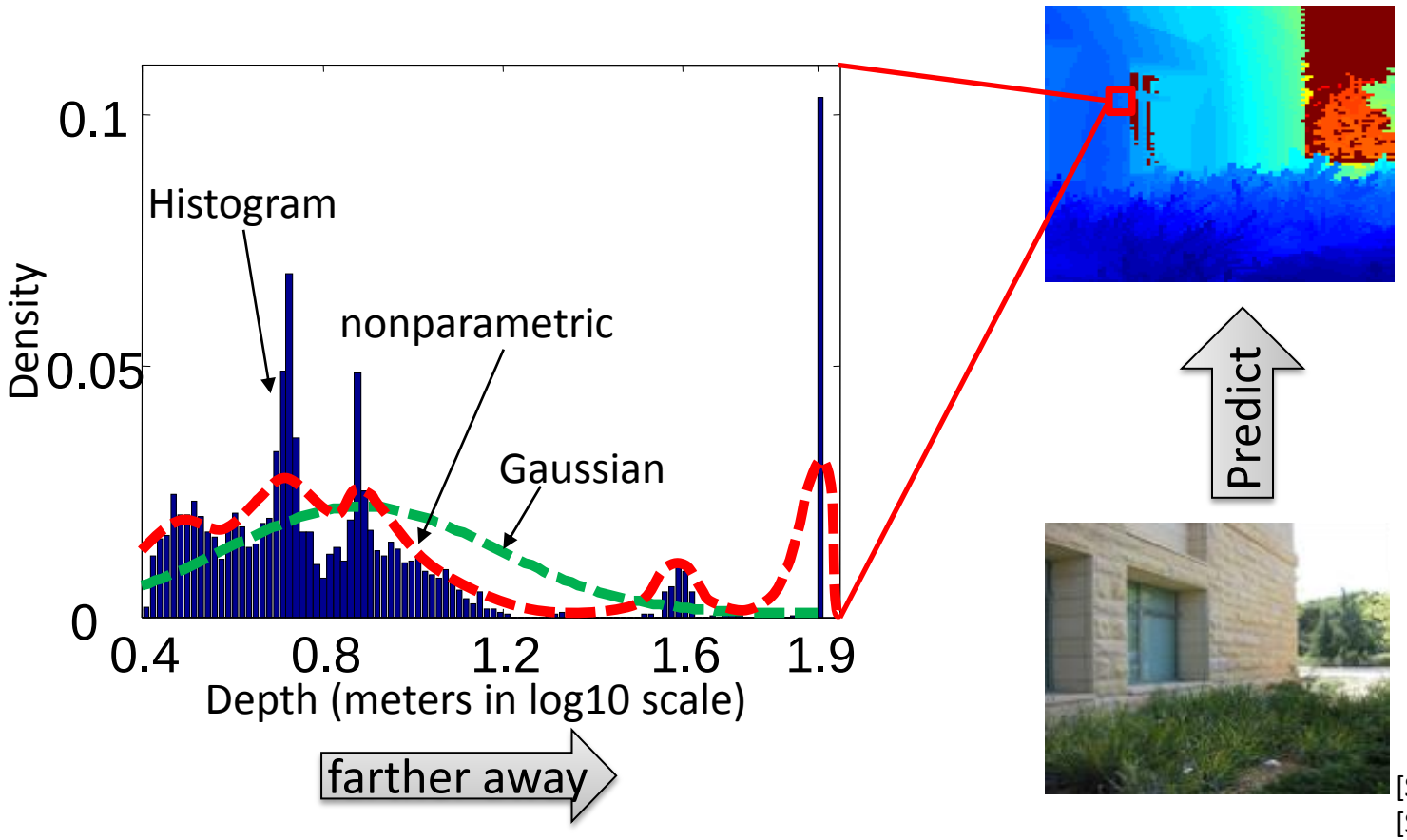
Scalable Kernel Methods for Latent Variable Models

Le Song

College of Computing
Georgia Institute of Technology

Depth prediction from still images

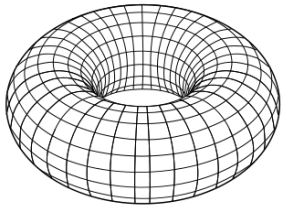
Distribution of depth: a mixture over R

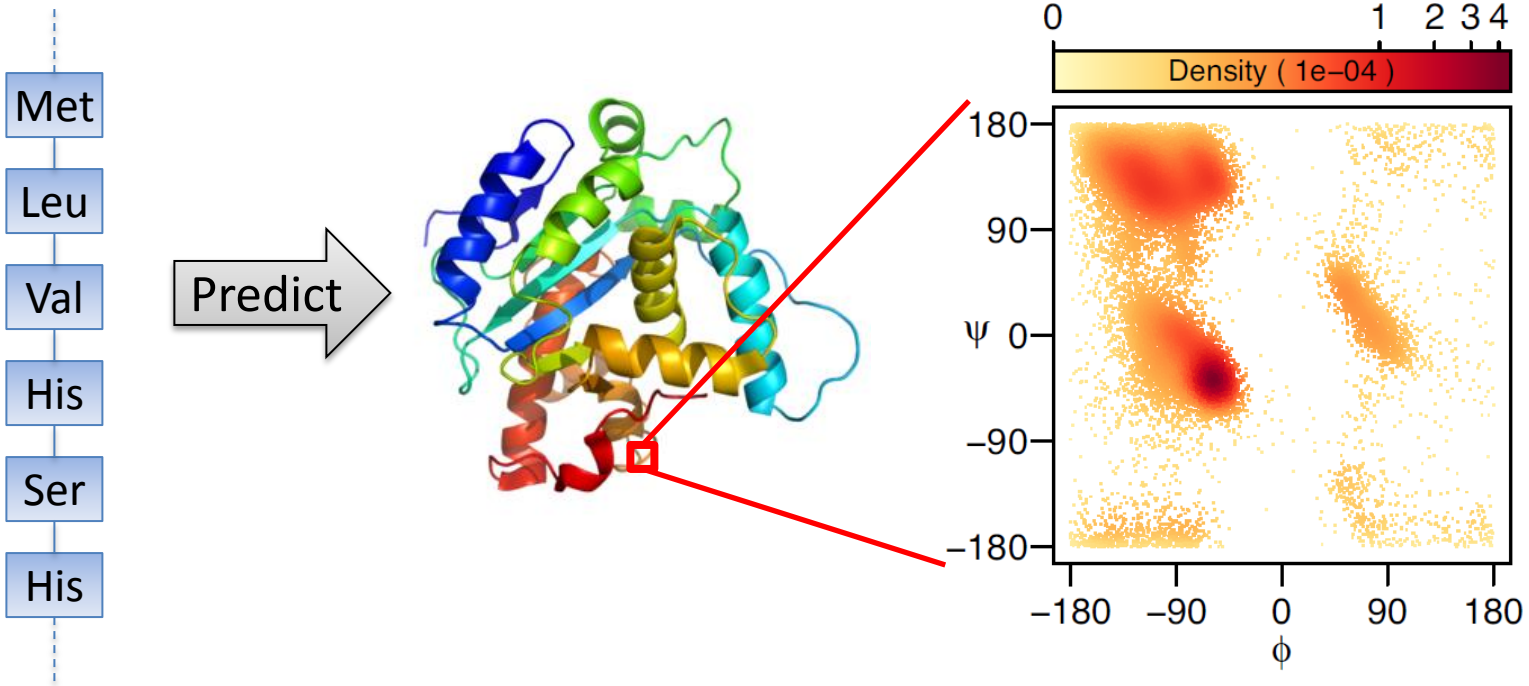


[Saxena05]
[Song11]

Protein structure prediction

Distribution of angle pair (ϕ, ψ): a mixture over torus





Amino-acid sequence

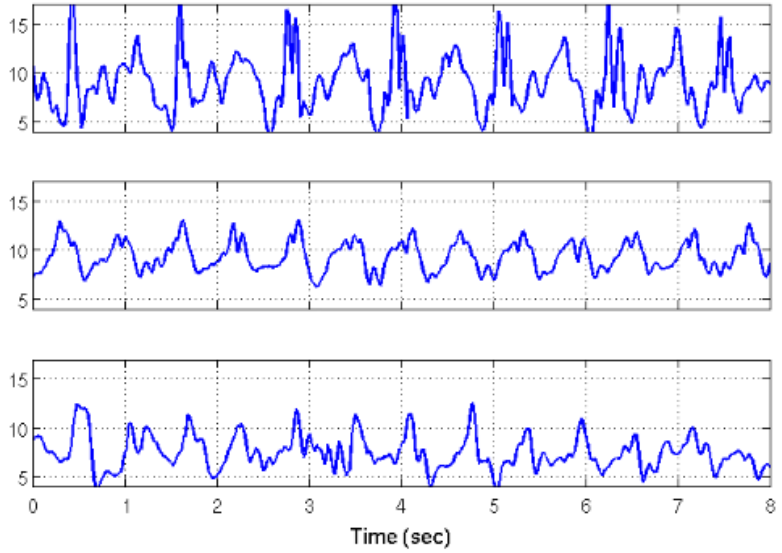
3D structure

Ramachandran plot

[Boomsma08]
[Song11]

