Bandit PCA

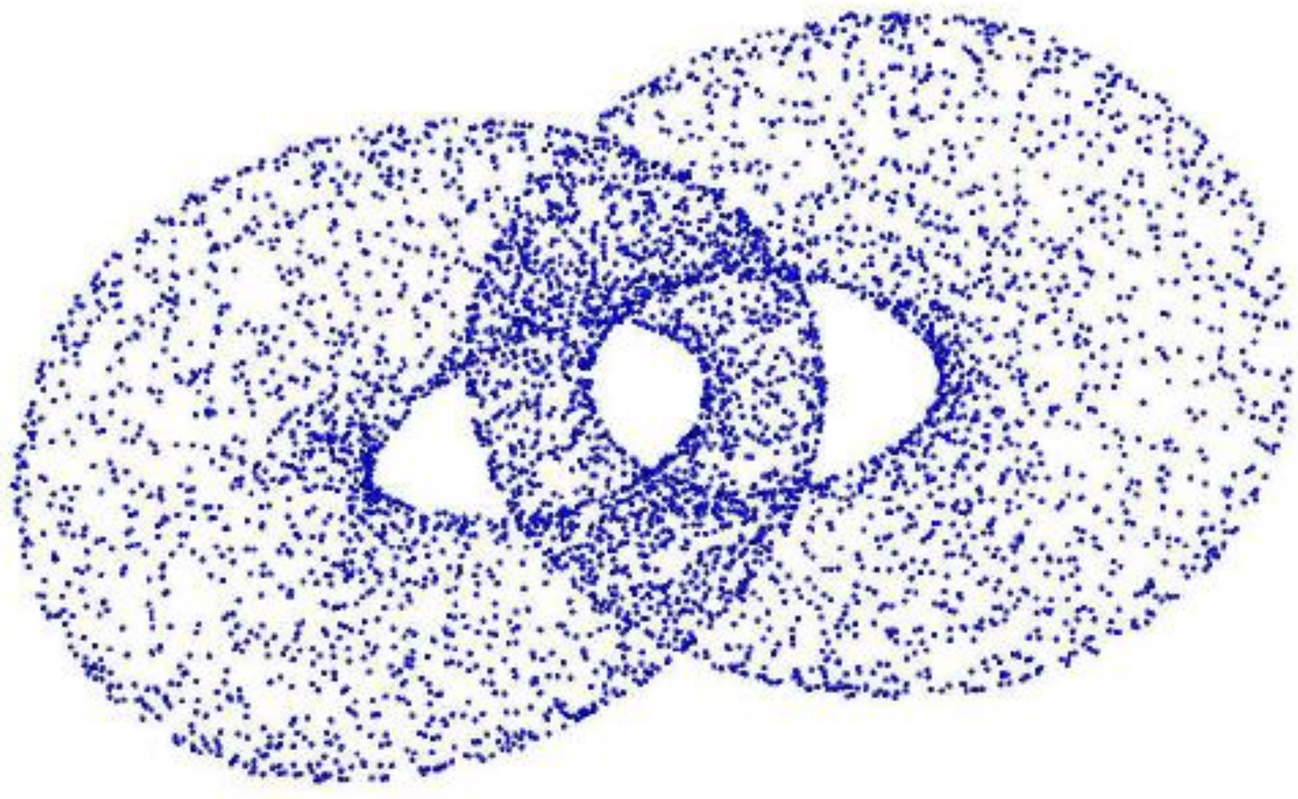
Gergely Neu
Univ. Pompeu Fabra (Barcelona, Spain)

