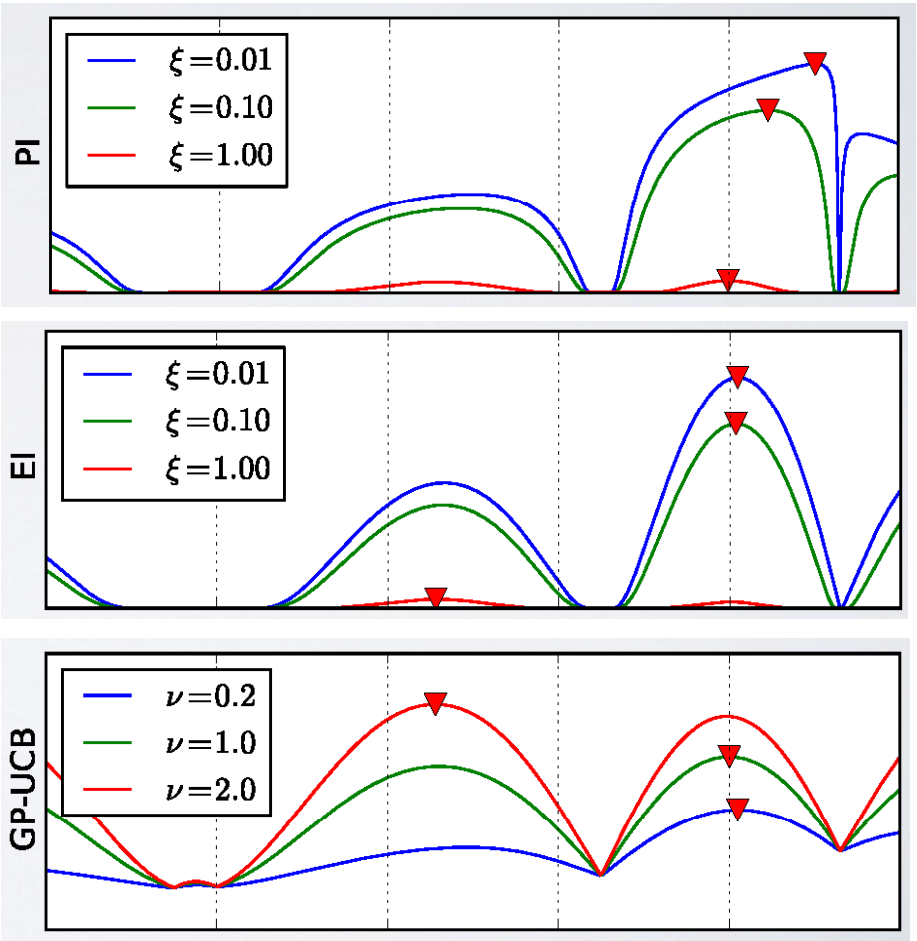
$$Z = \frac{\mu(\mathbf{x}) - \mu^+ - \xi}{\sigma(\mathbf{x})}$$

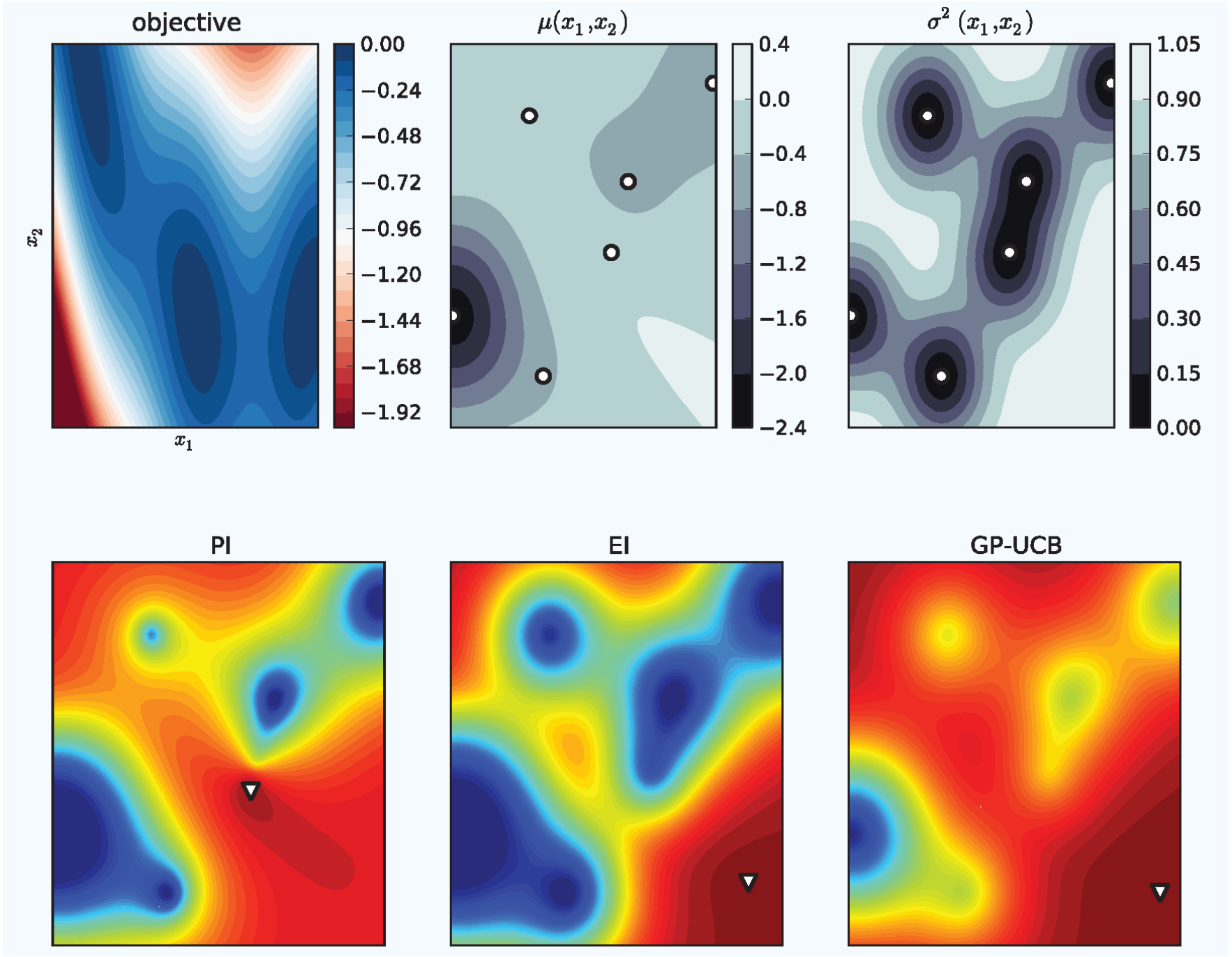
Mockus 1978

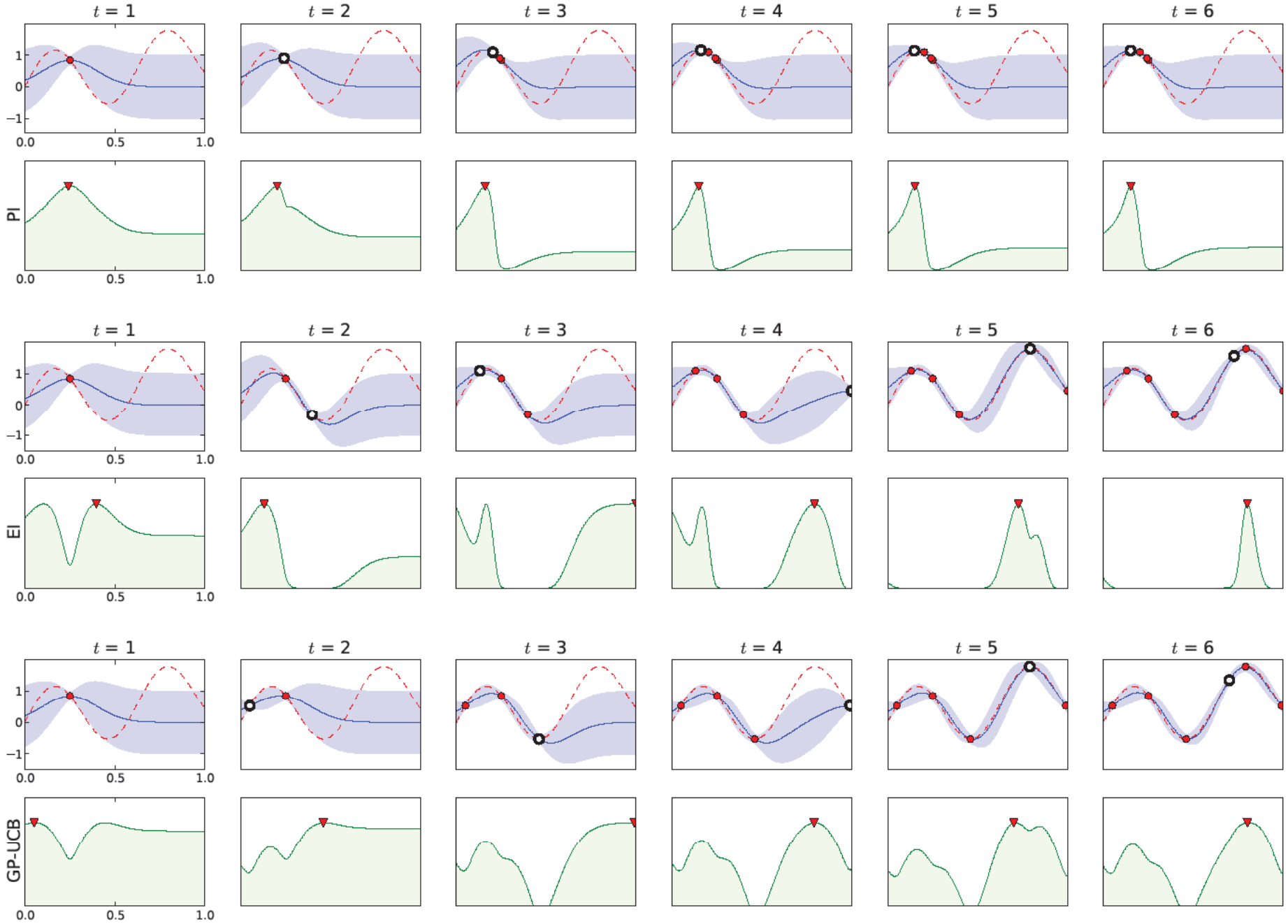
- Upper Confidence Bound

$$\text{GP-UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \sqrt{\nu\tau_t}\sigma(\mathbf{x})$$