Inertia sensor time series prediction

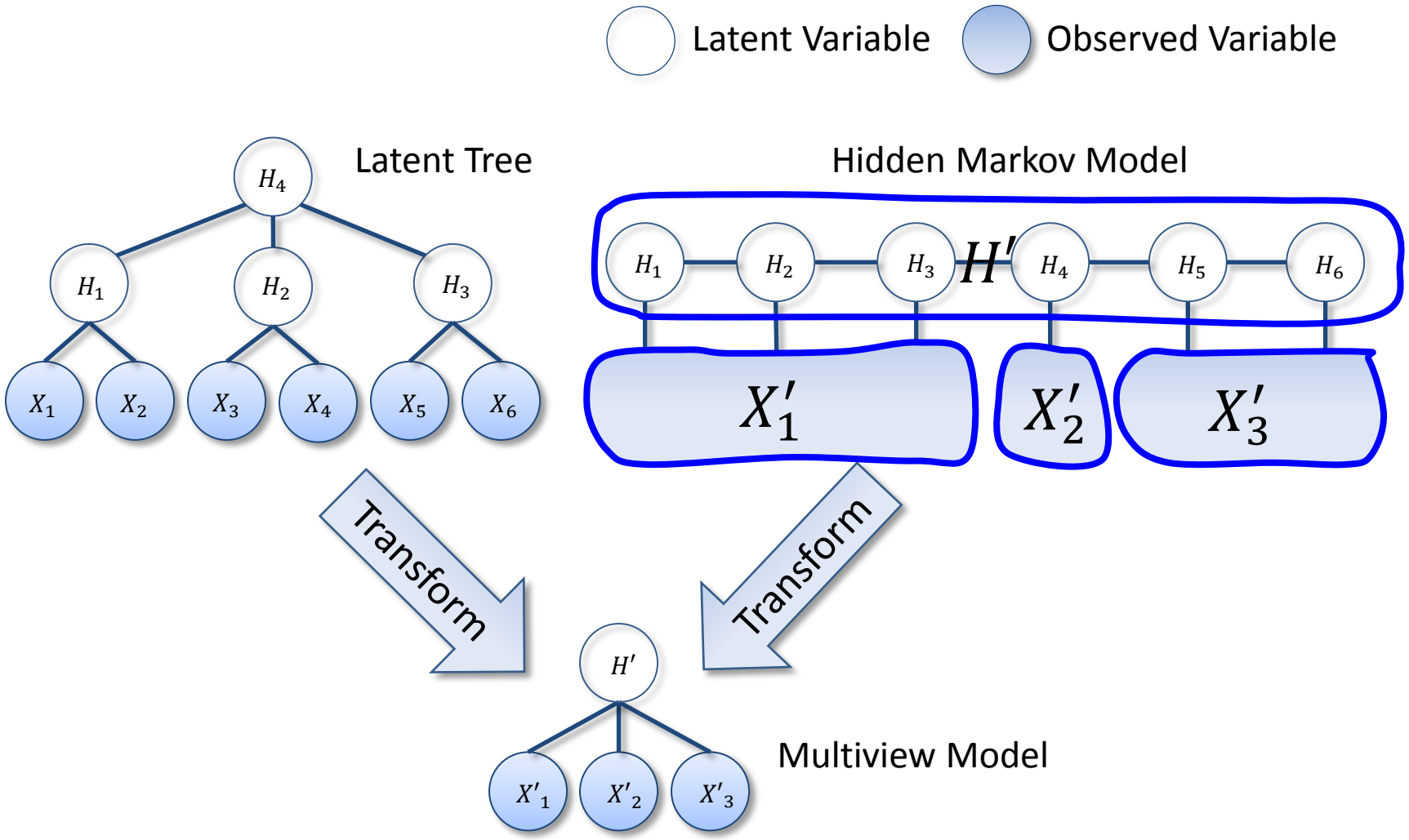
Distribution of 3d acceleration data: a mixture over R^3



[Song10]

Latent variable models (LVMs)

Hypothesizing observed variables are related via hidden variables



Representation and learning of LVM

Likelihood predominantly parametric (may not fit the data)

Learning typically via EM (no provable guarantee)

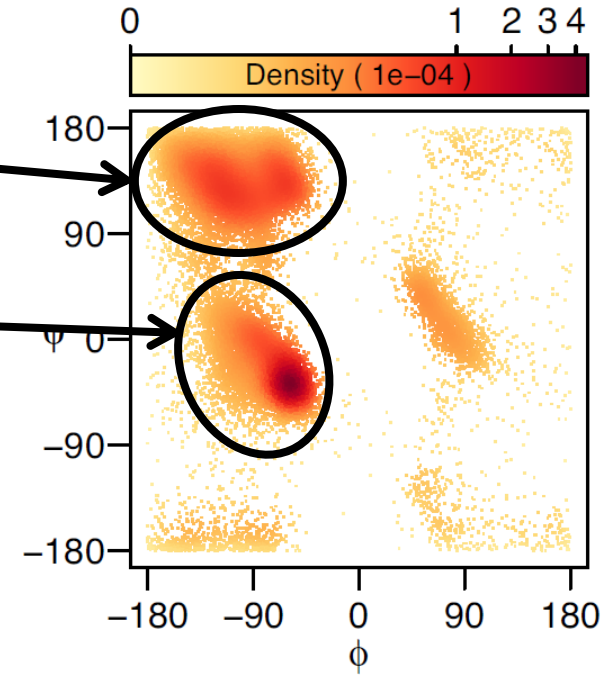
Latent parameters

Likelihood 1
$P(\phi, \psi | H = 1)$

Likelihood 2
$P(\phi, \psi | H = 2)$

$\vdots$

Prior
$P(H)$



Marginal

$$P(\phi, \psi) = \sum_{H} P(\phi, \psi | H) P(H)$$

Spectral algorithms

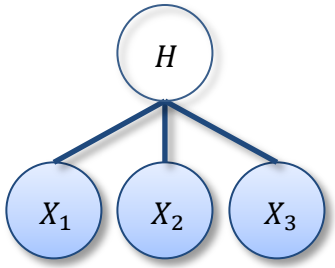
- Use matrix and tensor decompositions to learn latent variable models provably
  - Marginals
    - HMMs, tree and junction trees [Hsu09,Bailly09,Parikh11,12,Foster12]
    - Language modeling [Cohen12, Balle12, Parikh13]
  - Structures
    - Recursive grouping [Choi11]
    - Quartet tests [Anandkumar11,Ishteva13]
  - Latent parameters
    - PCA approach [Mossel06]
    - SVD and tensor approach [Anandkumar12,Song13]
- Limited to discrete and Gaussian likelihoods

Factorization of multi-view model

Joint probability ($X_i \in \{1, \dots, d\}, H \in \{1, \dots, r\}$)

$$P(X_1, X_2) = \sum_{h=1}^{r} P(X_1|h)P(X_2|h)P(h)$$

$$P(X_1, X_2, X_3) = \sum_{h=1}^{r} P(X_1|h)P(X_2|h)P(X_3|h)P(h)$$



Joint probability table ($\pi_h := P(h), \mu_{X|h} := P(X|h) = \mathbb{E}_{X|h}[\phi(X)]$)

$$\boldsymbol{\mathcal{C}}_{X_1X_2} := \mathbb{E}_{X_1X_2}[\phi(X_1) \otimes \phi(X_2)]$$

$$= \sum_{h=1}^{r} \pi_h \cdot \mu_{X_1|h} \otimes \mu_{X_2|h}$$

$$\boldsymbol{\mathcal{C}}_{X_1X_2X_3} := \mathbb{E}_{X_1X_2}[\phi(X_1) \otimes \phi(X_2) \otimes \phi(X_3)]$$

$$= \sum_{h=1}^{r} \pi_h \cdot \mu_{X_1|h} \otimes \mu_{X_2|h} \otimes \mu_{X_3|h}$$

$$\phi(1) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\phi(2) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\vdots$$

Low rank structure

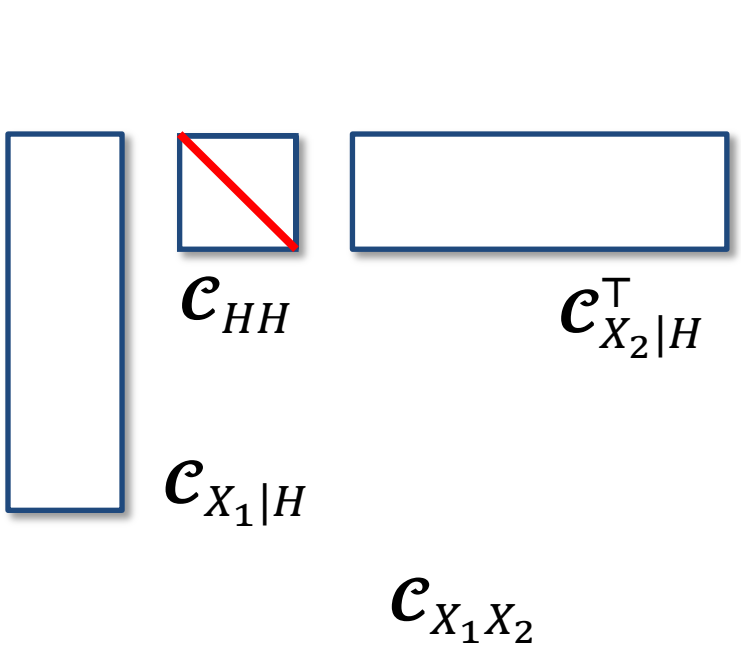
Let $\mathcal{C}_{X|H} \coloneqq (\mu_{X|h=1}, \mu_{X|h=2}, \cdots, \mu_{X|h=r})$

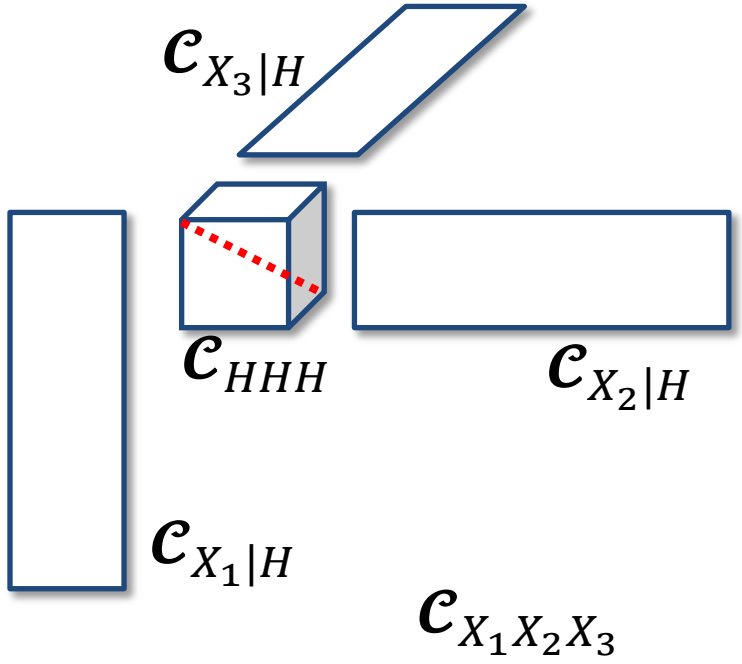
$$\mathcal{C}_{X_1X_2} = \mathcal{C}_{X_1|H}\mathcal{C}_{HH}\mathcal{C}_{X_2|H}^{\top}$$

$$(\mathcal{C}_{HH} \coloneqq \mathrm{diag}(\pi_h))$$

$$\mathcal{C}_{X_1X_2X_3} = \mathcal{C}_{HHH} \circ_1 \mathcal{C}_{X_1|H} \circ_2 \mathcal{C}_{X_2|H} \circ_2 \mathcal{C}_{X_3|H}$$

$$(\mathcal{C}_{HHH} \coloneqq \mathrm{tdiag}(\pi_h))$$





Recovering latent parameters

Given: $\mathcal{C}_{X_1X_2}$ and $\mathcal{C}_{X_1X_2X_3}$ and relation

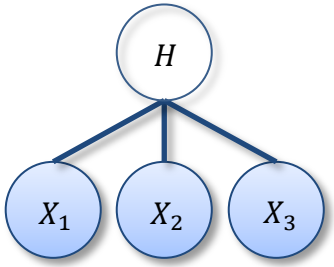
- $\mathcal{C}_{X_1X_2} = \mathcal{C}_{X_1|H}\mathcal{C}_{HH}\mathcal{C}_{X_2|H}^{\top}$
- $\mathcal{C}_{X_1X_2X_3} = \mathcal{C}_{HHH} \circ_1 \mathcal{C}_{X_1|H} \circ_2 \mathcal{C}_{X_2|H} \circ_2 \mathcal{C}_{X_3|H}$