joint work with Wojciech Kotłowski

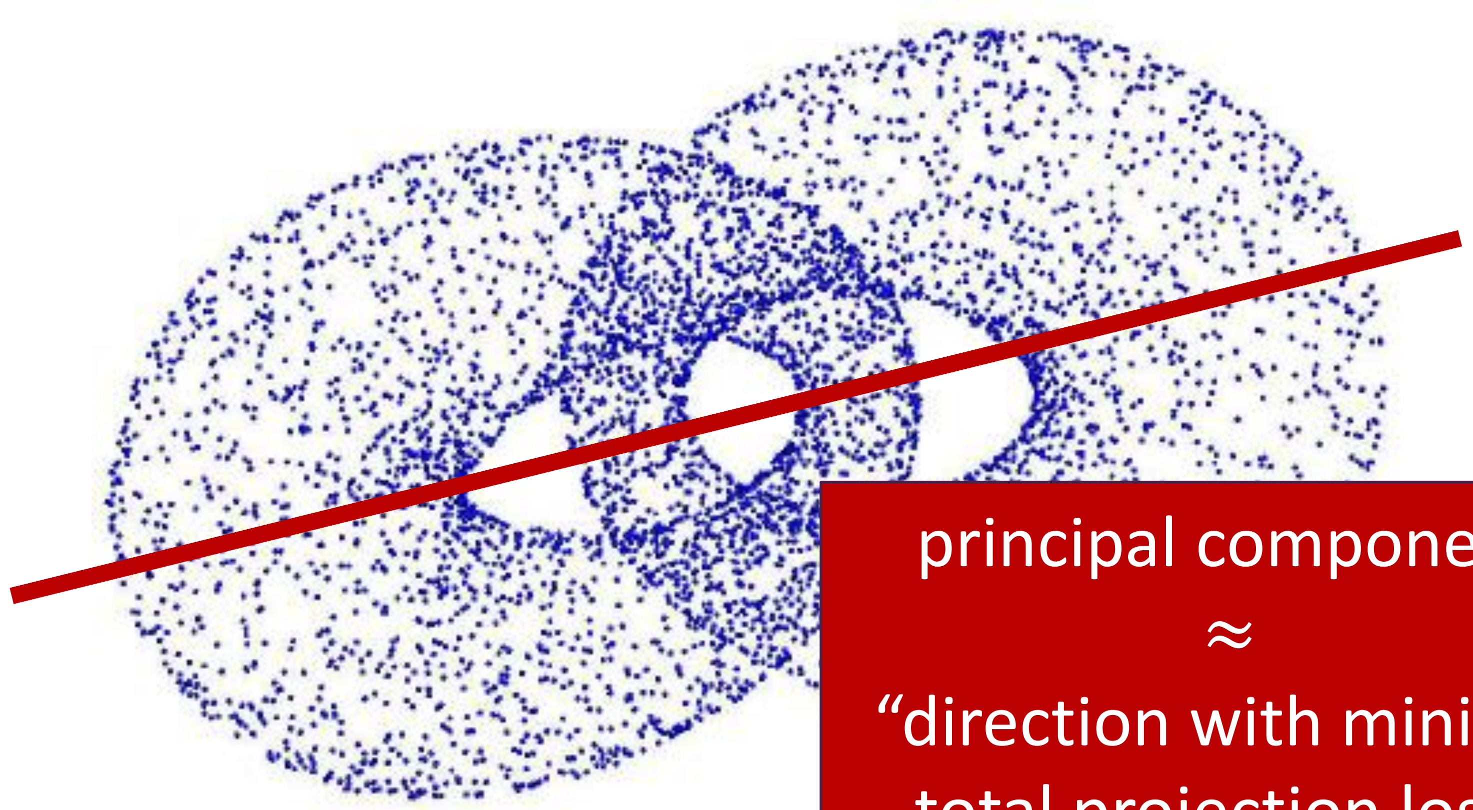
Appetizer

PCA,
bandit PCA,
phase retrieval

Principal component analysis (PCA)



Principal component analysis (PCA)



principal component
≈
"direction with minimal total projection loss"

Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

$t = 1$

Bandit PCA

Principal Component Analysis with

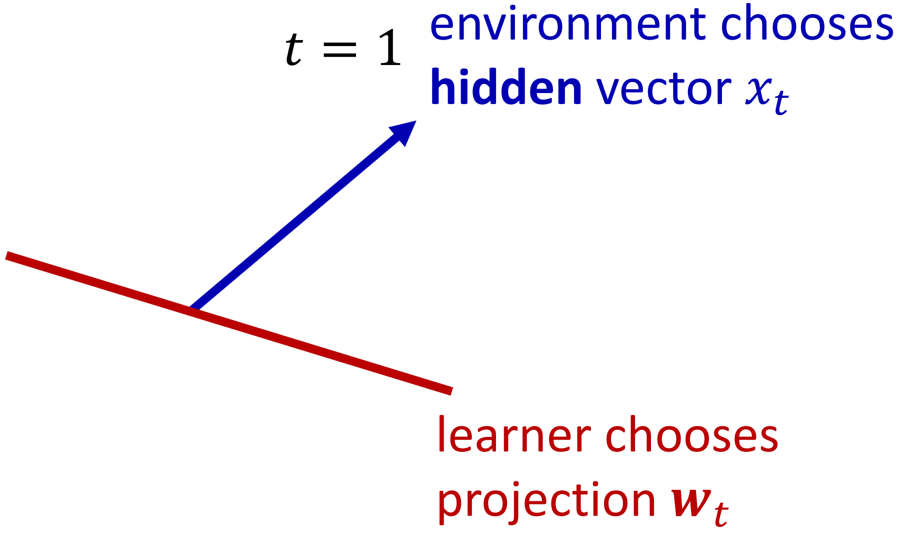
- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

$t = 1$ environment chooses **hidden** vector x_t

Bandit PCA

Principal Component Analysis with

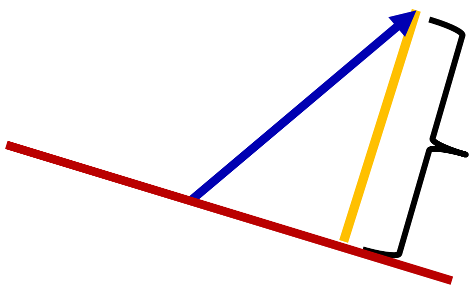
- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



Bandit PCA

Principal Component Analysis with
- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

$t = 1$ environment chooses
hidden vector x_t



Learner incurs and observes projection loss $1 - (w_t^\top x_t)^2$

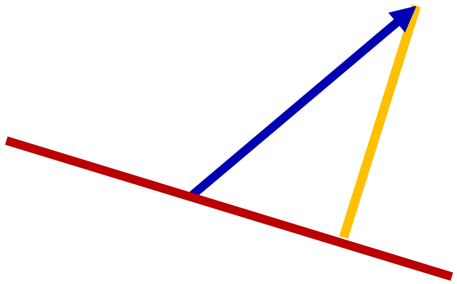
learner chooses
projection $\boldsymbol{w}_t$

Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

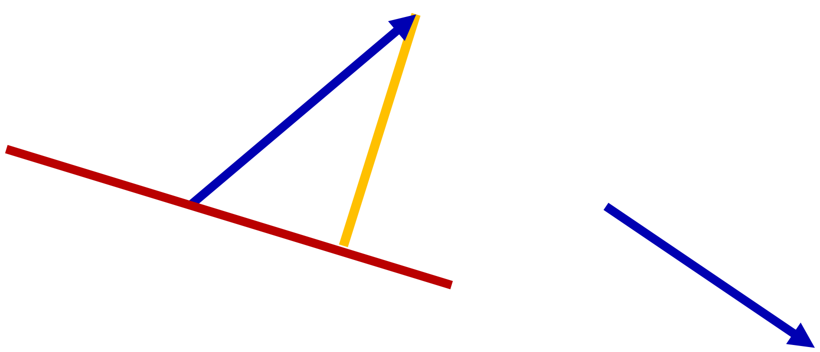
$t = 1$ $t = 2$



Bandit PCA

Principal Component Analysis with

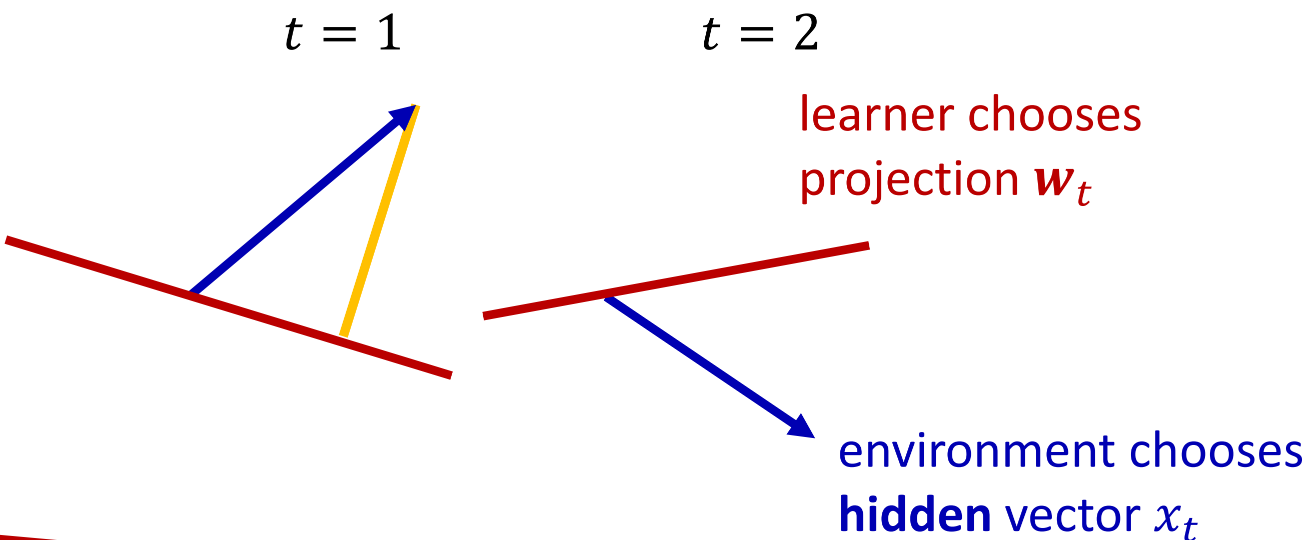
- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



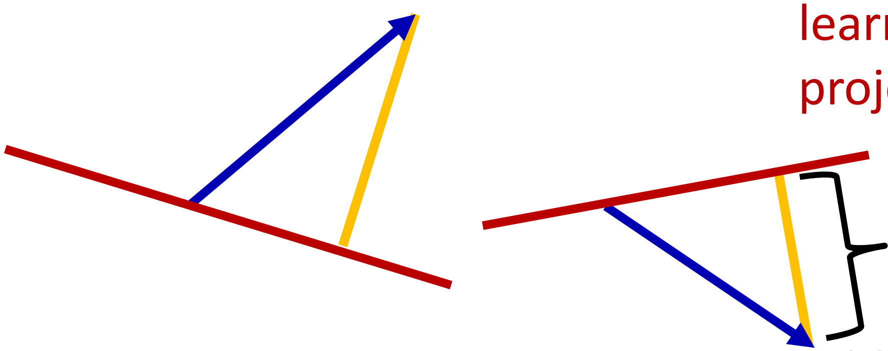
Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)