Srinivas *et al.* 2010

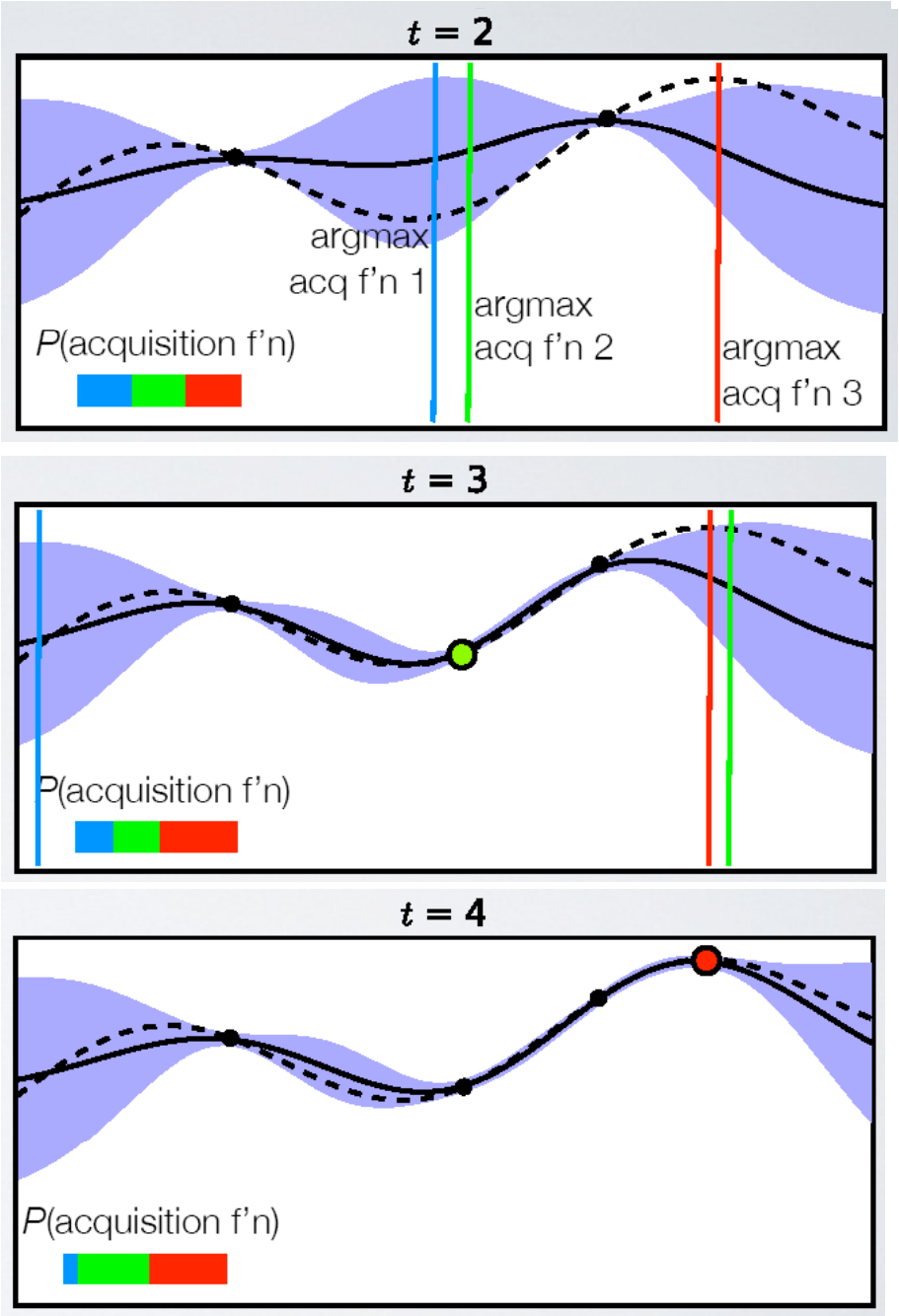



objective
$\mu(x_1, x_2)$
$\sigma^2(x_1, x_2)$
x_2
x_1
0.00
−0.24
−0.48
−0.72
−0.96
−1.20
−1.44
−1.68
−1.92
0.4
0.0
−0.4
−0.8
−1.2
−1.6
−2.0
−2.4
1.05
0.90
0.75
0.60
0.45
0.30
0.15
0.00
PI
EI
GP-UCB


$t = 1$
$t = 2$
$t = 3$
$t = 4$
$t = 5$
$t = 6$
PI
EI
GP-UCB
0.0
0.5
1.0
1
0
−1

- treat acquisition functions as strategies -- each nominates a point and we sample based on past performance

- pick one, update cumulative rewards for *all*, according to the posterior mean at the nominee location



Algorithm 3 Hedge [Auer *et al.*, 1998]

Select parameter $\eta \in \mathbb{R}^+$.
Set $g_0^i = 0$ for $i = 1, \ldots, K$.
for $t = 1, 2, \ldots$ **do**
    Choose action i_t with probability $p_t(i) = \exp(\eta g_{t-1}^i) / \sum_{\ell=1}^{K} \exp(\eta g_{t-1}^\ell)$.
    Receive rewards r_t^i.
    Update gains $g_t^i = g_{t-1}^i + r_t^i$.
end for

Algorithm 4 GP-Hedge

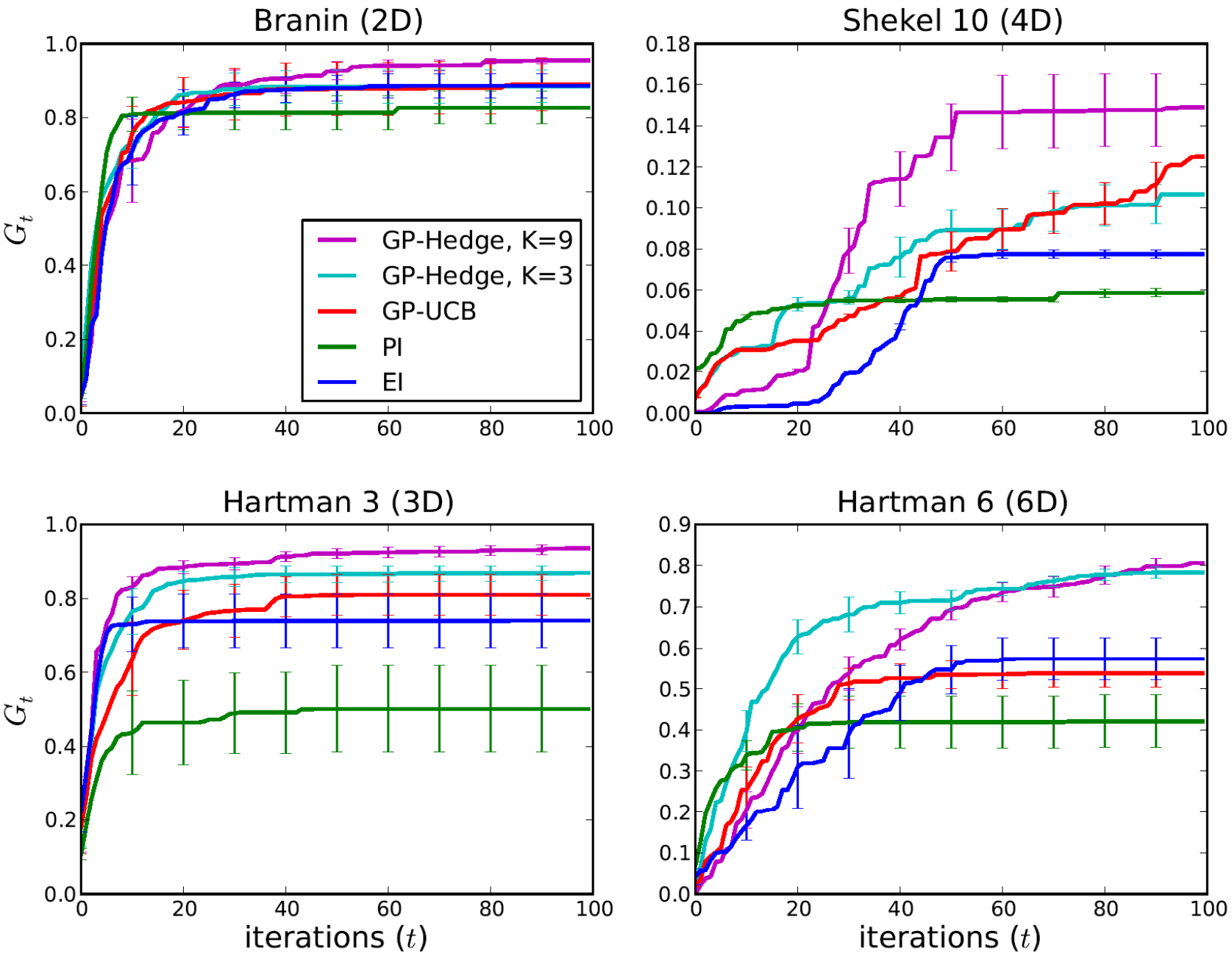
Select parameter $\eta \in \mathbb{R}^+$.
Set $g_0^i = 0$ for $i = 1, \ldots, K$.
for $t = 1, 2, \ldots$ **do**
    Nominate points from each acquisition function: $\mathbf{x}_t^i = \text{argmax}_{\mathbf{x}}\, u_i(\mathbf{x} | \mathcal{D}_{1:t-1})$.
    Select nominee $\mathbf{x}_t = \mathbf{x}_t^j$ with probability $p_t(j) = \exp(\eta g_{t-1}^i) / \sum_{\ell=1}^{K} \exp(\eta g_{t-1}^\ell)$.
    Sample the objective function $y_t = f(\mathbf{x}_t) + \epsilon_t$.
    Augment the data $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.
    Receive rewards $r_t^i = \mu_t(\mathbf{x}_t^i)$ from the updated GP.
    Update gains $g_t^i = g_{t-1}^i + r_t^i$.
end for