Want: recover parameters $\mathcal{C}_{X_1|H}$, $\mathcal{C}_{X_2|H}$, $\mathcal{C}_{X_3|H}$ and $\mathcal{C}_{HH}$

For simplicity, assume: $\mathcal{C}_{X_1|H}$, $\mathcal{C}_{X_2|H}$, $\mathcal{C}_{X_3|H}$ are the same

Spectral Algorithm outline:

- Use $\mathcal{C}_{X_1X_2}$ to obtain a whitening operator
- Whiten the 3rd order tensor $\mathcal{C}_{X_1X_2X_3}$
- Run orthogonal tensor decomposition
- Unwhiten to recover parameters



Step 1: Obtain whitening matrix

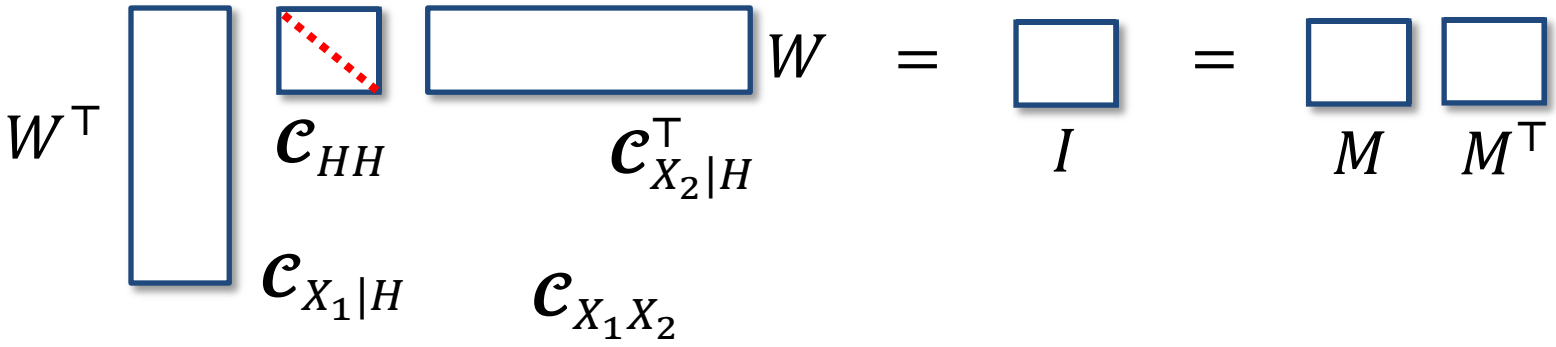
Perform eigendecomposition of $\boldsymbol{\mathcal{C}}_{X_1X_2}$

$$\boldsymbol{\mathcal{C}}_{X_1X_2} = \boldsymbol{\mathcal{C}}_{X_1|H}\boldsymbol{\mathcal{C}}_{HH}\boldsymbol{\mathcal{C}}_{X_2|H}^{\top} = U_r S_r U_r^{\top}$$

Define whitening matrix $W := U_r S_r^{-1/2}$

$$W^{\top}\boldsymbol{\mathcal{C}}_{X_1X_2}W = I = \underbrace{W^{\top}\boldsymbol{\mathcal{C}}_{X_1|H}\boldsymbol{\mathcal{C}}_{HH}^{1/2}}_{M}\underbrace{\boldsymbol{\mathcal{C}}_{HH}^{1/2}\boldsymbol{\mathcal{C}}_{X_2|H}^{\top}W}_{M^{\top}}$$

$r \times r$ orthogonal matrix

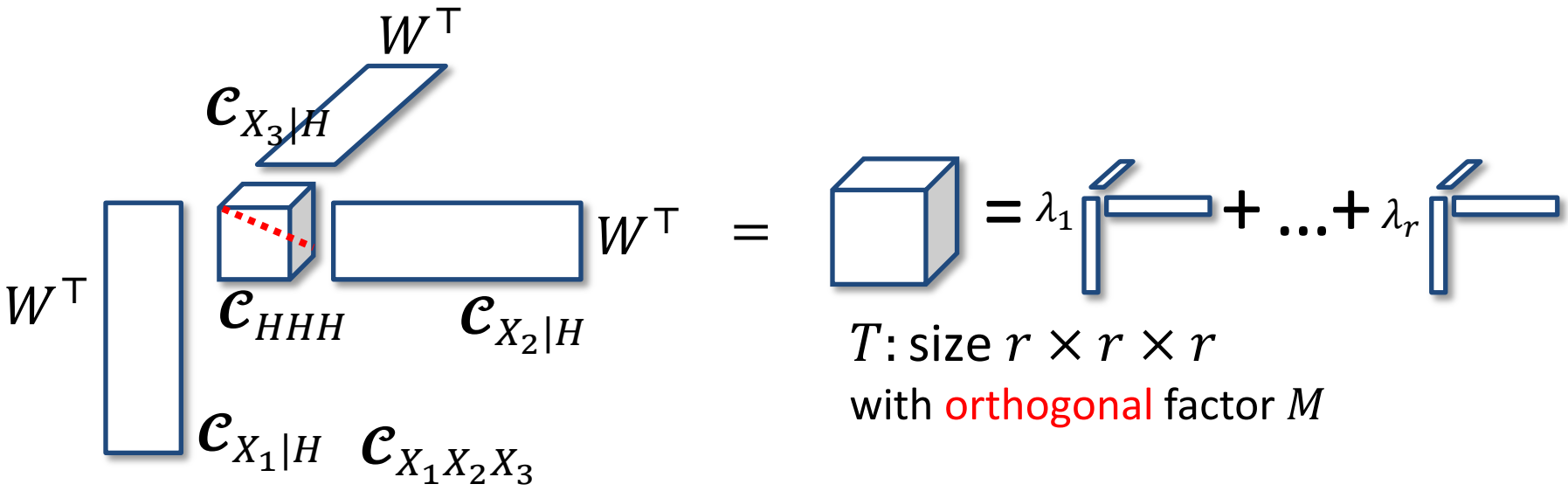


Step 2: whiten 3rd order tensor

Whiten tensor $\mathcal{C}_{X_1X_2X_3} = \mathcal{C}_{HHH} \circ_1 \mathcal{C}_{X_1|H} \circ_2 \mathcal{C}_{X_2|H} \circ_3 \mathcal{C}_{X_3|H}$

$$T = \mathcal{C}_{X_1X_2X_3} \circ_1 W^\top \circ_2 W^\top \circ_3 W^\top$$

$$= \mathcal{C}_{HHH}^{-1/2} \circ_1 \underbrace{\left(W^\top \mathcal{C}_{X_1|H} \mathcal{C}_{HH}^{1/2}\right)}_{M} \circ_2 \underbrace{\left(W^\top \mathcal{C}_{X_2|H} \mathcal{C}_{HH}^{1/2}\right)}_{M} \circ_3 \underbrace{\left(W^\top \mathcal{C}_{X_3|H} \mathcal{C}_{HH}^{1/2}\right)}_{M}$$



T: size $r \times r \times r$

with orthogonal factor M

Step 3: tensor power method

Find eigenvector of T. Power method: iterate till convergence [Anima13]

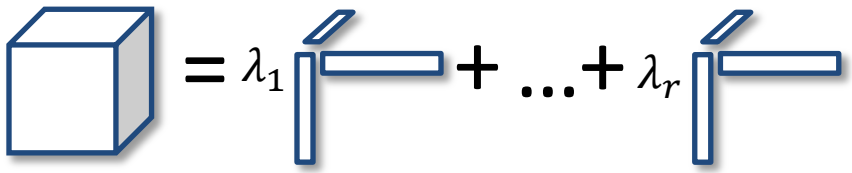
$$v \leftarrow T \circ_1 I \circ_2 v \circ_3 v$$

$$v \leftarrow \frac{v}{\|v\|}$$

Eigenvalue: $\lambda = T \circ_1 v \circ_2 v \circ_3 v = \pi_h^{-1/2}$

Deflate $T \leftarrow T - \lambda * v \otimes v \otimes v$ and find the next v

$$M = (v_1, v_2, \ldots, v_r)$$



T: size $r \times r \times r$

Step 4: recover latent parameters

Prior over latent variable

$$\pi_i := P(h = i) = \frac{1}{\lambda_i^2}$$

Remember: whitening matrix $W := U_r S_r^{-1/2}$ from $\boldsymbol{\mathcal{C}}_{X_1 X_2}$

$$W^\top \boldsymbol{\mathcal{C}}_{X_1 X_2} W = I = W^\top \boldsymbol{\mathcal{C}}_{X_1|H} \boldsymbol{\mathcal{C}}_{HH}^{1/2} C_{HH}^{1/2} C_{X_2|H}^\top W = M M^\top$$

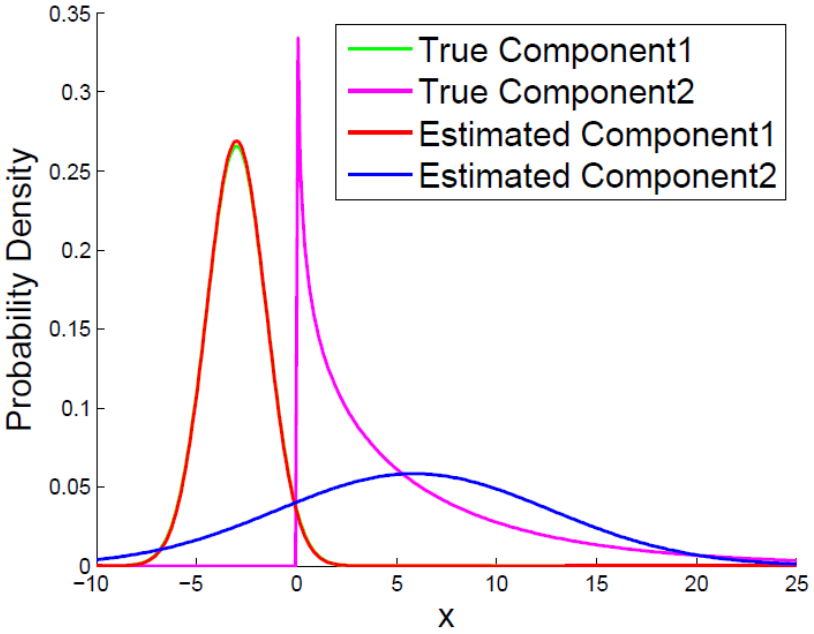
Undo whitening

$$\boldsymbol{\mathcal{C}}_{X|H} = \left(\mu_{X|h=1}, \mu_{X|h=2}, \cdots, \mu_{X|h=r}\right)$$

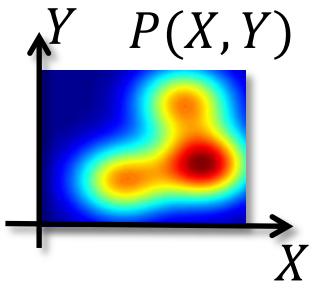
$$= W^\dagger M \; diag\left(\pi_i^{-1/2}\right)$$

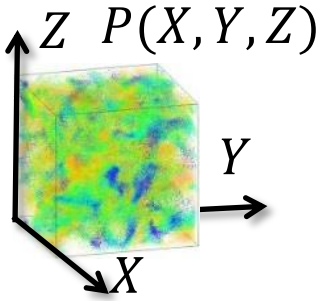
Non-discrete, non-Gaussian case?