$t = 1$ $\qquad$ $t = 2$

learner chooses projection $\boldsymbol{w}_t$



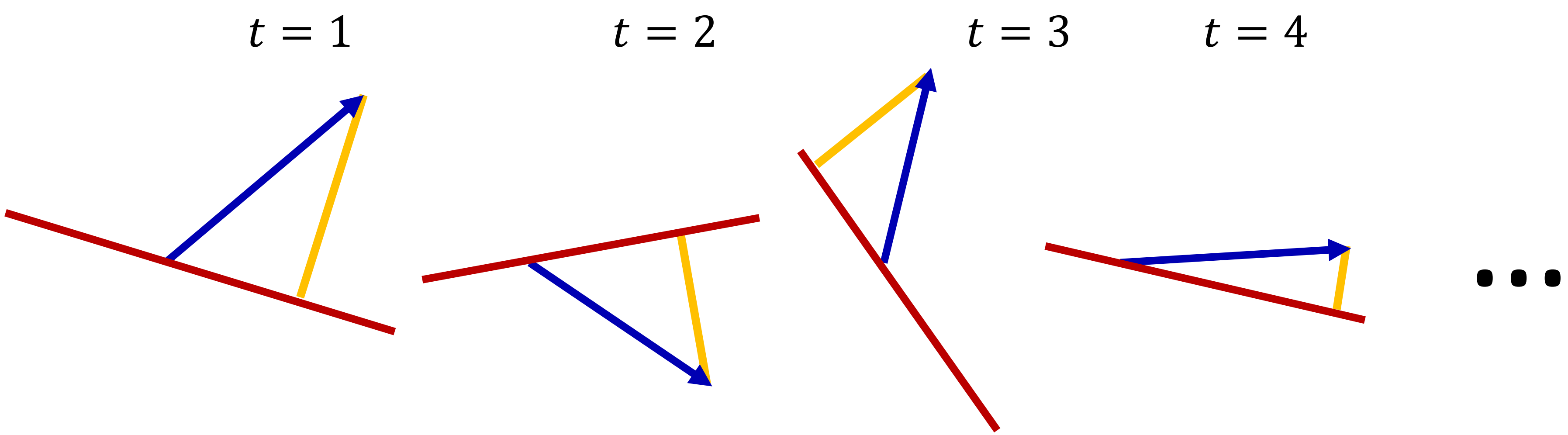
Learner incurs and observes projection loss $1 - (w_t^\top x_t)^2$

environment chooses **hidden** vector x_t

Bandit PCA

Principal Component Analysis with

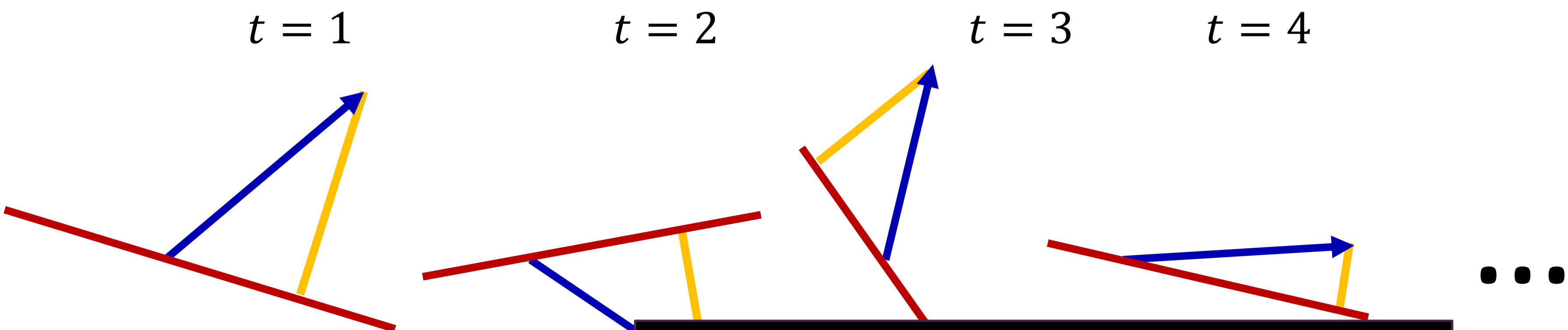
- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



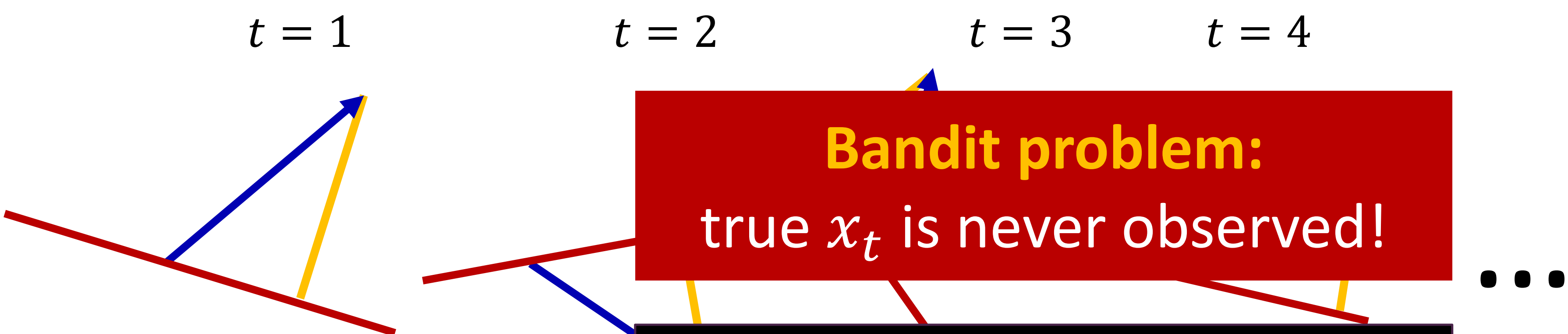
GOAL:

minimize total projection loss

Bandit PCA

Principal Component Analysis with

- sequentially chosen projections (online PCA)
- partial observability (bandit PCA)



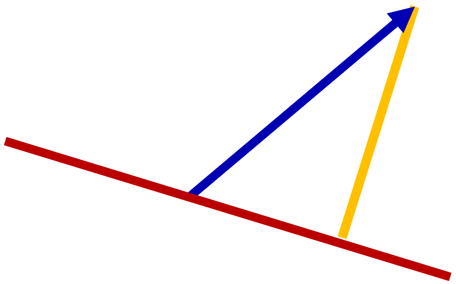
Bandit problem:

true x_t is never observed!