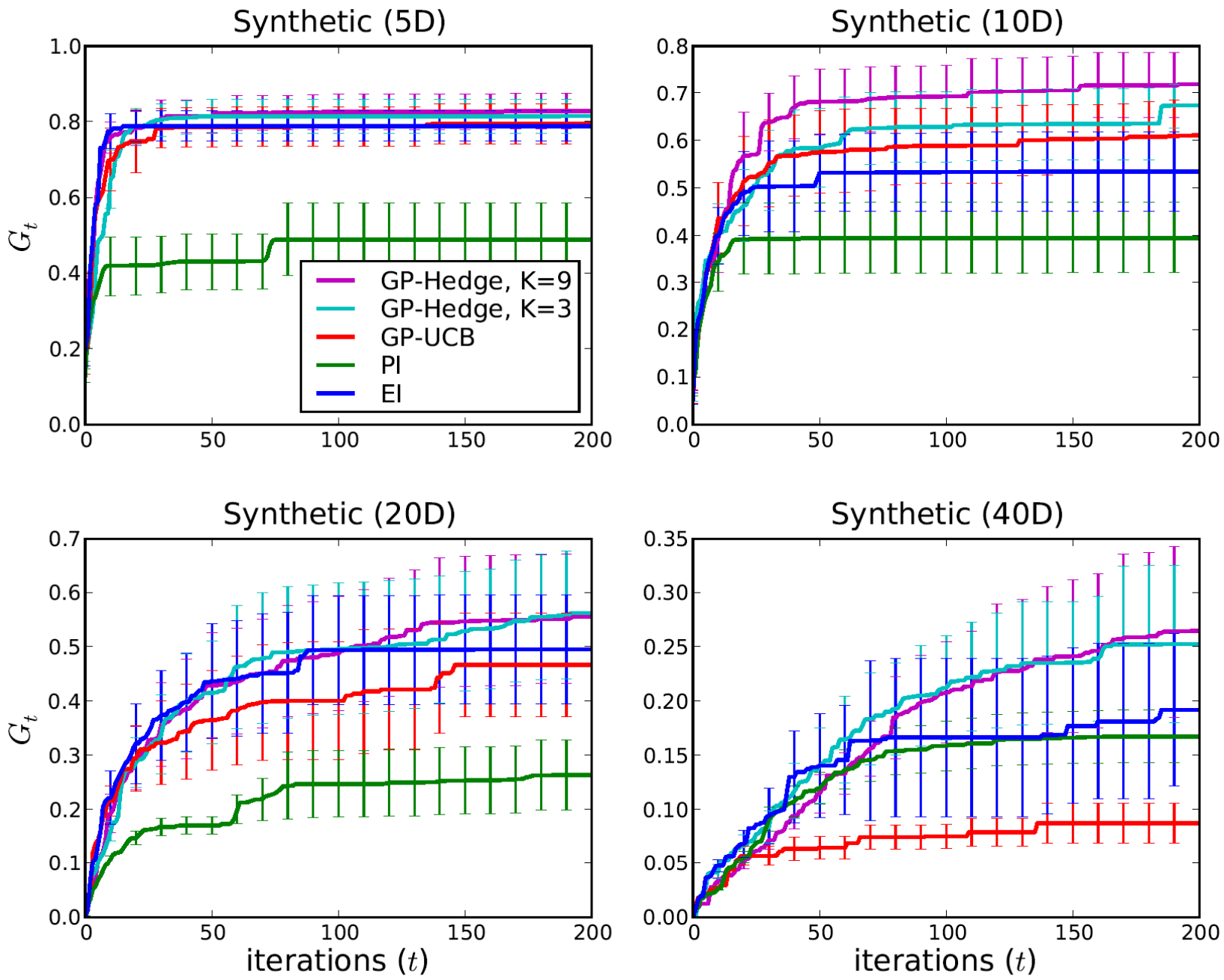
Could use other bandit algorithms instead…
e.g. algorithms for imperfect observation games

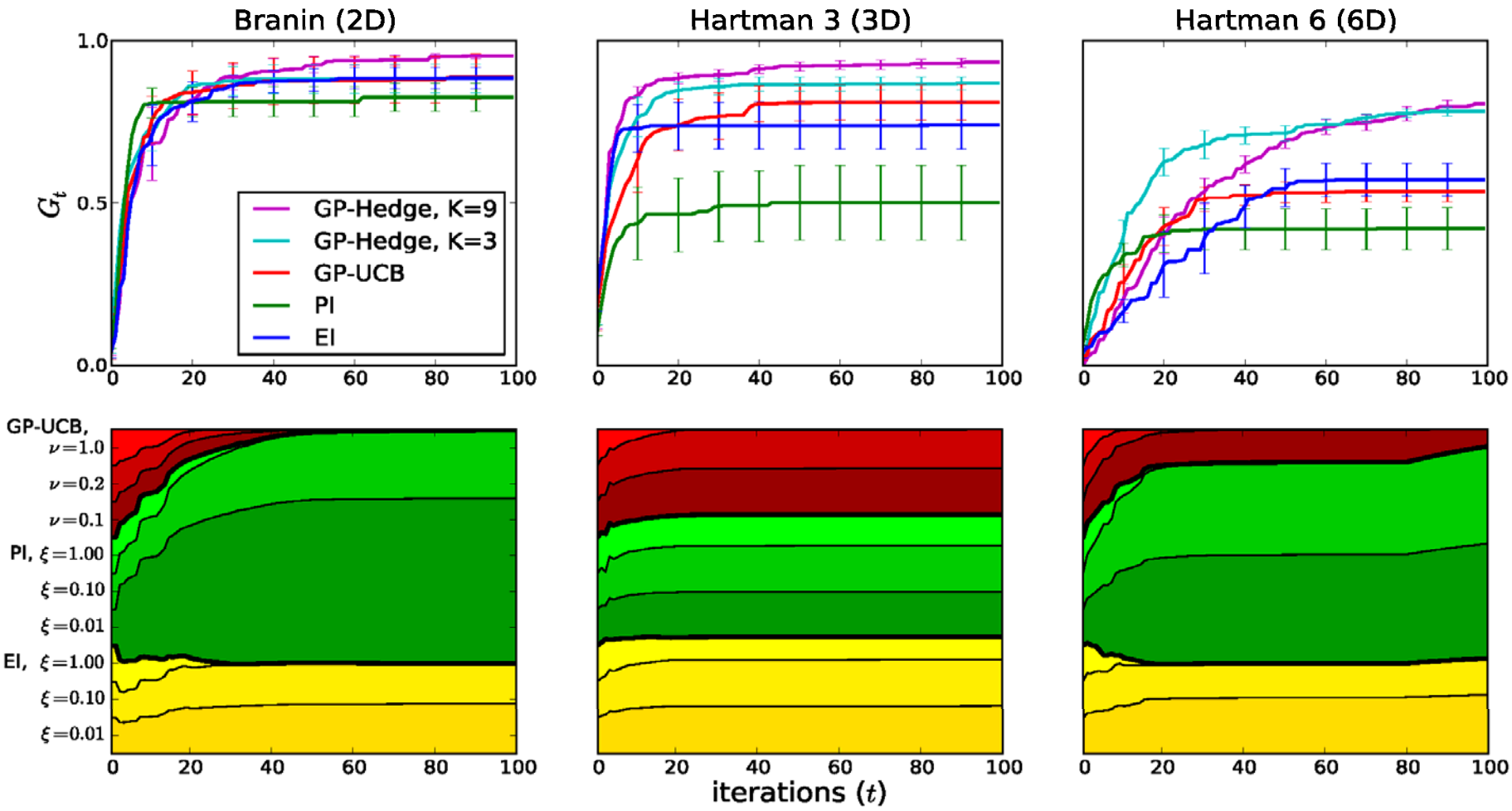
Algorithm 5 Exp3 [Auer *et al.*, 1998]

Select parameters $\eta \in \mathbb{R}^+$, $\gamma \in (0, 1]$.
for $t = 1, 2, \ldots$ **do**
  Get distribution $\mathbf{p}_t$ from Hedge (Algorithm 3).
  Choose action i_t to be j with probability $\hat{p}(j) = (1 - \gamma)p_t(j) + \gamma/K$.
  Receive reward $r_t^{i_t} \in [0, 1]$.
  Return simulated reward vector $\hat{\mathbf{r}}_t$ to Hedge, where $\hat{r}_t^i = r_t^i/\hat{p}_t(i)$ if $j = i_t$, or $\hat{r}_t^i = 0$ otherwise.
end for

$$G_t = \frac{f(\mathbf{x}^+) - f(\mathbf{x}^{\text{first}})}{f(\mathbf{x}^\star) - f(\mathbf{x}^{\text{first}})}$$




Synthetic (5D)
Synthetic (10D)
Synthetic (20D)
Synthetic (40D)
G_t
iterations (t)
GP-Hedge, K=9
GP-Hedge, K=3
GP-UCB
PI
EI


Branin (2D)
Hartman 3 (3D)
Hartman 6 (6D)
G_t
1.0
0.5
0.0
0
20
40
60
80
100
GP-Hedge, K=9
GP-Hedge, K=3
GP-UCB
PI
EI
GP-UCB,
ν=1.0
ν=0.2
ν=0.1
PI, ξ=1.00
ξ=0.10
ξ=0.01
EI, ξ=1.00
ξ=0.10
ξ=0.01
iterations (t)

Theorem 1. *Assume Exp3 is used to select between acquisition strategies, one of which is GP-UCB with parameters β_t, and we have a bound γ_T on the information gained at points selected by the algorithm after T iterations. Then with probability at least $1-\delta$ the cumulative regret is bounded by*

$$R_T \leq \sqrt{TC_1\beta_T\gamma_T} + \left[\sum_{t=1}^{T} \beta_t \sigma_{t-1}(\mathbf{x}_t^{\mathrm{UCB}})\right] + O(T^{2/3}),$$

Talk outline

1. Bayesian optimization.
   - Overview.
   - Gaussian process priors.
   - Acquisition functions.

2. Application to Adaptive MCMC
   - Overview of two-stage adaptation process.
   - Cost functions.
   - Stochastic policy.

3. Application to discrete Ising models
   - 2D Ferromagnet.
   - 3D Spinglass
   - Bolzmann machine with “Gabor” filters.