Real world data: continuous, multimodal, skew, complex



(a) EM using Mixture of Gaussians Model





Kernel embedding of distributions [Smola07]

$$k(x,x') = \langle k(x,\cdot), \phi(x',\cdot) \rangle_{\mathcal{F}} = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$$

$$\mu_X = \mathbb{E}_{X \sim P(X)}[\phi(X)]$$

Kernel embedding and covariance operator [Song13]

Georgia Tech

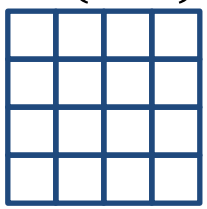
Discrete Eg.

1	0	0
0	1	0
0	0	1

$P(X)$

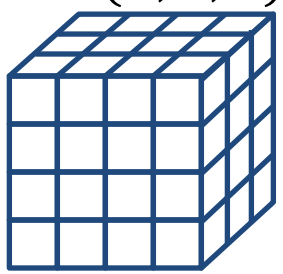
$d \times 1$

$P(X,Y)$



$d \times d$

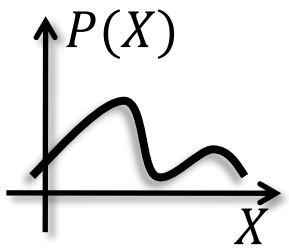
$P(X,Y,Z)$



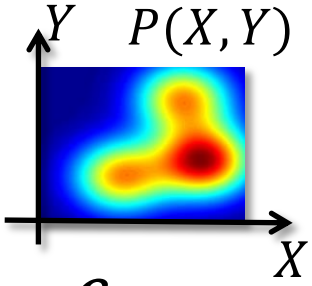
$d \times d \times d$

Kernel Embedding Eg.

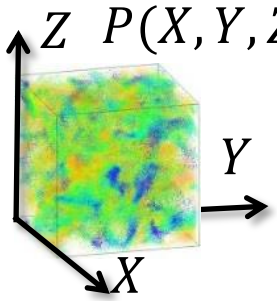
$\exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$



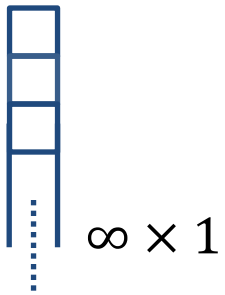
$\mu_X := \mathbb{E}_X[\phi(X)]$



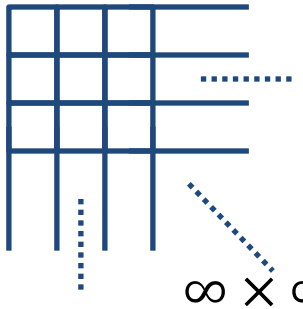
$\mathcal{C}_{XY} := \mathbb{E}_{XY}[\phi(X) \otimes \phi(Y)]$



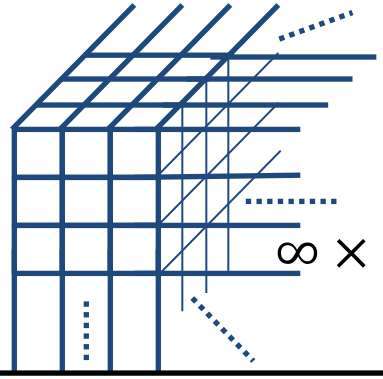
$\mathcal{C}_{XYZ} := \mathbb{E}_{XYZ}[\phi(X) \otimes \phi(Y) \otimes \phi(Z)]$



$\infty \times 1$

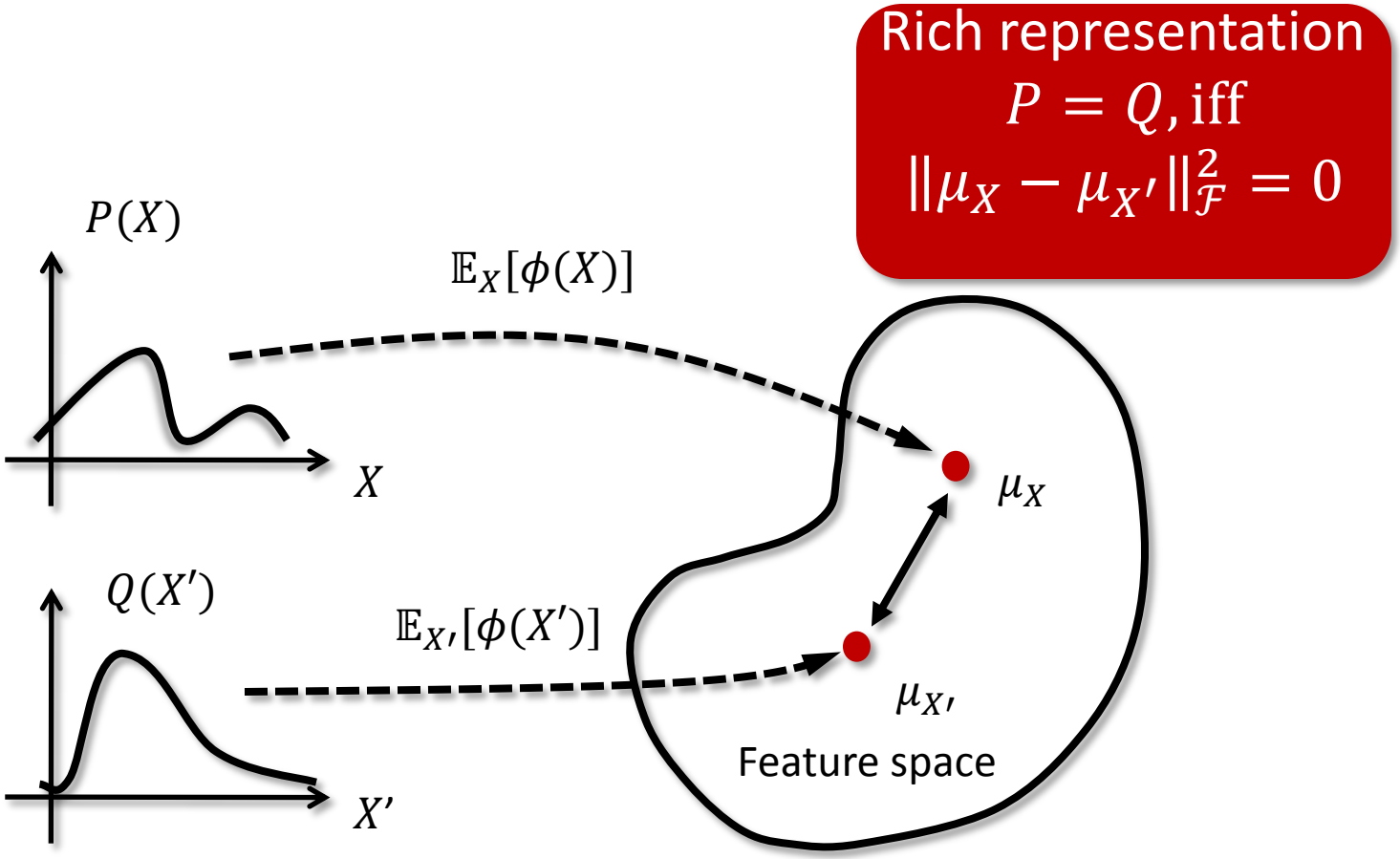


$\infty \times \infty$



$\infty \times \infty \times \infty$

Represent and compare distributions



Applications: two-sample test, independence measure, change point detection, domain adaptation, feature selection, dimentionality reduction, belief propagation

Kernel spectral algorithm for LVMs [Song14]

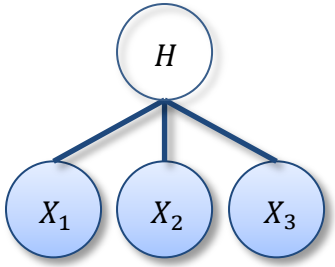
Given m observation $\{(x_1^i, x_2^i, x_3^i)\}_{i\in[m]}$ drawn iid from a multi-view model $p(X_1, X_2, X_3)$

Step 1: feature matrix with kernel $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$

$$\Phi := \left(\phi(x_1^1), \ldots, \phi(x_1^m), \phi(x_2^1), \ldots, \phi(x_2^m)\right), K = \Phi^\top \Phi$$
$$\Psi := \left(\phi(x_2^1), \ldots, \phi(x_2^m), \phi(x_1^1), \ldots, \phi(x_1^m)\right), L = \Psi^\top \Psi$$

Empirical covariance: $\hat{C}_{X_1X_2} := \frac{1}{2m}\Phi\Psi^\top$

Need eigendecomposition of $\hat{C}_{X_1X_2}$ (**kernel PCA**)



Other steps more or less the same

Conditional density estimation

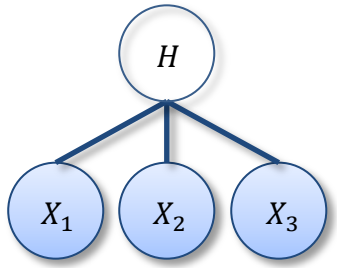
Use Gaussian RBF kernel

$$k(x, x') = \frac{1}{(2\pi)^{d/2} s^d} \exp\left(-\frac{\|x - x'\|^2}{2s^2}\right)$$

Recovered $\widehat{\mathcal{C}}_{X|H} := (\hat{\mu}_{X|h=1}, \ldots, \hat{\mu}_{X|h=r})$ using spectral algorithm

Estimate of $\hat{p}(X_1|H)$ obtained by inner product

$$\hat{p}(x_1|H) = \left\langle k(x_1, \cdot), \hat{\mu}_{X_1|H} \right\rangle_{\mathcal{F}}$$



Estimate conditional density $p(X_1|H)$ **without samples for H**

Consistency and sample complexity

Combine concentration inequalities for kernel embeddings and analysis of tensor power method

$$Let\ \pi_{min} \coloneqq \min_{i\in[r]} \pi_i\ , \theta \coloneqq \max\left(\frac{r^2}{\sigma_r(\boldsymbol{\mathcal{C}}_{X_1X_2})}, \frac{r^{2/3}}{\pi_{min}^{1/3}}\right), \text{ if}$$

$$m > O\left(\frac{\theta \log\frac{\delta}{2}}{\sigma_r(\boldsymbol{\mathcal{C}}_{X_1X_2})}\right)$$

