Online Kernel PCA with Entropic Matrix Updates

Dima Kuzmin Manfred K. Warmuth

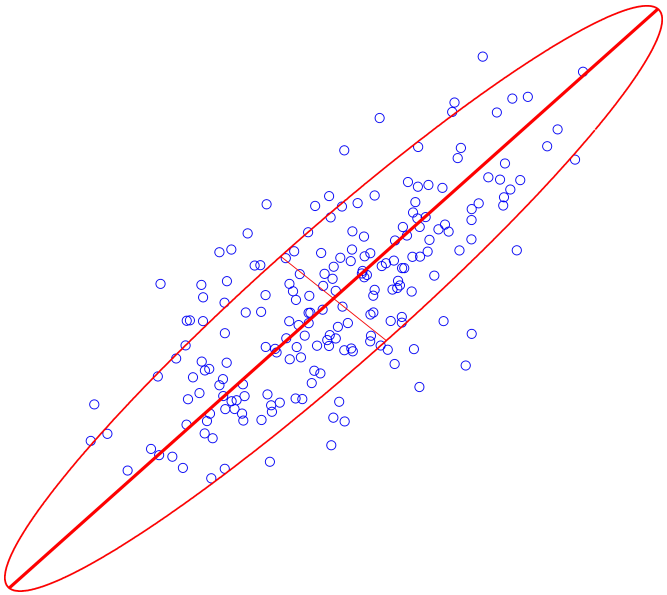
University of California - Santa Cruz

ICML 2007, Corvallis, Oregon
April 23, 2008

Outline

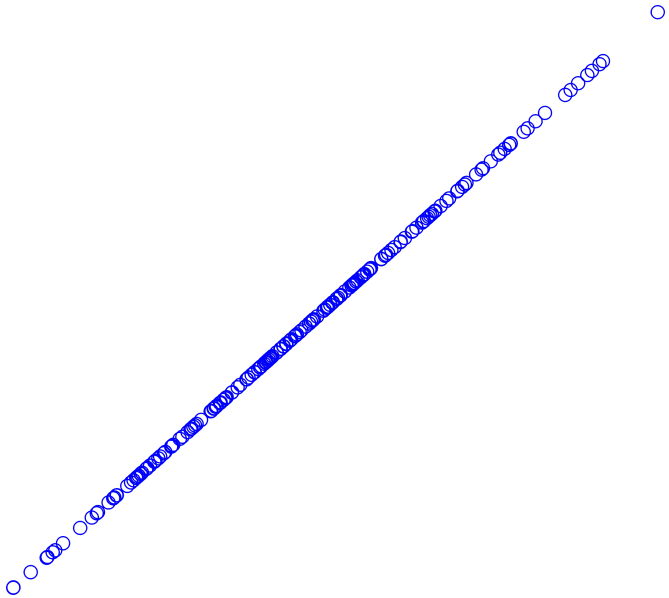
Outline

PCA: dimensionality reduction



project onto subspace

preserve most information



Objective of batch PCA

$$\inf_{\text{center } \mathbf{c}} \quad \inf_{k\text{-dim. proj. matrix } \mathbf{P}} \quad \sum_t || \underbrace{\mathbf{P}(\mathbf{x}_t - \mathbf{c})}_{\text{compressed}} - \underbrace{(\mathbf{x}_t - \mathbf{c})}_{\text{uncompressed}} ||_2^2$$

Solution:

$\mathbf{c}^* =$ average point

$\mathbf{P}^* =$ subspace spanned by k longest axes of covariance matrix $\sum_t (\mathbf{x}_t - \mathbf{c}^*)(\mathbf{x}_t - \mathbf{c}^*)^\top$

What we do

Online PCA

- Update subspace after each point

Goals

- Total compression loss close to batch PCA
- Regret bounds logarithmic in the dimension
- Kernelize the online algorithm

Outline

Why online?

- Data points produced online
- Data changes over time
  - Our algorithms can be adapted to that case
- Want to exploit the sequential nature of the data

Protocol

For t=1 to T

- Algorithm picks k-dimensional projection $\mathbf{P}_t$
- Nature picks data point $\mathbf{x}_t$
- Algorithm suffers loss $\|\mathbf{P}_t\mathbf{x}_t - \mathbf{x}_t\|_2^2$

End For

Regret

$$\underbrace{\sum_{t=1}^{T}\|\mathbf{P}_t\mathbf{x}_t - \mathbf{x}_t\|_2^2}_{\text{online loss}} - \underbrace{\inf_{\mathbf{P}}\sum_{t=1}^{T}\|\mathbf{P}\mathbf{x}_t - \mathbf{x}_t\|_2^2}_{\text{batch loss}}$$

How do we do it?