Bayesian optimized MCMC

Algorithm 1 Adaptive MCMC with Bayesian Optimization

1: **for** $i = 1, 2, \ldots, I$ **do**
2: Run Markov chain for L steps with parameters θ_i.
3: Use the drawn samples to obtain a noisy evaluation of the objective function: $z_i = h(\theta_i) + \epsilon$.
4: Augment the data $\mathcal{D}_{1:i} = \{\mathcal{D}_{1:i-1}, (\theta_i, z_i)\}$.
5: Update the GP's sufficient statistics.
6: Find θ_{i+1} by optimizing the acquisition function: $\theta_{i+1} = \arg\max_\theta u(\theta|\mathcal{D}_{1:i})$.
7: **end for**

Compare with stochastic optimization:

$$g(\theta) = \nabla h(\theta)$$

$$\theta_{i+1} = \theta_i + \gamma_{i+1} G\left(\theta_i, \mathbf{x}^{(0)}, \mathcal{X}_{i+1}, \mathcal{Y}_{i+1}\right)$$

Objective function

$$r(l,\theta) \triangleq \frac{1}{\delta_\theta^2}\mathbb{E}\left[(\mathbf{x}_\theta^{(t)} - \bar{\mathbf{x}}_\theta)^T(\mathbf{x}_\theta^{(t+l)} - \bar{\mathbf{x}}_\theta)\right]$$

$$h(\theta) = 1 - ({l_{\max}}^{-1}) \sum_{l=1}^{l_{\max}} |r(l,\theta)|$$

$$\widehat{r}(l,\mathcal{X}^\theta) \triangleq \frac{1}{(L-l)\delta_\theta^2}\sum_{t=1}^{L-l}(\mathbf{x}_\theta^{(t)} - \bar{\mathbf{x}}_\theta)^T(\mathbf{x}_\theta^{(t+l)} - \bar{\mathbf{x}}_\theta)$$

$$\widehat{a}(\mathcal{X}^\theta) = 1 - ({l_{\max}}^{-1}) \sum_{l=1}^{l_{\max}} |\widehat{r}(l,\mathcal{X}^\theta)|$$

$$\widehat{h}(\mathcal{X}^\theta) \triangleq \frac{1}{L-l_{\min}+1} \sum_{i=l_{\min}}^{L} \widehat{a}(\mathcal{E}_i)$$

Markov chain $\mathcal{X}^\theta = \{\mathbf{x}_\theta^{(1)}, \mathbf{x}_\theta^{(2)}, \ldots\}$

Stochastic policy / mixture of MCMC kernels

Algorithm 2 SIR-based policy construction

1: Generate a set of candidate points $\widetilde{\theta}_{1:N} \triangleq \{\widetilde{\theta}_1, \ldots, \widetilde{\theta}_N\}$, using either Latin hypercubes or optimization methods.
2: Obtain the weights $\widetilde{w}_i = \exp(\mu(\widetilde{\theta}_i))$ for $i = 1 : N$ by evaluating the Gaussian process mean function at each point in $\widetilde{\theta}_{1:N}$ and exponentiating it.
3: Normalize the weights: $w_i = \frac{\widetilde{w}_i}{\sum_{j=1}^{N} \widetilde{w}_j}$.
4: Resample, with replacement, M samples $\{\theta_i | i = 1, \ldots, M\}$ from the weighted discrete measure $\{(\widetilde{\theta}_i, w_i) | i = 1, \ldots, N\}$.

Talk outline

1. Bayesian optimization.
   - Overview.
   - Gaussian process priors.
   - Acquisition functions.

2. Application to Adaptive MCMC
   - Overview of two-stage adaptation process.
   - Cost functions.
   - Stochastic policy.

3. Application to discrete Ising models
   - 2D Ferromagnet.
   - 3D Spinglass
   - Bolzmann machine with “Gabor” filters.

Ising models

$$\text{space } \mathcal{S} \triangleq \{0,1\}^M$$

$$\pi(\mathbf{x}) = \frac{1}{Z(\beta)} e^{-\beta E(\mathbf{x})}$$

β is an *inverse temperature*

$$E(\mathbf{x}) \triangleq -\sum_{i,j} x_i J_{ij} x_j - \sum_i b_i x_i$$

Constrained Ising models

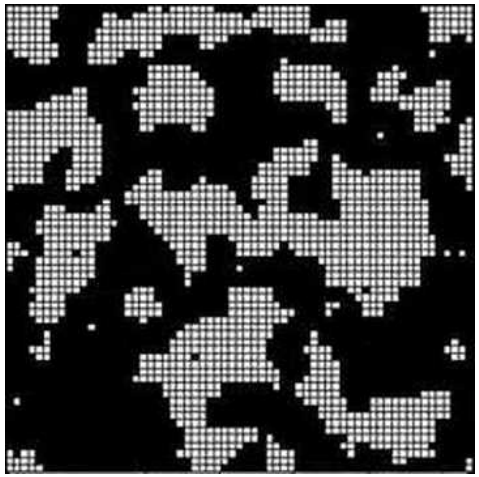
Consider a particular state $\mathbf{c}$, which for now can be considered arbitrary. We are interested in drawing samples from the set of all states at Hamming Distance n from $\mathbf{c}$, which we call $\mathcal{S}_n(\mathbf{c})$. For example if $M = 3$ and $\mathbf{c} = [1, 1, 1]$, then $\mathcal{S}_0(\mathbf{c}) = \{[1, 1, 1]\}$, $\mathcal{S}_1(\mathbf{c}) = \{[0, 1, 1], [1, 0, 1], [1, 1, 0]\}$, etc. Clearly, $|\mathcal{S}_n(\mathbf{c})| = \binom{M}{n}$. The partition function on $\mathcal{S}_n(\mathbf{c})$ is given by:

$$Z_n(\mathbf{c}) = \sum_{\mathbf{x} \in \mathcal{S}_n(\mathbf{c})} e^{-\beta E(\mathbf{x})}$$

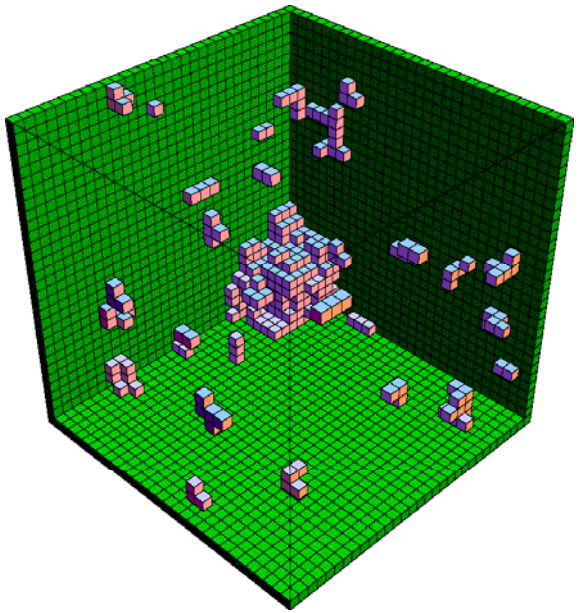
$$\pi_n(\mathbf{x}) \triangleq \begin{cases} \frac{1}{Z_n} e^{-\beta E(\mathbf{x})} & \mathbf{x} \in \mathcal{S}_n \\ 0 & \text{otherwise} \end{cases}$$

Model variations

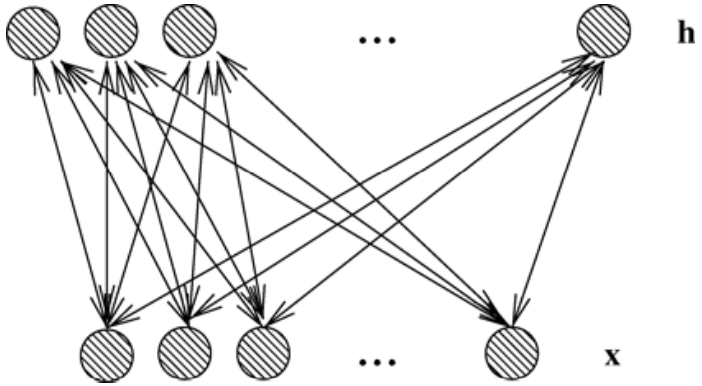
2D Ferro-magnet



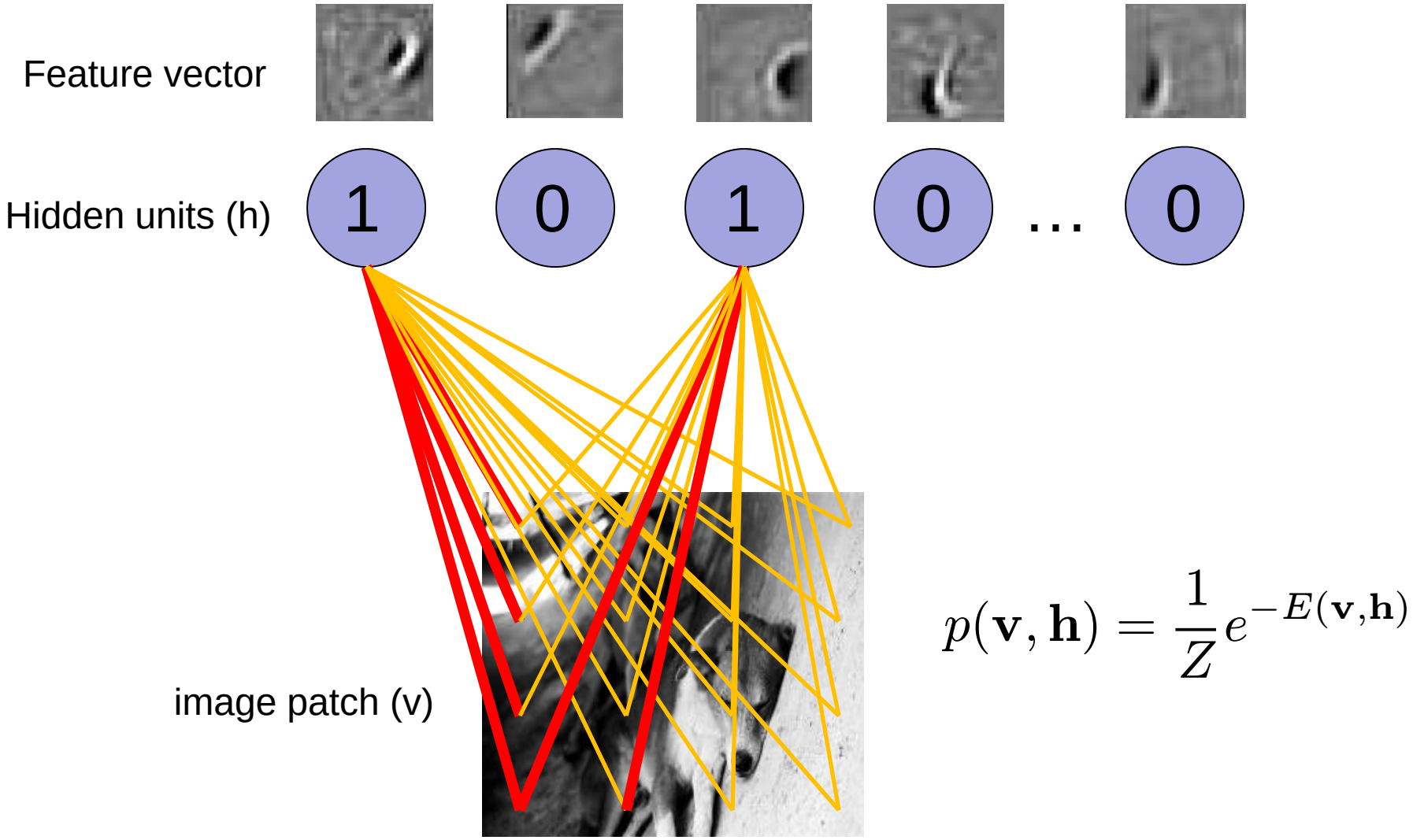
2D Ising model



Constrained RBM



All are conserved-order parameter Ising models



$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})}$$

Insight: Nature's parameters are nice !