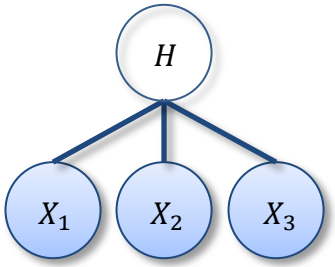
and run tensor power long enough, then with probability $1 - 4\delta$

$$\left|\pi_i^{-1/2} - \hat{\pi}_i^{-1/2}\right| \leq O\left(\frac{\sqrt{\log\frac{\delta}{2}}}{\sqrt{m}\,\sigma_r(\boldsymbol{\mathcal{C}}_{X_1X_2})}\right)$$

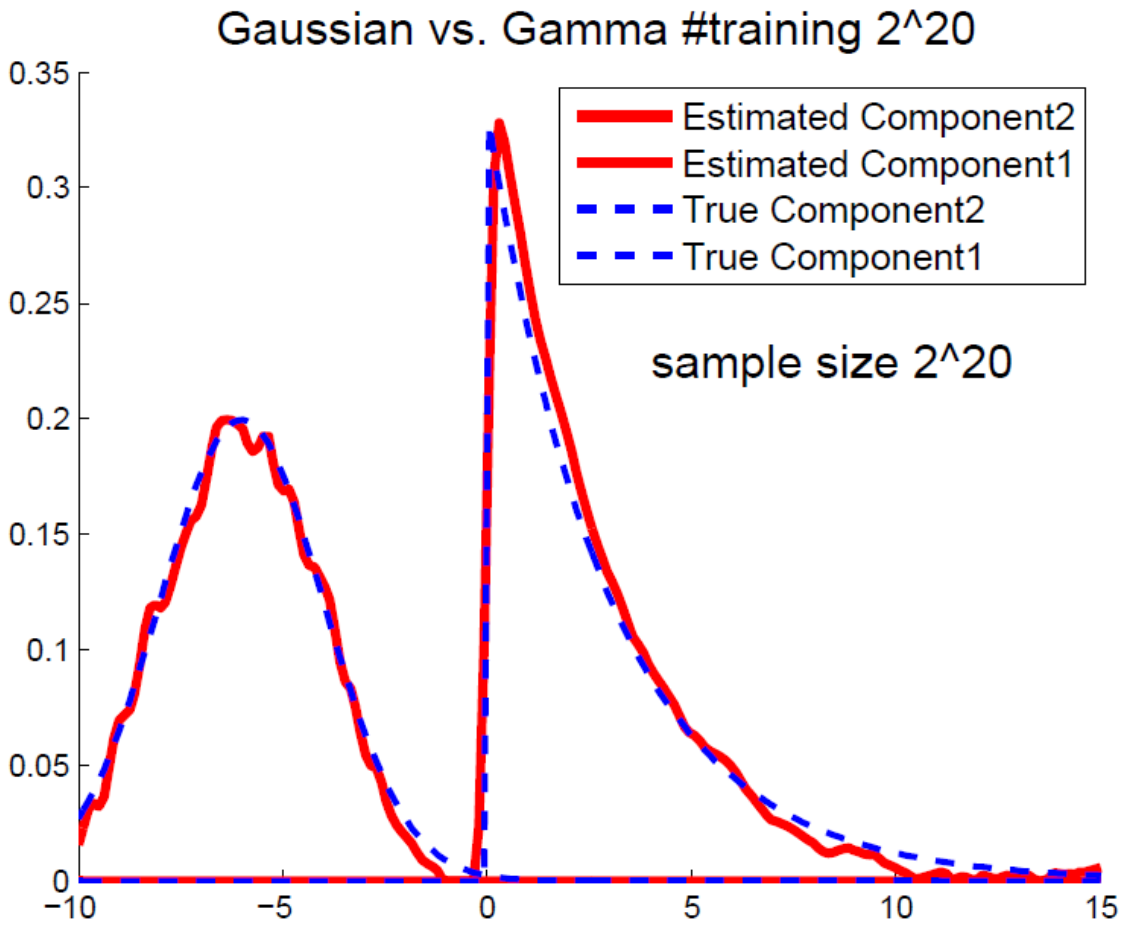
$$\left|\pi_i^{-1/2}\mu_{X|h=i} - W^{\dagger}v_i\right| \leq O\left(\frac{\pi_i^{-1/2}\sqrt{\log\frac{\delta}{2}}}{\sqrt{m}\,\sigma_r(\boldsymbol{\mathcal{C}}_{X_1X_2})}\right)$$

Recovery of conditional density

Multi-view latent variable model



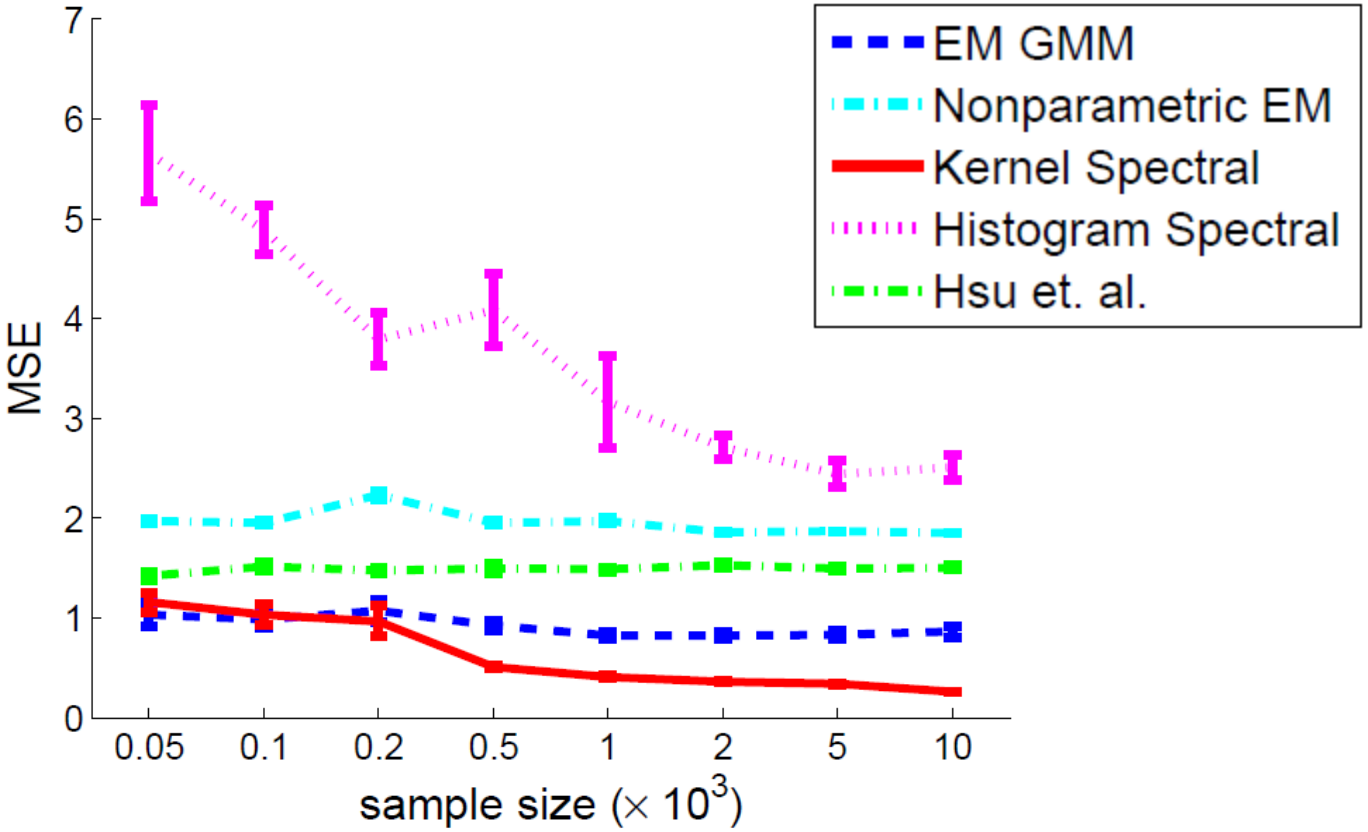
Each view: mixture of Gaussian and shifted Gamma



Effect of sample sizes

Measure the error using prior weighted ℓ_2 difference, $MSE :=$

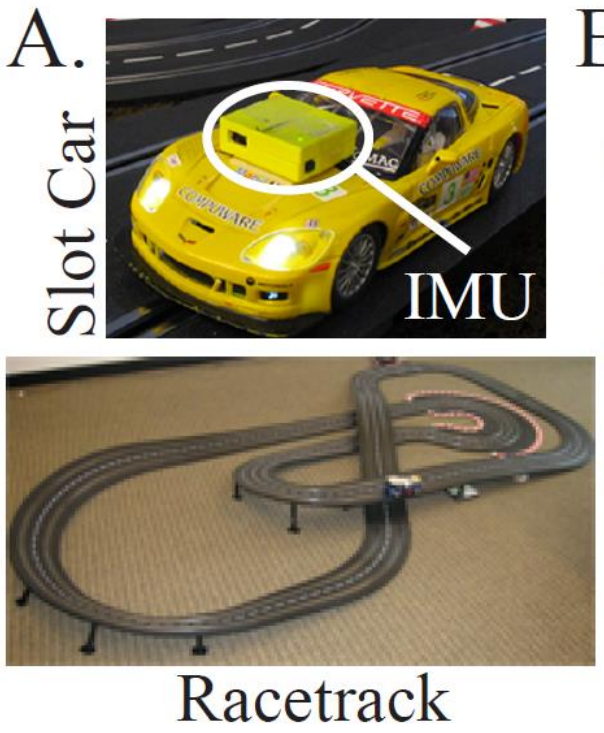
$$\sum_{h=1}^{k} \pi_h \sqrt{\sum_{j=1}^{m'} \left(p(x^j|h) - \hat{p}(x^j|h)\right)^2}$$