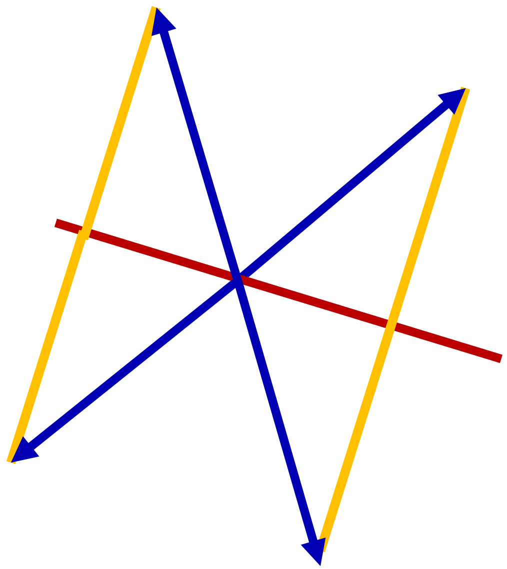
GOAL:

minimize total projection loss

Why is this hard?

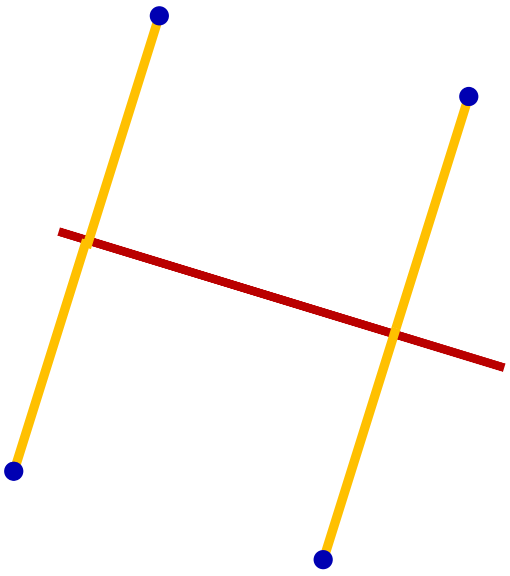


Why is this hard?



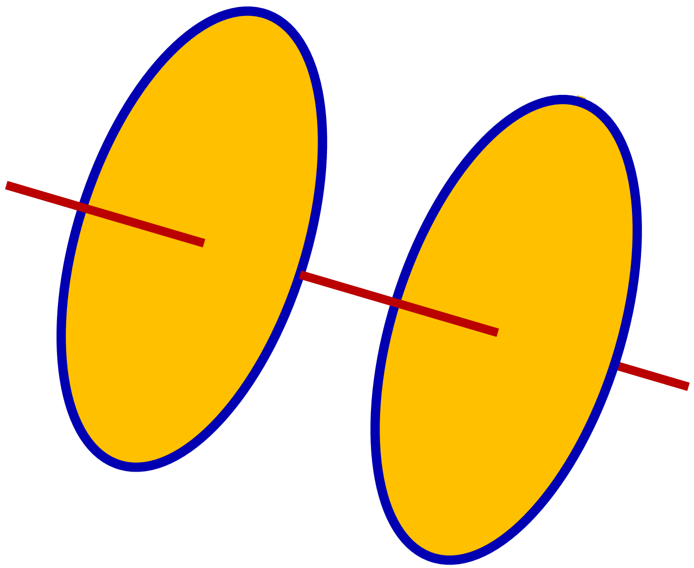
x_t is ambiguous!

Why is this hard?



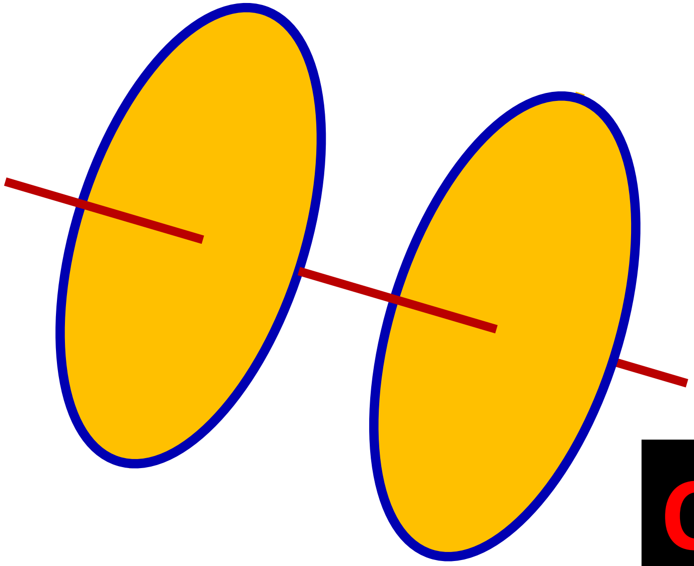
x_t is ambiguous!