Intra-cluster move sampler

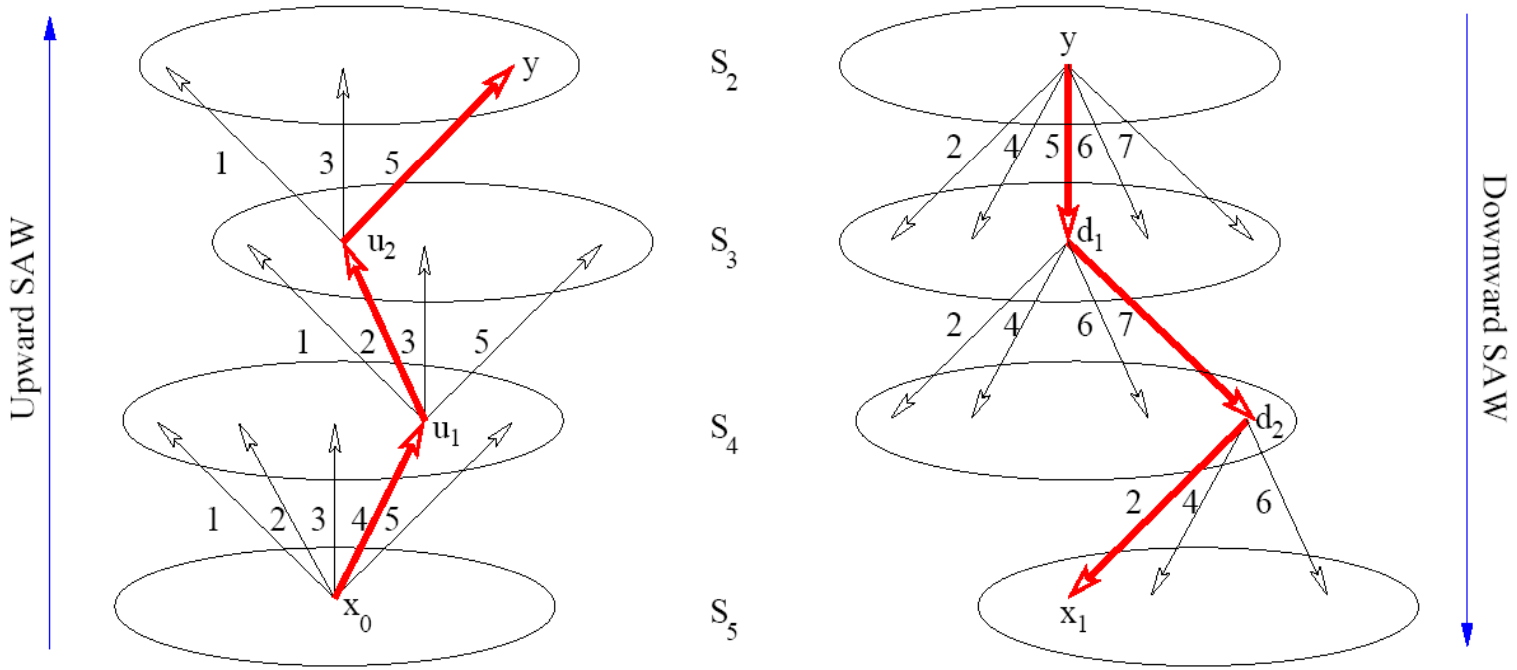


Figure 1: A visual illustration of the move process for up/down SAWs of length 3. The arrows represent the allowable moves from a state at that step; the red arrow shows the actual move taken in this example. From the system at state $\mathbf{x}_0 = [1,1,1,1,1,0,0]$ on $\mathcal{S}_5$, the *upward* SAW begins. Bit 4 of $\mathbf{x}_0$ is flipped to yield state $\mathbf{u}_1 = [1,1,1,0,1,0,0]$; the process is repeated until state $\mathbf{y} = [1,0,1,0,0,0,0]$ on $\mathcal{S}_2$ is reached. From there, the *downward* SAW considers and selects sequences of states in an analogous manner (in this case, $\mathbf{d}_1 = [1,0,1,0,1,0,0]$ is visited from $\mathbf{y}$, etc.) until the final state $\mathbf{x}_1 = [1,1,1,0,1,0,1]$ is reached. The sequence of states taken by the upward and downward SAWs are, respectively, $\boldsymbol{\sigma} = [4,2,5]$ and $\boldsymbol{\rho} = [5,7,2]$.

Intra-cluster move sampler

$$f_{up}(\sigma_1 = i|\mathbf{x}_0) = \begin{cases} \frac{e^{-\gamma E(F(\mathbf{x}_0,i))}}{\sum_{j\in\mathcal{P}(\mathbf{x}_0)} e^{-\gamma E(F(\mathbf{x}_0,j))}} & i \in \mathcal{P}(\mathbf{x}_0) \\ 0 & \text{otherwise} \end{cases}$$

$$f_{down}(\rho_1 = i|\mathbf{y}) = \begin{cases} \frac{e^{-\gamma E(F(\mathbf{y},i))}}{\sum_{j\in\mathcal{N}(\mathbf{y})} e^{-\gamma E(F(\mathbf{y},j))}} & i \in \mathcal{N}(\mathbf{y}) \\ 0 & \text{otherwise} \end{cases}$$

$$f(\mathbf{x}_1, \boldsymbol{\sigma}, \boldsymbol{\rho}|\mathbf{x}_0) \triangleq \delta_{\mathbf{x}_1}[F(\mathbf{x}_0, \boldsymbol{\sigma}, \boldsymbol{\rho})] \prod_{i=1}^{M} f_{up}(\sigma_i|\mathbf{u}_{i-1}) \prod_{i=1}^{M} f_{down}(\rho_i|\mathbf{d}_{i-1})$$

$$\alpha(\mathbf{x}_0, \mathbf{x}_1, \boldsymbol{\sigma}, \boldsymbol{\rho}) \triangleq \min\left(1, \frac{\pi_n(\mathbf{x}_1) f(\mathbf{x}_0, R(\boldsymbol{\rho}), R(\boldsymbol{\sigma})|\mathbf{x}_1)}{\pi_n(\mathbf{x}_0) f(\mathbf{x}_1, \boldsymbol{\sigma}, \boldsymbol{\rho}|\mathbf{x}_0)}\right)$$

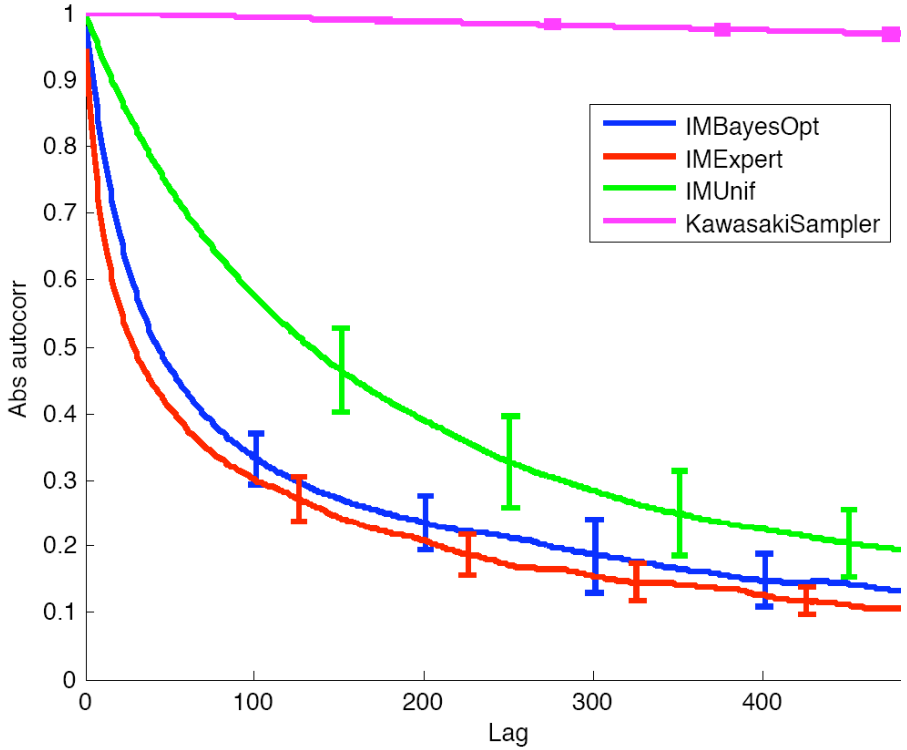
$$\sum_{\boldsymbol{\sigma}'\boldsymbol{\rho}'} \pi_n(\mathbf{x}_0) K(\mathbf{x}_1, \boldsymbol{\sigma}', \boldsymbol{\rho}'|\mathbf{x}_0) = \sum_{\boldsymbol{\sigma}'\boldsymbol{\rho}'} \pi_n(\mathbf{x}_1) K(\mathbf{x}_0, R(\boldsymbol{\rho}'), R(\boldsymbol{\sigma}')|\mathbf{x}_1)$$

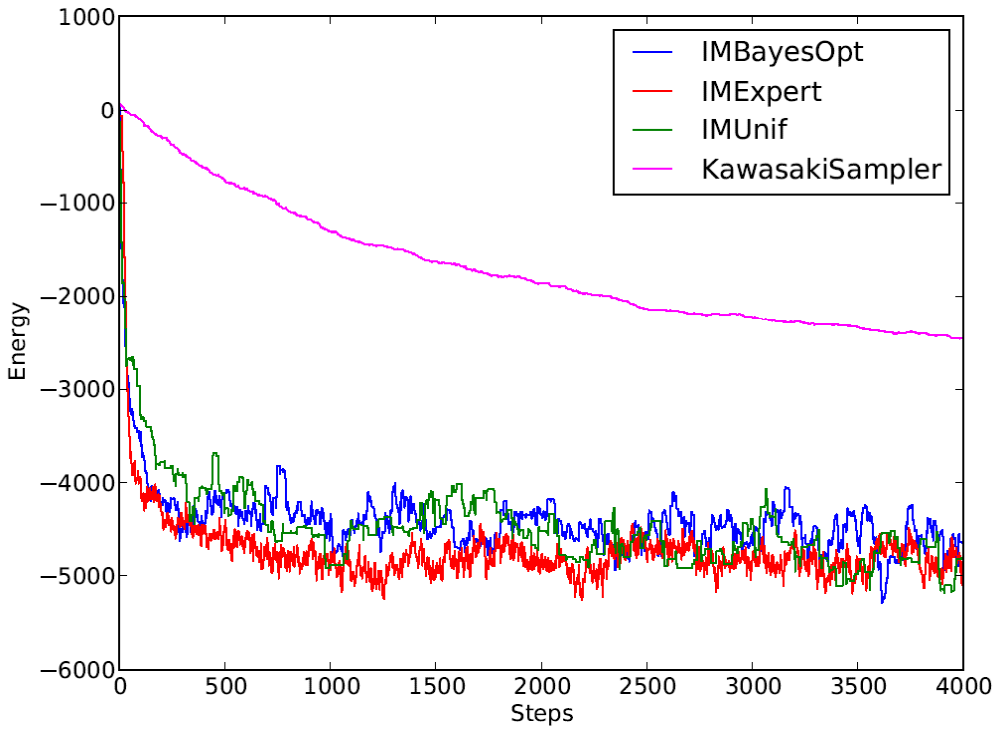
Simulation parameters

Model	β^{-1}	Size	n
2DGrid	2.27	60×60	1800 of 3600
3DCube	1.0	$9 \times 9 \times 9$	364 of 729
RBM	1.0	$v = 784, h = 500$	428 of 1284