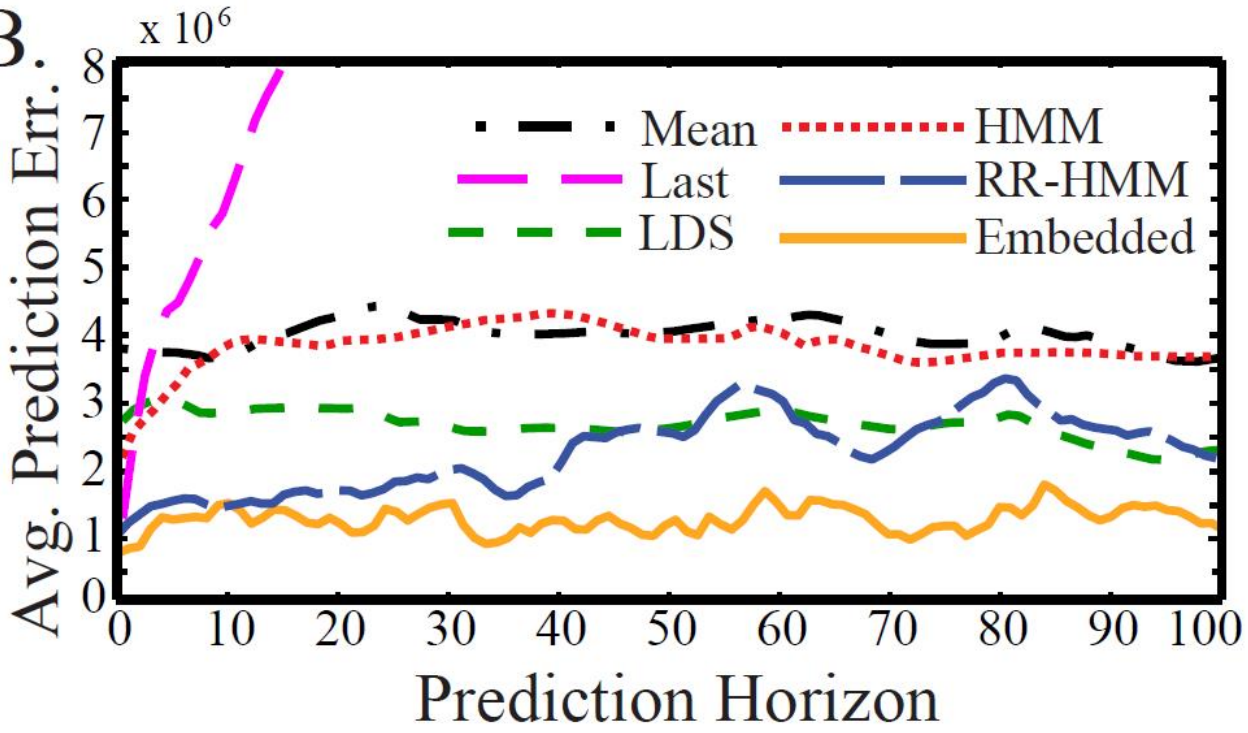


Gaussian/Gamma $k = 3$

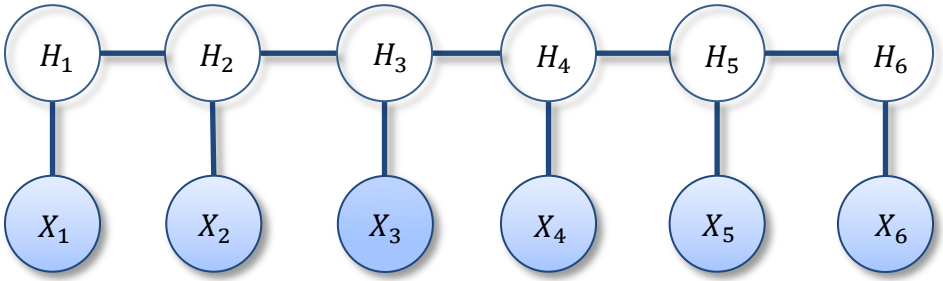
Sensor time series prediction

Predict future sensor readings based on history



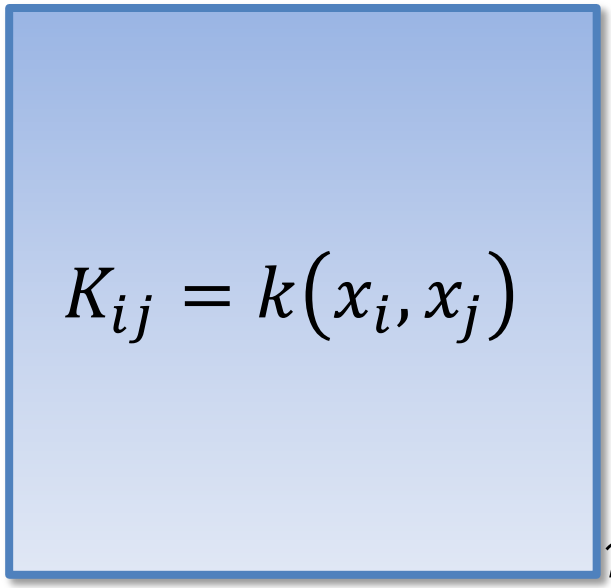


Hidden Markov Model



Bottleneck of kernel spectral algorithm

Storage and computation on the dense kernel matrix, K

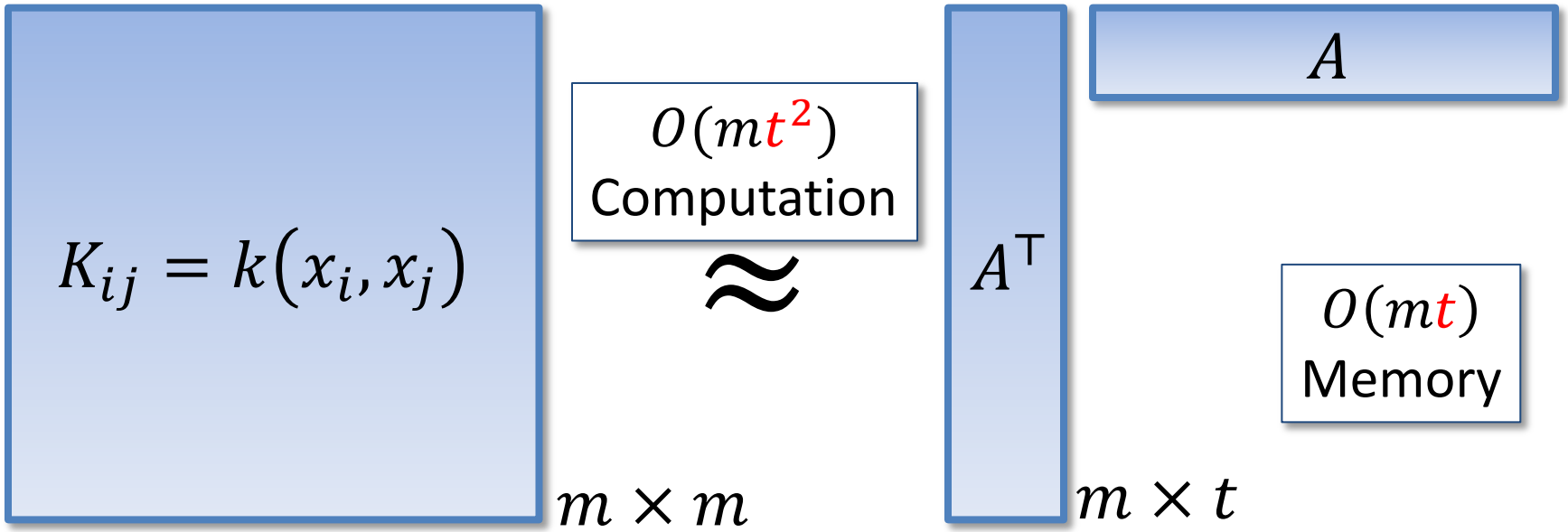


Most costly in spectral algorithm: kernel PCA

- Memory: $O(m^2)$
- Computation for r eigenvectors: $O(m^2 d + m^2 r)$

Low rank decomposition

Nyström's method [Smola00, William'00], incomplete Cholesky decomposition [Fine00]



Solve t dimensional problem afterwards

Generalization bound: $O\left(\frac{1}{\sqrt{t}} + \frac{1}{\sqrt{m}}\right)$

Random features approximation

Duality between kernels and random processes, eg. RBF kernel [Rahimi08, Le13], Polynomial kernels [Pham'13]

$$k(x, x') = \int_{\Omega} \phi_\omega(x)\phi_\omega(x')dP(\omega) = \mathbb{E}_\omega[\phi_\omega(x)\phi_\omega(x')]$$

Sample $\{\omega_i\}_{i=1}^t \sim P(\omega)$, and

$$k(x, x') \approx \frac{1}{t}\sum_{i=1}^{t} \phi_{\omega_i}(x)\phi_{\omega_i}(x')$$

$O(mt \log d)$ Computation

$O(mt)$ Memory

Solve t dimensional problem afterwards

Generalization bound: $O\left(\frac{1}{\sqrt{t}} + \frac{1}{\sqrt{m}}\right)$

Stochastic kernel PCA

Variational characterization of eigenfunctions $U^* = (u^1, \dots, u^r)$

$$U^* = argmax_{U^\top U = I} \frac{1}{2} tr(U^\top \boldsymbol{\mathcal{C}}_{XX} U)$$

Gradient ascent algorithm [Oja85]

$$g_i = \boldsymbol{\mathcal{C}}_{XX} U_i - U_i U_i^\top \boldsymbol{\mathcal{C}}_{XX} U_i$$
$$U_{i+1} = U_i + \gamma_i g_i$$

Stochastic approximation to $\boldsymbol{\mathcal{C}}_{XX}$

$$x_i \sim P(x)$$
$$\boldsymbol{\mathcal{C}}_{XX} \approx \phi(x_i) \otimes \phi(x_i) = k(x_i, \cdot) \otimes k(x_i, \cdot)$$

Stochastic gradient ($y_i := U_i^\top k(x_i, \cdot)$)

$$\hat{g}_i = k(x_i, \cdot) y_i^\top - U_i \, y_i y_i^\top$$

Doubly stochastic PCA

Stochastic gradient ($y_i := U_i^\top k(x_i, \cdot)$): need to store all data points

$$\hat{g}_i = k(x_i, \cdot) y_i^\top - U_i \, y_i y_i^\top$$

Doubly stochastic gradient: make a second stochastic approximation to kernel function using random features

$$x_i \sim P(x), \, \omega_i \sim P(\omega)$$

$$\tilde{g}_i = \phi_{\omega_i}(x_i) \phi_{\omega_i}(\cdot) \, y_i^\top - U_i \, y_i y_i^\top$$

Unbiased, bounded, variance bounded [Bo14]

Doubly stochastic kernel PCA

Eigenfunctions

$$U(\cdot) = (u^1(\cdot), \dots, u^r(\cdot)) = \sum_{i=1}^{t} \phi_{\omega_i}(\cdot)\boldsymbol{\alpha_i}^\top$$

$Train(P(x))$

- For $i = 1, \dots, t$ do
  - $x_i \sim P(x)$
  - $\omega_i \sim P(\omega)$ with seed i
  - $y_i = Evaluate\left(x_i, \{\boldsymbol{\alpha_j}\}_{j=1}^{i}\right)$
  - $\boldsymbol{\alpha_i} = \gamma_i \phi_{\omega_i}(x_i) y_i$
  - $\boldsymbol{\alpha_j} = \left(I - \gamma_i y_i y_i^\top\right)\boldsymbol{\alpha_j}$ for $j < i$
- End for