- Lift methods from expert setting of online learning to matrix setting
- Use density matrices to express uncertainty over best subspace

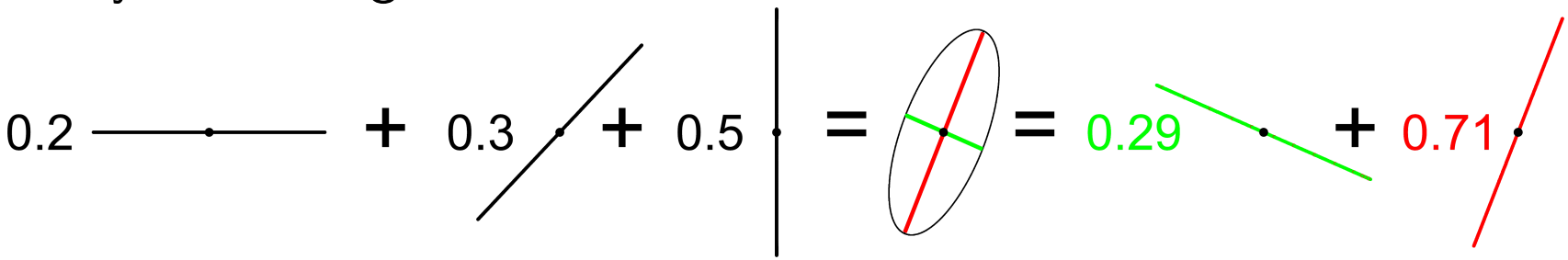
Trick 1: density matrices

Natural parameter for expressing uncertainty over directions

- Symmetric positive definite matrix of trace 1
- Eigenvalues λ_i form probability vector (mixture)

$$\mathbf{W} = \sum_{i=1}^{n} \lambda_i \, \mathbf{w}_i \mathbf{w}_i^\top$$

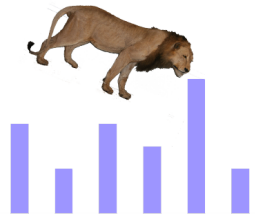
- Many mixtures give same matrix



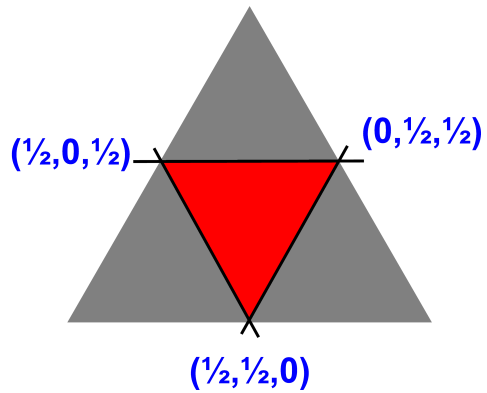
  - Decomposition into n eigendirections

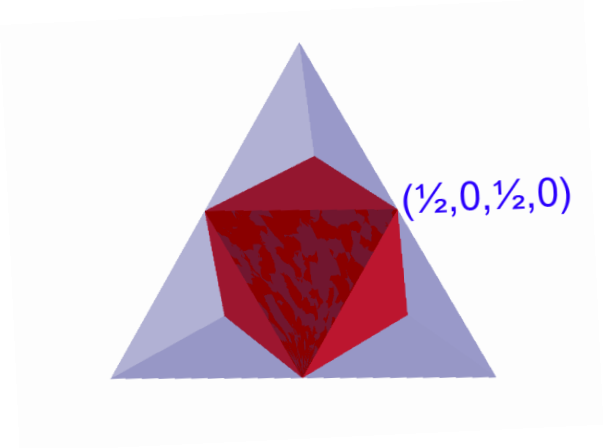
Trick 2: capping the eigenvalues of density matrix

- Probability vector: uncertainty over eigendirections



- Capping prevents concentration on single corner
- $\frac{1}{m}$ capped probability vector: uncertainty over m-sets of directions
- Such sets represented as m-corners: $(0, \frac{1}{m}, 0, 0, \frac{1}{m}, 0, \frac{1}{m})$
- The convex hull of the m-corners = capped probability simplex