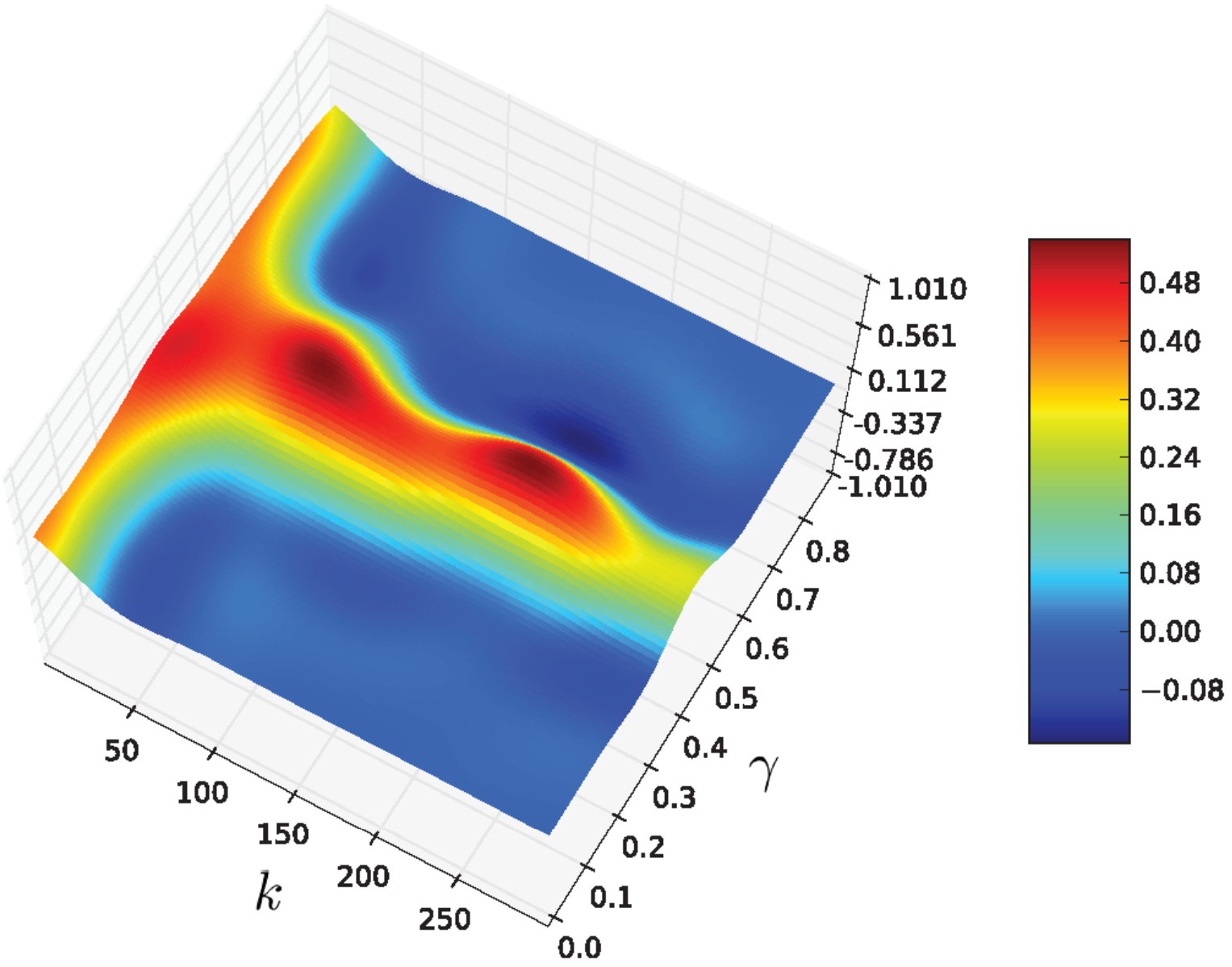
Model	Algorithm	$\mathcal{L}$	$\mathcal{G}$
2DGrid	IMExpert	$\{90\}$	$\{0.44\}$
2DGrid	Others	$\{1, \ldots, 300\}$	$[0, 0.88]$
3DCube	IMExpert	$\{1, \ldots, 25\}$	$\{0.8\}$
3DCube	Others	$\{1, \ldots, 50\}$	$[0, 1.6]$
RBM	IMExpert	$\{1, \ldots, 20\}$	$\{0.8\}$
RBM	Others	$\{1, \ldots, 50\}$	$[0, 1.6]$