$Evaluate\left(x_i, \{\boldsymbol{\alpha_j}\}_{j=1}^{i}\right)$

- Set $y = 0$
- For $j = 1, \dots, i$ do
  - $\omega_j \sim P(\omega)$ with seed j
  - $y = y + \phi_{\omega_j}(x_i)\boldsymbol{\alpha_j}$
- End for

Advantage of doubly stochastic gradients

Simplicity and generality: works for other loss functions

Flexibility: works for streaming data

Efficiency: $O(mt \log d)$, inner product between data and random features

Small memory: $O(t)$, no need to remember data points or random features

Theoretical guarantee: $O\left(\frac{1}{t}\right)$ convergence

Strong empirical performance: MNIST digits and ImageNet

Convergence

The error can be decomposed into two terms

$$(u_{t+1}(x) - u_*(x))^2 \leq \underbrace{2(u_{t+1}(x) - h_{t+1}(x))^2}_{\text{Err1: due to random features}} + \underbrace{2\kappa\|h_{t+1} - u_*\|_{\mathcal{F}}^2}_{\text{Err2: due to random data}}$$

Err1: generalize existing analysis

Err2: design a martingale difference sequence

- $u_{t+1}(\cdot) = \sum_{i=1}^{t} \beta_t^i \, \tilde{g}_i(\cdot)$
- $h_{t+1}(\cdot) = \sum_{i=1}^{t} \beta_t^i \, \hat{g}_i(\cdot)$
- $\mathcal{D}^i = \{x_1, \ldots, x_i\}, \boldsymbol{\omega}^i = \{\omega_1, \ldots, \omega_i\}$

$$\mathbb{E}_{\omega_i}\left[\hat{g}_i(\cdot) - \tilde{g}_i(\cdot) | \mathcal{D}^i, \boldsymbol{\omega}^{i-1}\right] = 0$$

Martingale difference

$V_i(x) = \beta_t^i(g_i(x) - \tilde{g}_i(x))$: a martingale difference sequence

Since $|V_i(x)| \leq c_i \leq O\left(\frac{1}{t}\right)$, Azuma's Inequality

$$P\left\{\left|\sum_{i=1}^{t} V_i(x)\right| \geq \epsilon\right\} < 2\exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^{t} c_i^2}\right)$$

Equivalently

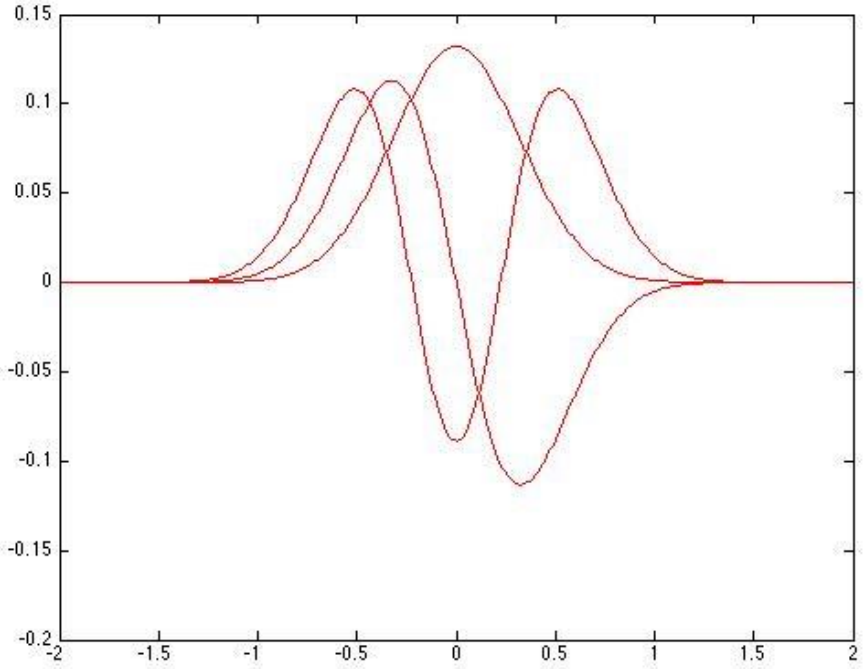
$$\mathbb{E}_{\mathcal{D}^t,\boldsymbol{\omega}^t}[(u_{t+1}(x) - h_{t+1}(x))^2] = \mathbb{E}_{\mathcal{D}^t,\boldsymbol{\omega}^t}\left[\left(\sum_{i=1}^{t} V_i(x)\right)^2\right]$$

$$\leq \sum_{i=1}^{t} c_i^2 \leq O\left(\frac{1}{t}\right)$$

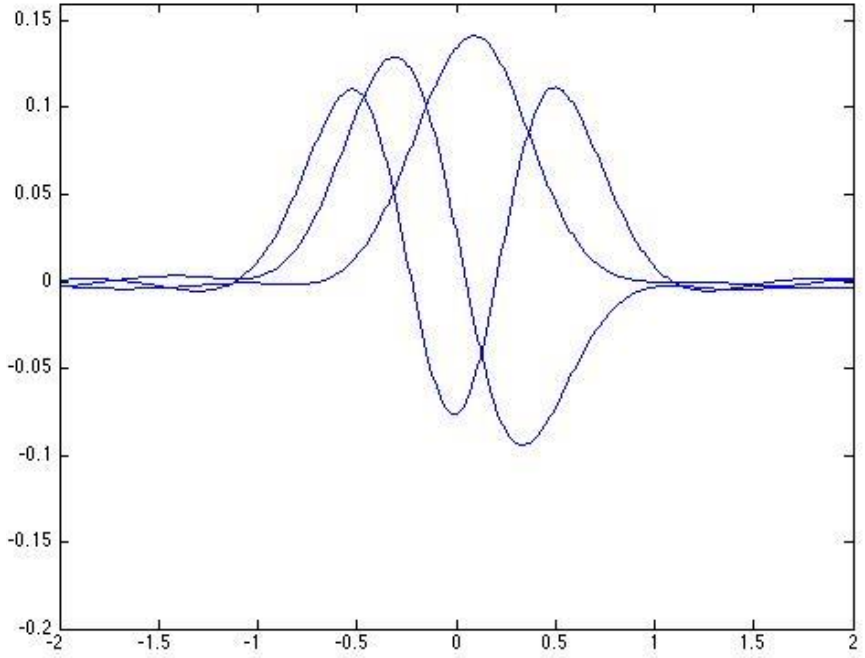
Estimate eigenfunctions

Gaussian distribution with Gaussian RBF kernel [Williams00]

Closed form eigenfunctions available



Leading 3 eigenfunctions

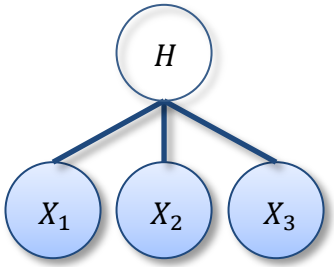


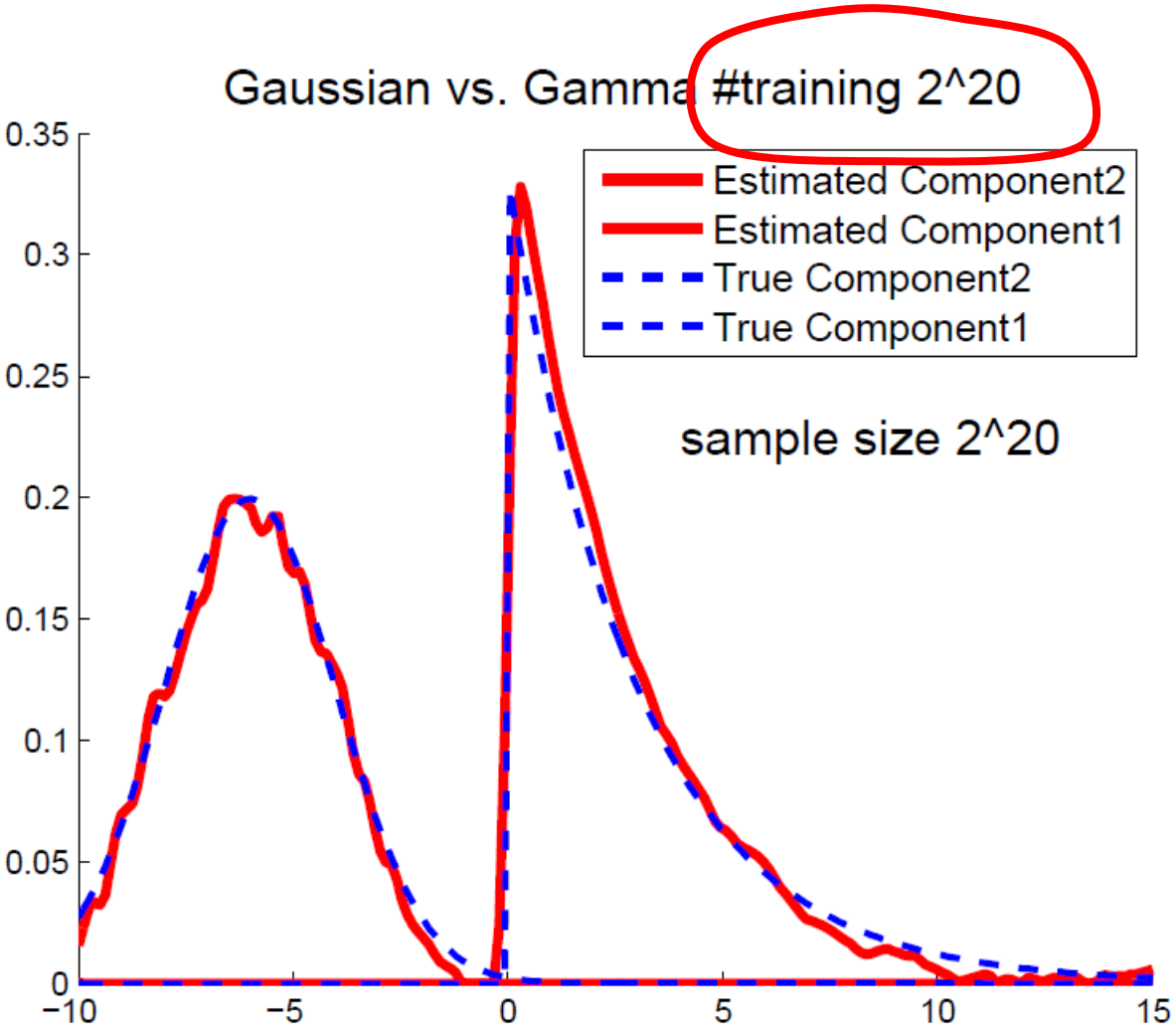
Estimated with
doubly stochastic kernel PCA