Why is this hard?



x_t is ambiguous!
even worse in high dim

Why is this hard?



x_t is ambiguous!
even worse in high dim

Challenge:
How do we reconstruct x_t from a **single** projection?

NB: $\pm x_t$ are impossible to tell apart with any number of projections

Why is this interesting?

Why is this interesting?

Bandit PCA $\approx$ online phase retrieval

Why is this interesting?

Bandit PCA ≈ online phase retrieval

Phase retrieval:

- $\boldsymbol{w}_t \sim \mathcal{N}(0, I_{d\times d})$ i.i.d.
- $\boldsymbol{x}_t = \boldsymbol{x}$ fixed
- Observations:

  $|\boldsymbol{x}^\top \boldsymbol{w}_t|^2$ (+noise)

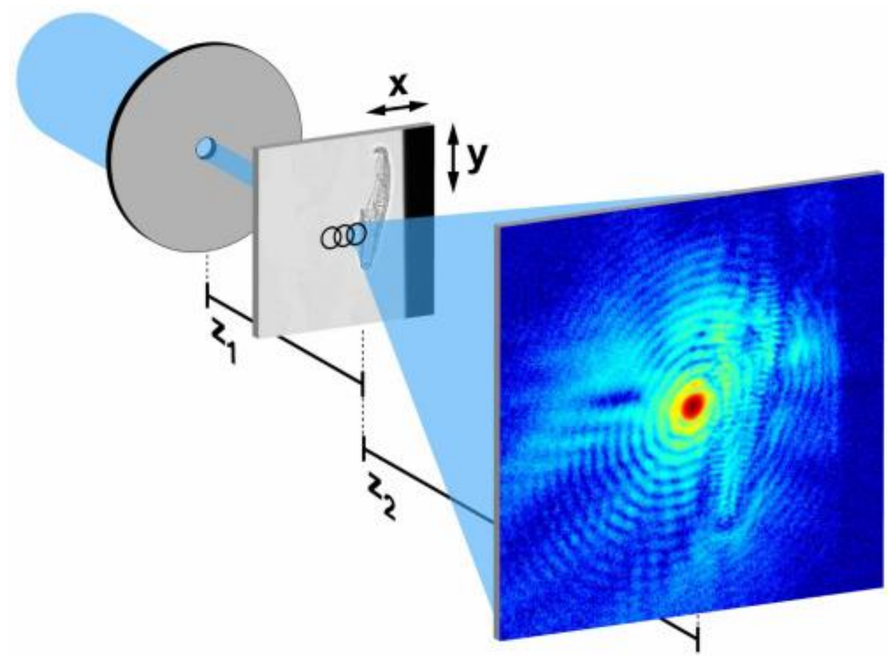
Fienup (1982), Millane (1990)

Why is this interesting?

Bandit PCA ≈ online phase retrieval

Phase retrieval:

- $\boldsymbol{w}_t \sim \mathcal{N}(0, I_{d\times d})$ i.i.d.
- $\boldsymbol{x}_t = \boldsymbol{x}$ fixed
- Observations:
  $|\boldsymbol{x}^\top \boldsymbol{w}_t|^2$ (+noise)
- Applications in
  - *diffractive imaging*
  - *X-ray crystallography*
  - *astronomy…*



Fienup (1982), Millane (1990)

Why is this interesting?

Bandit PCA ≈ online phase retrieval

Phase retrieval:

- $\boldsymbol{w}_t \sim \mathcal{N}(0, I_{d\times d})$ i.i.d.
- $\boldsymbol{x}_t = \boldsymbol{x}$ fixed
- Observations:
  $|\boldsymbol{x}^\top \boldsymbol{w}_t|^2$ (+noise)
- Applications in
  - *diffractive imaging*
  - *X-ray crystallography*
  - *astronomy…*

Fienup (1982), Millane (1990)

Bandit PCA:

- $\boldsymbol{w}_t$ chosen adaptively
- $\boldsymbol{x}_t$ arbitrary
- Observations:
  $|\boldsymbol{x}_t^\top \boldsymbol{w}_t|^2$ (+noise)

Why is this interesting?

Bandit PCA ≈ online phase retrieval

Phase retrieval:

- $\boldsymbol{w}_t \sim \mathcal{N}(0, I_{d\times d})$ i.i.d.
- $\boldsymbol{x}_t = \boldsymbol{x}$ fixed
- Observations:

  $|\boldsymbol{x}^\top \boldsymbol{w}_t|^2$ (+noise)
- Applications in
  - *diffractive imaging*
  - *X-ray crystallography*
  - *astronomy…*

Fienup (1982), Millane (1990)

Bandit PCA:

- $\boldsymbol{w}_t$ chosen adaptively
- $\boldsymbol{x}_t$ arbitrary
- Observations:

  $|\boldsymbol{x}_t^\top \boldsymbol{w}_t|^2$ (+noise)

Applicable in the same settings but with **adaptive measurements**!