- Any distribution in hull decomposable into n out of $\binom{n}{m}$ corners

Trick 3: rewrite quadratic loss as linear loss

Assume $\mathbf{c} = 0$ for now

$$\begin{aligned} \| \underbrace{\mathbf{P}}_{k} \mathbf{x} - \mathbf{x} \|_2^2 &= \| (\mathbf{P} - \mathbf{I})\mathbf{x} \|_2^2 \\ &= \mathbf{x}^\top (\mathbf{I} - \mathbf{P})^2 \mathbf{x} \\ \overset{\mathbf{I}-\mathbf{P} \text{ proj.matr.}}{=} & \operatorname{tr}(\underbrace{(\mathbf{I} - \mathbf{P})}_{n-k} \mathbf{x}\mathbf{x}^\top) \end{aligned}$$

Want to choose $n - k$ dimensional subspace of minimum variance

Projection matrices are symmetric positive matrices w. $\{0, 1\}$ eigenvals

$$\mathbf{P}^2 = \mathbf{P}, \quad (\mathbf{I} - \mathbf{P})^2 = \mathbf{I} - \mathbf{P}$$

Online PCA alg

Initalize $\mathbf{W}_0 = \frac{1}{n}\mathbf{I}$

- Pick $n-k$ dimensional subspace based on capped density matrix $\underbrace{\mathbf{W}_t}_{n-k}$
- Choose complementary subspace $\underbrace{\mathbf{P}_t}_{k}$
- Receive instance $\mathbf{x}_t$
- Incur loss $||\mathbf{P}_t\mathbf{x}_t - \mathbf{x}_t||_2^2 = \mathrm{tr}(\underbrace{(\mathbf{I} - \mathbf{P}_t)}_{n-k}\ \mathbf{x}_t\mathbf{x}_t^\top)$

  and expected loss $(n-k)\,\mathrm{tr}(\mathbf{W}_t\ \mathbf{x}_t\mathbf{x}_t^\top)$
- Update $\mathbf{W}_{t+1} = \mathbf{exp}(\mathbf{log}\,\mathbf{W}_t - \eta\mathbf{x}_t\mathbf{x}_t^\top)/Z$, where $\mathbf{exp}, \mathbf{log}$ are matrix ops
- Cap eigenvals of $\mathbf{W}_{t+1}$ to $\leq \frac{1}{n-k}$

Update and Winnow-like bound

$$\widehat{\mathbf{W}}_t = \frac{\exp(\log \mathbf{W}_t - \eta\, \mathbf{x}_t \mathbf{x}_t^\top)}{\mathrm{tr}(\exp(\log \mathbf{W}_t - \eta\, \mathbf{x}_t \mathbf{x}_t^\top))}$$

$$\mathbf{W}_{t+1} = \inf_{\substack{\mathbf{W} \text{ dens.matrix} \\ \text{w.eigenvals } \leq \frac{1}{n-k}}} \Delta(\mathbf{W}, \hat{\mathbf{W}}_t)$$

$$\text{regret} \quad \leq \quad \sqrt{2 \text{ loss of best } k \text{ subspace } k \log \frac{n}{k} + k \log \frac{n}{k}}$$

Outline

Feature maps

- Expand instance vectors: $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^N$, $N >> n$
- Example:
  $\phi((x_1, \ldots, x_n)) = (x_1^2, x_1 x_2, \ldots, x_1 x_n, x_2^2, \ldots, \ldots, x_{n-1} x_n, x_n^2)$, $N = n^2$
- Dot products can often be computed efficiently: $\underbrace{\phi(\mathbf{x}) \cdot \phi(\mathbf{y})}_{k(\mathbf{x},\mathbf{y})} = (\mathbf{x} \cdot \mathbf{y})^2$
- Kernel PCA computes normal PCA for expanded data instances $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_m)$

Covariance and kernel matrices

- Data matrix $\mathbf{X}$

$$\mathbf{X} = \underbrace{\left(\phi(\mathbf{x}_1)|\ldots|\phi(\mathbf{x}_n)\right)}_{\text{expanded instances as cols}}$$

- Covariance matrix - too big

$$\mathbf{C} = \underset{N\times N}{\mathbf{X}\mathbf{X}^\top} = \sum_{i=1}^{m} \phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\top$$

- Kernel matrix - small

$$\mathbf{K} = \underset{m\times m}{\mathbf{X}^\top\mathbf{X}}, \quad K_{ij} = \underbrace{\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)}_{k(\mathbf{x},\mathbf{y})}$$

- Eigensystems of $\mathbf{K}$ and $\mathbf{C}$ are related !

Eigendecomposition of $\mathbf{K}$ and $\mathbf{C}$

Theorem

If $\mathbf{K}$ *has eigensystem* $(\lambda_i, \mathbf{u}_i)$, *then* $\mathbf{C}$ *has eigensystem* $\left(\lambda_i, \frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}}\right)$

Proof.