Ferro-magnet

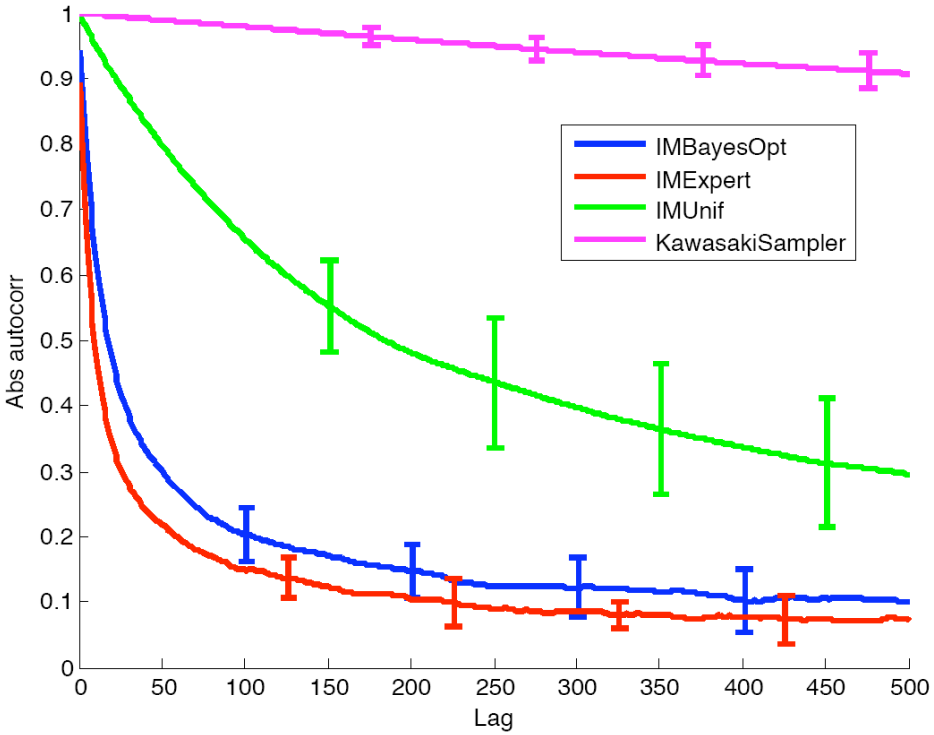


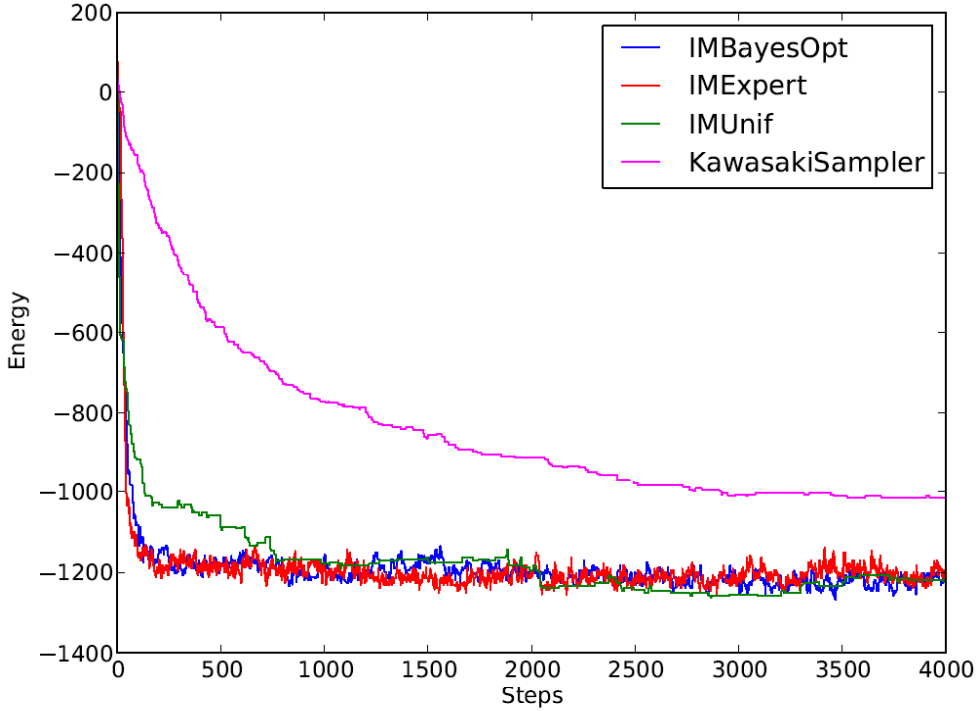


Ferro-magnet

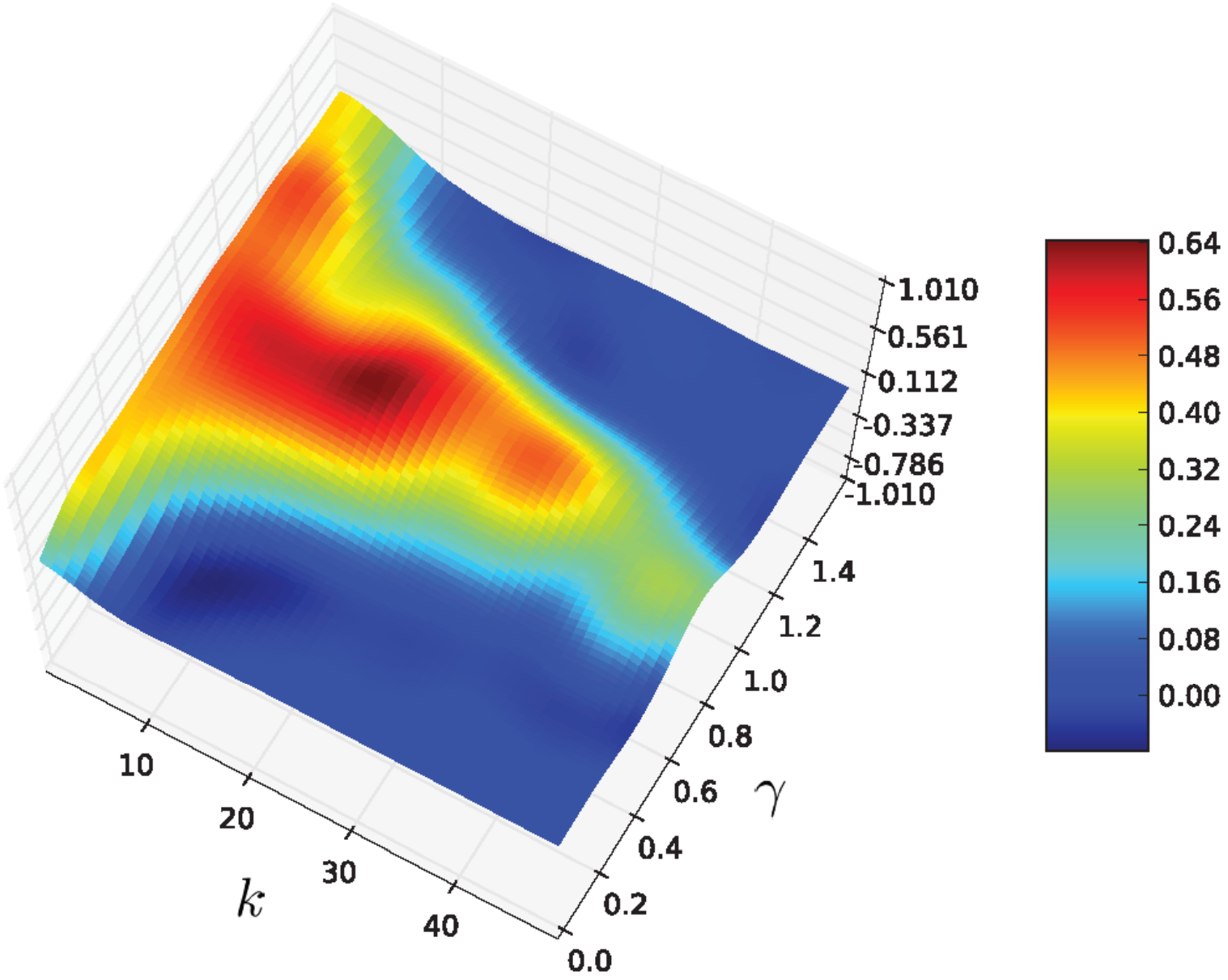


3D Ising model

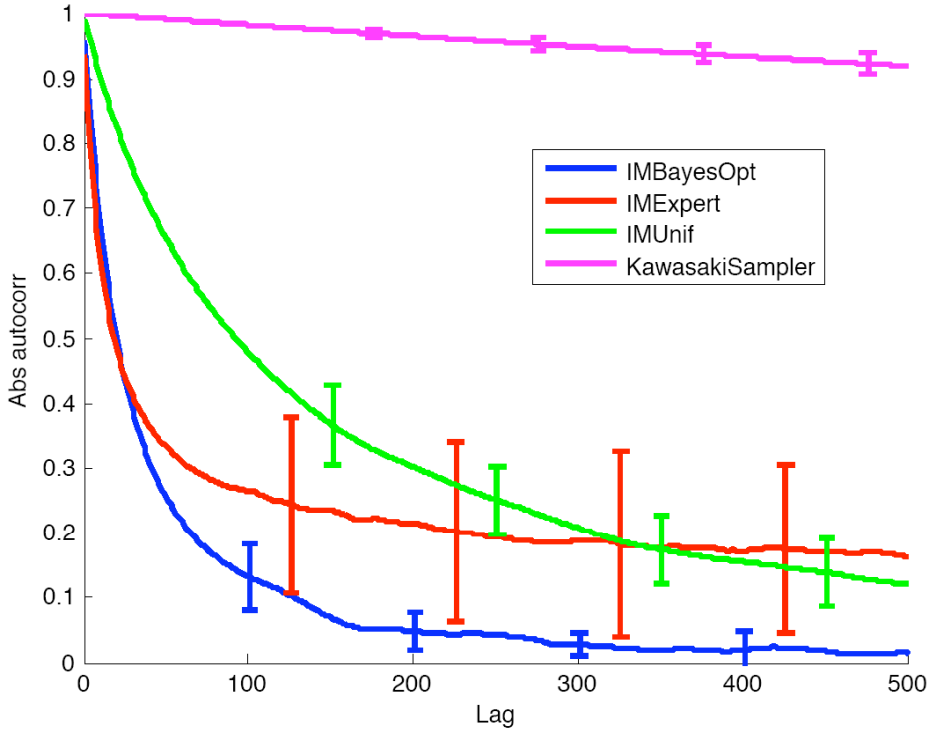


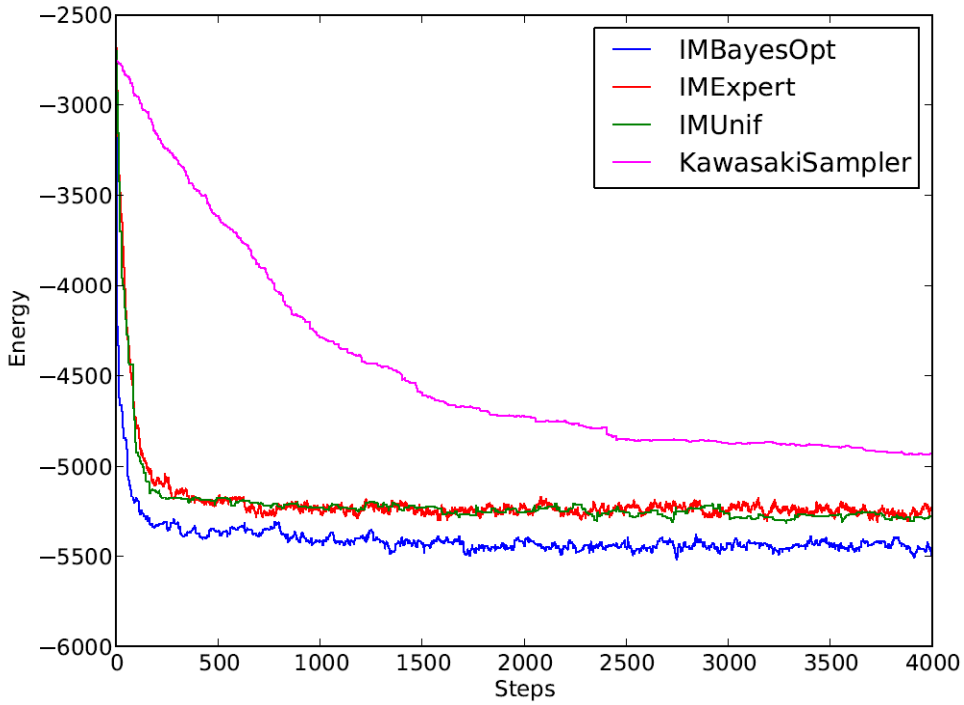


3D Ising model

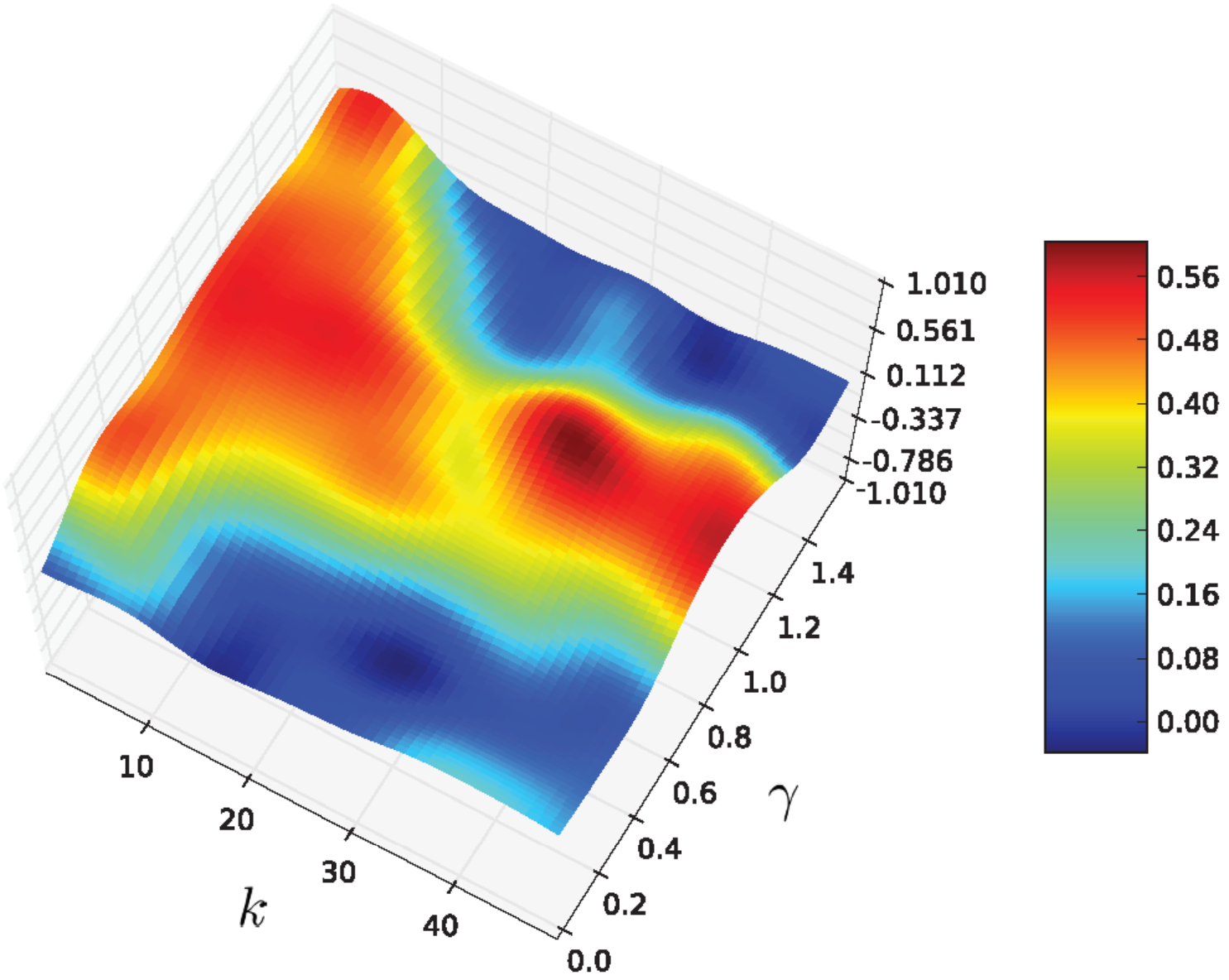


Restricted Boltzmann machine





Restricted Boltzmann machine



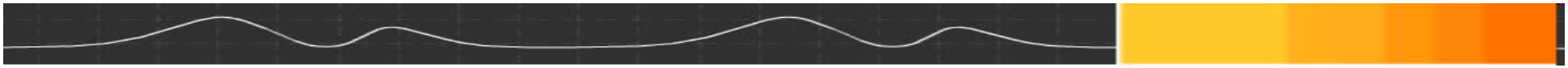
Conclusions & Remarks

1. Bayesian optimization
   - Could use X-armed bandits or parametric bandits instead (Munos, Svepesvari, Cappe, …) for vanishing, infinite adaptation.
   - Convergence analysis with Markov chains would then be needed.
   - Not always a competitor for stochastic approximation, but in cases where we have a few discrete and continuous parameters and where the objective is non-differentiable or expensive, it does better.