Let's get technical

Classic tricks for online PCA

Bandit PCA – general framework

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks unit-norm vector $\boldsymbol{w}_t$
- Learner incurs and observes loss $\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t$

Generalizes the basic PCA setup with $\boldsymbol{L}_t = \boldsymbol{x}_t \boldsymbol{x}_t^\top$

Bandit PCA – general framework

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks unit-norm vector $\boldsymbol{w}_t$
- Learner incurs and observes loss $\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t$

Generalizes the basic PCA setup with $\boldsymbol{L}_t = \boldsymbol{x}_t \boldsymbol{x}_t^\top$

GOAL:

minimize total expected regret

$$\mathrm{regret}_T = \max_{\boldsymbol{u}:\|\boldsymbol{u}\|=1} \mathbb{E}\left[\sum_{t=1}^{T} (\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t - \boldsymbol{u}^\top \boldsymbol{L}_t \boldsymbol{u})\right]$$

Bandit PCA – general framework

For $t = 1,2, \ldots, T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks unit-norm vector $\boldsymbol{w}_t$
- Learner incurs and observes loss $\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t$

Generalizes the basic PCA setup with $\boldsymbol{L}_t = \boldsymbol{x}_t \boldsymbol{x}_t^\top$

GOAL:

m ret

$$\text{regret}_T = \max_{\boldsymbol{u}:\|\boldsymbol{u}\|=1} \mathbb{E}\left[\sum_{t=1} (\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t - \boldsymbol{u}^\top \boldsymbol{L}_t \boldsymbol{u})\right]$$

Nonlinear loss!!!!

Linearizing the losses: SDP formulation

Warmuth and Kuzmin (2006,2008)

Observation #1:

- Loss is linear in matrix variable $\boldsymbol{w}_t \boldsymbol{w}_t^\top$:

$$\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t = \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$$

Linearizing the losses: SDP formulation

Warmuth and Kuzmin (2006,2008)

Observation #1:

- Loss is linear in matrix variable $\boldsymbol{w}_t \boldsymbol{w}_t^\top$:

$$\boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t = \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$$

Observation #2:

- The non-convex set of matrices $\boldsymbol{w}\boldsymbol{w}^\top$ can be convexified through randomization:

$$\begin{aligned} \mathcal{S} &= \mathrm{conv}(\boldsymbol{w}\boldsymbol{w}^\top : \|\boldsymbol{w}\| = 1) \\ &= \{\boldsymbol{W} : \boldsymbol{W} \succcurlyeq 0, \mathrm{tr}(\boldsymbol{W}) = 1\} \end{aligned}$$

Bandit PCA
= Bandit linear optimization

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks density matrix $\boldsymbol{W}_t \in \mathcal{S}$
- Learner draws random $\boldsymbol{w}_t$ s.t. $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$
- Learner incurs and observes loss

$$\langle \boldsymbol{w}_t \boldsymbol{w}_t^\top, \boldsymbol{L}_t \rangle \stackrel{\text{def}}{=} \operatorname{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$$

Bandit PCA
= Bandit linear optimization

For $t = 1,2,\dots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks density matrix $\boldsymbol{W}_t \in \mathcal{S}$
- Learner draws random $\boldsymbol{w}_t$ s.t. $\mathbb{E}[\boldsymbol{w}_t\boldsymbol{w}_t^\top] = \boldsymbol{W}_t$
- Learner incurs and observes loss

$$\langle \boldsymbol{w}_t\boldsymbol{w}_t^\top, \boldsymbol{L}_t\rangle \stackrel{\text{def}}{=} \operatorname{tr}(\boldsymbol{w}_t\boldsymbol{w}_t^\top\boldsymbol{L}_t)$$

Idea:

Apply a generic linear bandit algorithm!

Bandit PCA
= Bandit linear optimization

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks density matrix $\boldsymbol{W}_t \in \mathcal{S}$
- Learner draws random $\boldsymbol{w}_t$ s.t. $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$
- Learner incurs and observes loss
  $\langle \boldsymbol{w}_t \boldsymbol{w}_t^\top, \boldsymbol{L}_t \rangle \overset{\text{def}}{=} \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$

Idea:
Apply a generic linear bandit algorithm!

GeometricHedge guarantees
$$\mathrm{regret}_T = \tilde{O}\left(d^2\sqrt{T}\right)$$

Dani, Hayes, Kakade (2008),
Bubeck and Eldan (2015)

Bandit PCA
= Bandit linear optimization

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks density matrix $\boldsymbol{W}_t \in \mathcal{S}$
- Learner draws random $\boldsymbol{w}_t$ s.t. $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$
- Learner incurs and observes loss
  $\langle \boldsymbol{w}_t \boldsymbol{w}_t^\top, \boldsymbol{L}_t \rangle \stackrel{\text{def}}{=} \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$

Idea:
Apply a generic linear bandit algorithm!

GeometricHedge guarantees
$$\mathrm{regret}_T = \tilde{O}\left(d^2\sqrt{T}\right)$$

Dani, Hayes, Kakade (2008),
Bubeck and Eldan (2015)

BUT
no polytime implementation is known ☹☹☹

Bandit PCA
= Bandit linear optimization

For $t = 1,2,\ldots,T$

- Environment picks secret loss matrix $\boldsymbol{L}_t$
- Learner picks density matrix $\boldsymbol{W}_t \in \mathcal{S}$
- Learner draws random $\boldsymbol{w}_t$ s.t. $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$
-

Idea:

Apply a generic linear bandit ...thm!