- Eigenvalue: $\mathbf{C}\frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}} = \mathbf{X}\mathbf{X}^\top \frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}} = \frac{\mathbf{X}\mathbf{K}\mathbf{u}_i}{\sqrt{\lambda_i}} = \lambda_i \frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}}$
- Orthogonality: $\left(\frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}}\right)^\top \left(\frac{\mathbf{X}\mathbf{u}_j}{\sqrt{\lambda_j}}\right) = \frac{\mathbf{u}_i^\top \mathbf{X}^\top \mathbf{X}\mathbf{u}_j}{\sqrt{\lambda_i\lambda_j}} = \frac{\mathbf{u}_i^\top \mathbf{K}\mathbf{u}_j}{\sqrt{\lambda_i\lambda_j}} = \frac{\lambda_j \mathbf{u}_i^\top \mathbf{u}_j}{\sqrt{\lambda_i\lambda_j}} = 0$
- Normalization: $\|\frac{\mathbf{X}\mathbf{u}_i}{\sqrt{\lambda_i}}\|_2^2 = \frac{1}{\lambda_i}(\mathbf{X}\mathbf{u}_i)^\top(\mathbf{X}\mathbf{u}_i) = \frac{1}{\lambda_i}\mathbf{u}_i^\top \mathbf{K}\mathbf{u}_i = \frac{1}{\lambda_i}\lambda_i \mathbf{u}_i^\top \mathbf{u}_i$

Batch Kernel PCA

- Top k eigenvectors of $\mathbf{C}$ are implicitly given by top k eigenvectors of $\mathbf{K}$
- Projections can be computed based on small $\mathbf{K}$ matrix

Online Kernel PCA

- **Batch PCA**: **max**
  Pick the k eigenvectors of $\mathbf{C}$ with largest eigenvalues.

- **Online PCA**: **soft max**
  Pick a subset of k eigenvectors of $\mathbf{C}$ probabilistically, based on eigenvalues of $\mathbf{exp}(-\eta\mathbf{C})$

- **Online KPCA**
  Can compute everything about $\mathbf{exp}(-\eta\mathbf{C})$ i.t.o. $\mathbf{K}$

Bound

$$\text{regret} \leq \sqrt{2 \text{ loss of best } k \text{ subspace } k \log \frac{N}{k}} + k \log \frac{N}{k}$$

N - dimension of feature space

Kernelizable?

Vector case:

- Additive updates $w \sim \sum_t \phi(x_t)$ ✔
- Multiplicative updates $w_i \sim e^{-\eta \sum \phi(x_{t,i})}$ (✔)

Our results:

- Matrix multiplicative updates $\mathbf{W} \sim \mathbf{exp}(-\eta \sum_t \phi(\mathbf{x}_t)\phi(\mathbf{x}_t)^\top)$ ✔
- Essentially any matrix update based on spectral function $\mathbf{f}$ ✔

$$\mathbf{W} \sim \mathbf{f}(-\eta \sum_t \phi(\mathbf{x}_t)\phi(\mathbf{x}_t)^\top)$$

- Works only for rank 1 instances $\phi(\mathbf{x})\phi(\mathbf{x})^\top$
  - Standard basis vectors in vector case

Derivation and analysis

Online PCA

$$\mathbf{W}^{t+1} = \operatorname*{argmin}_{\substack{\mathrm{tr}(\mathbf{W})=1 \\ \mathbf{W} \preceq \frac{1}{N-r}\mathbf{I}}} \left(\overbrace{\Delta(\mathbf{W}, \mathbf{W}_t)}^{\text{quantum relative entropy}} + \eta \,\mathrm{tr}(\mathbf{W}\, \mathbf{x}_t \mathbf{x}_t^\top) \right)$$

Kernel Online PCA

$$\mathbf{W}^{t+1} = \operatorname*{argmin}_{\substack{\mathrm{tr}(\mathbf{W})=1 \\ \mathbf{W} \preceq \frac{1}{N-r}\mathbf{I}}} \left(\Delta(\mathbf{W}, \frac{1}{N}\mathbf{I}) + \eta \sum_t \mathrm{tr}(\mathbf{W}\, \phi(\mathbf{x}_t)\phi(\mathbf{x}_t)^\top) \right)$$

Analysis

Bregman projection methods or duality

$$\Delta(\mathbf{W}, \mathbf{U}) = \mathrm{tr}(\mathbf{W}(\log \mathbf{W} - \log \mathbf{U}))$$

What's next

- Bounds for estimating the center online ✔
- Shifting with kernels, in particular long term memory
- Approximation algs - efficiency
- Other apps for matrix soft min and soft smallest k

Additional loss of online algorithm