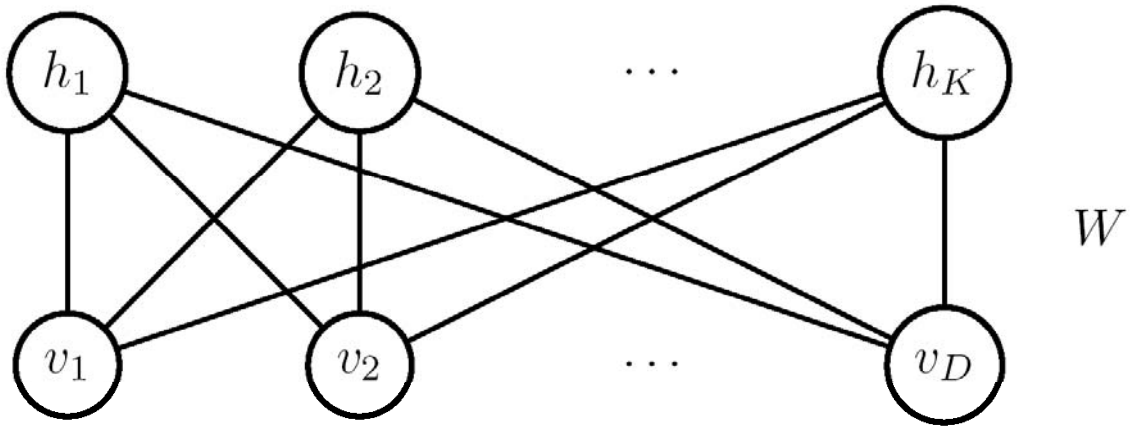
2. Cost function
   - Not clear what the best cost functions should be!

3. Other optimization schemes
   - Fixed learning rates & averaging.
   - Second order methods with Conjugate gradient and LBFGS.

4. Compactness
   - Not unrealistic assumption, specially with projection in mind.



Thank you



Given a dataset $\mathbf{v_N} = \{v_1, v_2, ...v_N\}$ where each data point v_n is a D dimensional vector, we wish to learn a parameterized probabilistic model to reveal structures in the data.

The K-dimensional latent vectors $\{h_1, h_2, ..h_N\}$ can be used in place of the data for classification, denoising, semantic hashing, and more.

$$p(v, h|W) = \frac{1}{Z(W)} \exp\left(-E(v, h, W)\right),$$

$E(v, h, W)$ is the energy function and $Z(W)$ is a normalizing term:

$$Z(W) = \sum_{v' \in V} \sum_{h' \in H} p(v', h'|W).$$

In the binary case where $V = \{0,1\}^D$ and $H = \{0,1\}^K$ the energy function can be expressed as:

$$E(v,h,W) = -\sum_{i=1}^{D}\sum_{j=1}^{K} v_i W_{ij} h_j - \sum_{i=1}^{D} v_i c_i - \sum_{j=1}^{K} h_j b_j.$$

The conditionals can be easily obtained:

$$p(v_i = 1|h,W) = sigm\left(\sum_{j=1}^{K} W_{ij} h_j\right)$$

$$p(h_j = 1|v,W) = sigm\left(\sum_{i=1}^{D} W_{ij} v_i\right),$$

where $sigm(a) = \frac{1}{1+exp(-a)}$. The model is therefore ideal for block Gibbs sampling.

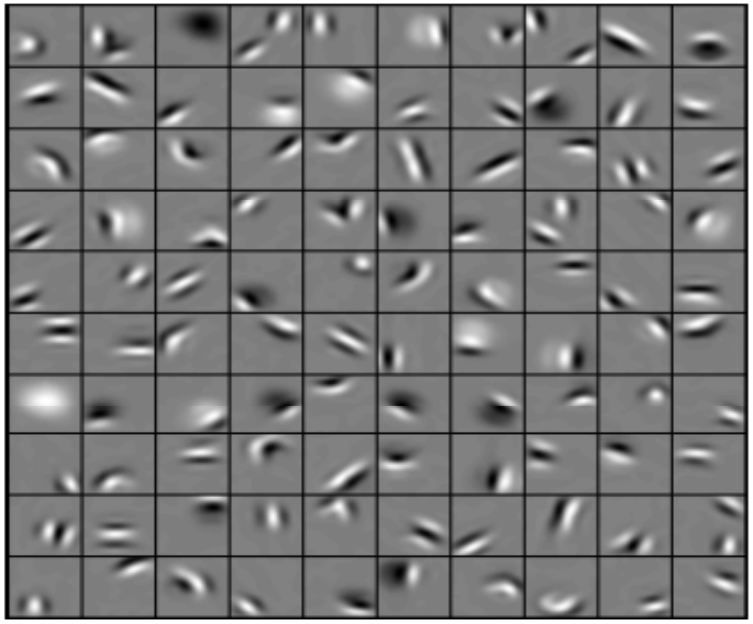
Stochastic maximum likelihood algorithm

1. Set $t = 1$, and $\widetilde{h_n}^{(0)}$ to be a random K-dimensional binary vector.
2. Sample $\widetilde{v_n}^{(t)}$ from $p(v|\widetilde{h_n}^{(t-1)}, W^{(t)})$, and $\widetilde{h_n}^{(t)}$ from $p(h|\widetilde{v_n}^{(t)}, W^{(t)})$.
3. Update the parameters:

$$W_{dk}^{(t+1)} = W_{dk}^{(t)} - \eta^{(t)} \left[-\frac{1}{N} \sum_{n=1}^{N} v_{dn} E[h_k | v_n, W^{(t)}] + \frac{1}{N} \sum_{n=1}^{N} \widetilde{v_{dn}}^{(t)} E[h_k | \widetilde{v_{dn}}^{(t)}, W^{(t)}] \right]$$

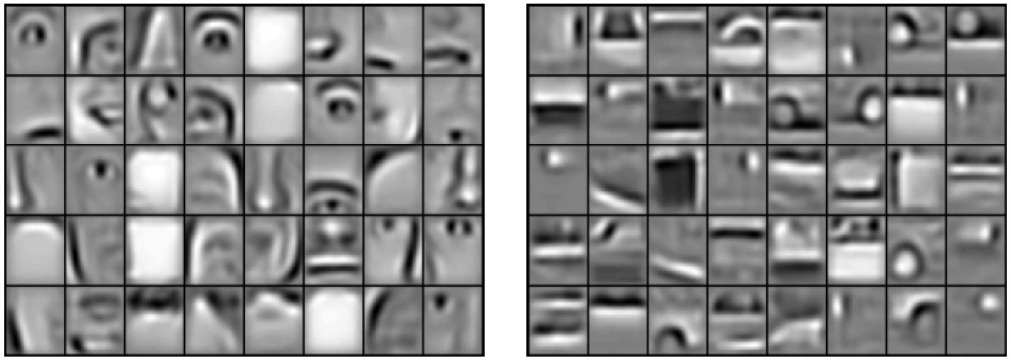
4. Increase t to $t + 1$ and go to step 2.

Layer 1

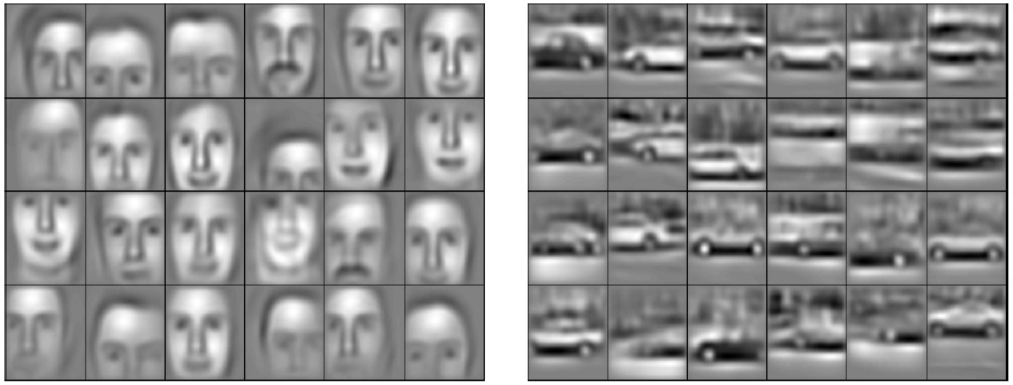


faces cars

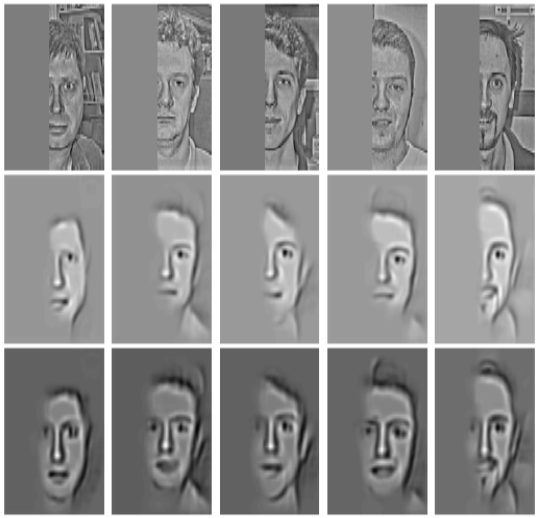
Layer 2



Layer 3



Completing scenes



[Honglak Lee et al 2009]