Recovery of conditional density

Multi-view latent variable model

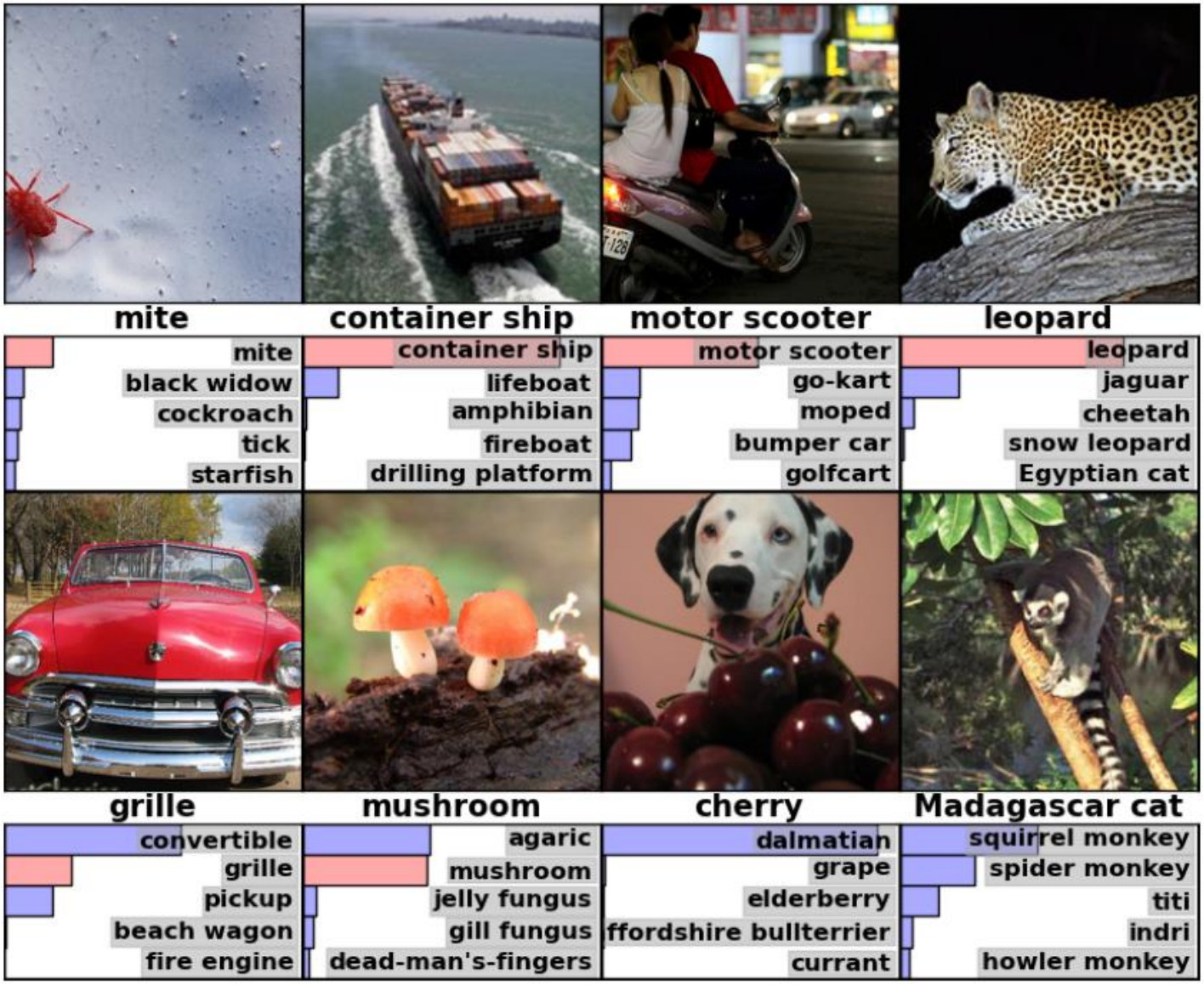
Each view: mixture of Gaussian and shifted Gamma





ImageNet

Image classification with 1.3M color images and 1000 classes



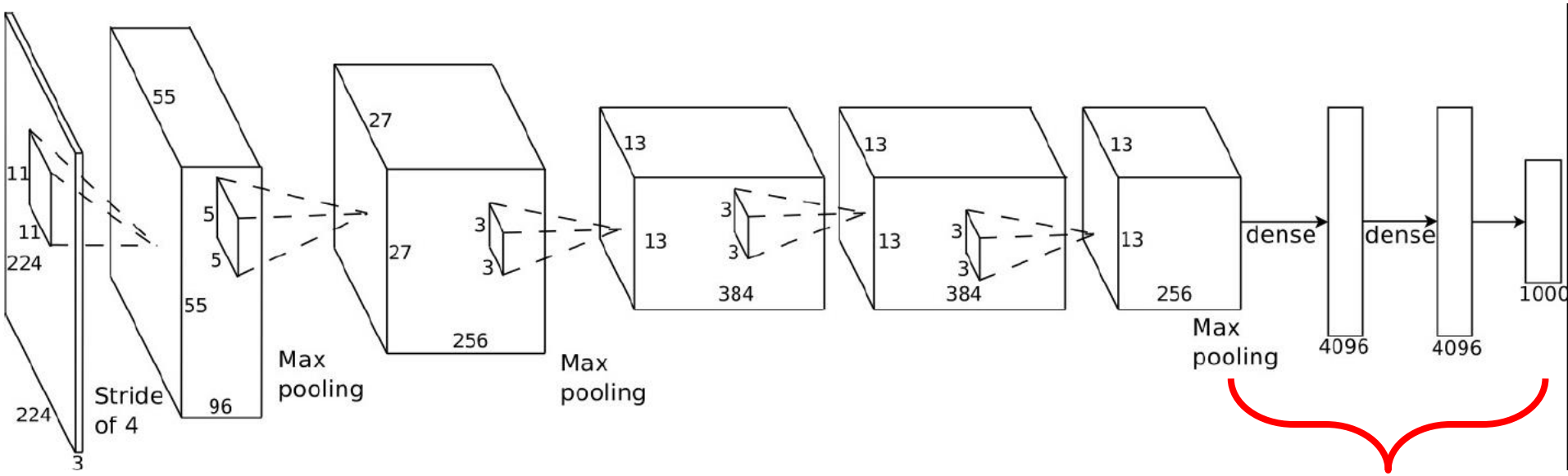
Neural networks and convolution features

8 layer convolution neural network [Krizhevsky12]

First 5 layers: convolution + max pooling

Next 2 layers: fully connected nonlinear neurons

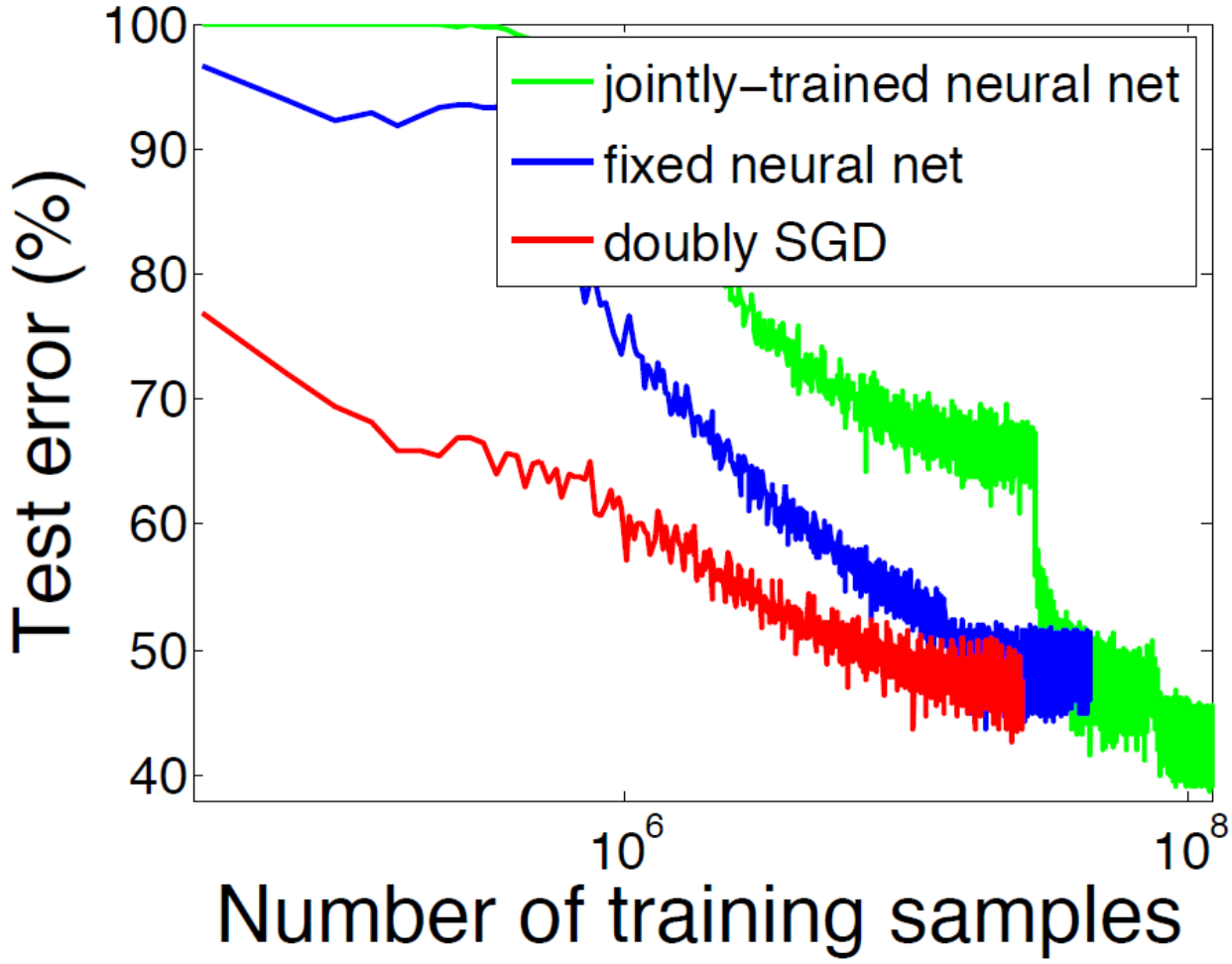
Last layer: multiclass logistic regression



ImageNet

9216 features from the last pooling layer of Alex-Net [Krizhevsky12]

Mini-batch doubly stochastic kernel logistic regression



https://github.com/zixu1986/Doubly_Stochastic_Gradients

Conclusion

Spectral algorithm

- Use algebraic structure of a latent variable model
- Provable guarantee

Kernel embeddings

- Nonparametric LVMs
- Generalize existing spectral algorithms

Doubly stochastic gradients: scale up kernel methods

Future work

- More complex latent variable models and latent processes?
- Remove convolution filters learned by neural nets?
- Doubly stochastic algorithm for other problems