Ge

Dani, Hayes, Kakade (2008),
Bubeck and Eldan (2015)

...T

...time

...ntation

is known ☹☹☹

Our contribution:

a fast algorithm with regret

$$O\left(d^{3/2}\sqrt{T\log T}\right)$$

Main course

Algorithm

Main results

Main course

Algorithm

Main results

Online Mirror Descent for bandit PCA

Idea: rely on the good old template

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \widehat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

Online Mirror Descent for bandit PCA

Idea: rely on the good old template

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \widehat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

+

Sample $\boldsymbol{w}_t$ so that $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

Online Mirror Descent for bandit PCA

loss estimate $\hat{\boldsymbol{L}}_t = ?$

divergence $D = ?$

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W} \in \mathcal{S}} \{\eta \langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t)\}$$

+

sampling scheme?

Sample $\boldsymbol{w}_t$ so that $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

sampling scheme?

loss estimate $\hat{L}_t =$?

Sampling scheme?

Recall:
$\sum_i \lambda_i = \mathrm{tr}(W) = 1$
$\lambda_i \geq 0$

First thought:

decompose $\boldsymbol{W}_t = \sum_i \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\top$
and sample w_t so that $\mathbb{P}[\boldsymbol{w}_t = \boldsymbol{u}_i] = \lambda_i$

Warmuth and Kuzmin (2006)

Sampling scheme?

Recall:
$\sum_i \lambda_i = \mathrm{tr}(W) = 1$
$\lambda_i \geq 0$

First thought:

decompose $\boldsymbol{W}_t = \sum_i \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\top$
and sample w_t so that $\mathbb{P}[\boldsymbol{w}_t = \boldsymbol{u}_i] = \lambda_i$

Warmuth and Kuzmin (2006)

Unbiased: $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

Sampling scheme?

Recall:
$\sum_i \lambda_i = \mathrm{tr}(W) = 1$
$\lambda_i \geq 0$

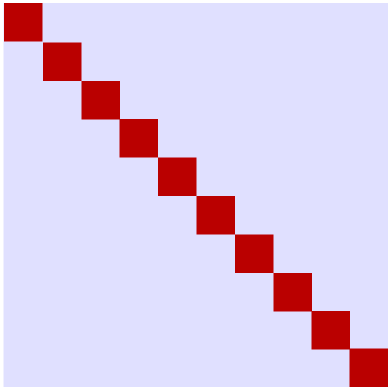
First thought:

decompose $\boldsymbol{W}_t = \sum_i \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\top$
and sample w_t so that $\mathbb{P}[\boldsymbol{w}_t = \boldsymbol{u}_i] = \lambda_i$

Warmuth and Kuzmin (2006)

Unbiased: $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

only senses "diagonal" elements of $\boldsymbol{L}_t$ ☹☹



Sampling scheme?

Recall:
$\sum_i \lambda_i = \mathrm{tr}(W) = 1$
$\lambda_i \geq 0$

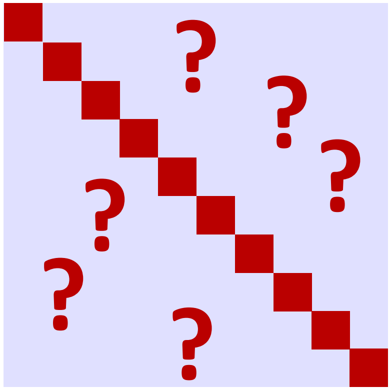
First thought:

decompose $\boldsymbol{W}_t = \sum_i \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^\top$
and sample w_t so that $\mathbb{P}[\boldsymbol{w}_t = \boldsymbol{u}_i] = \lambda_i$

Warmuth and Kuzmin (2006)

Unbiased: $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

only senses "diagonal" elements of $\boldsymbol{L}_t$ ☹☹



Sampling done right

Sparse sampling

- sample **two** indices

$$i, j \sim \lambda$$

- if $i = j$, set

$$\boldsymbol{w}_t = \boldsymbol{u}_i$$

- otherwise draw random sign $s \in \{-1,1\}$ and set

$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$

Sampling done right

Sparse sampling

- sample **two** indices

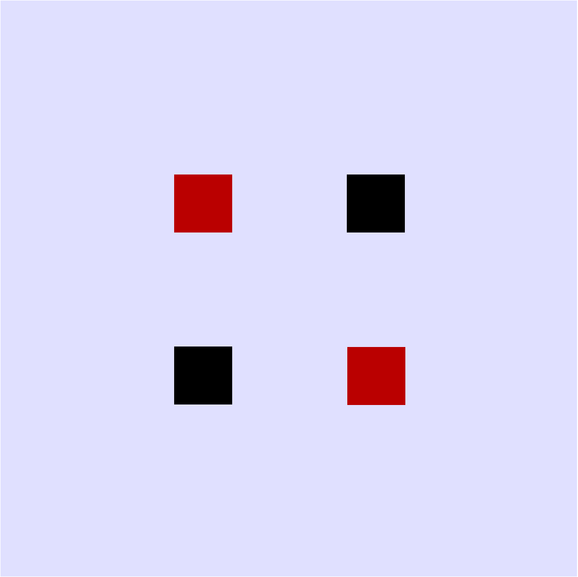
$$i, j \sim \lambda$$

- if $i = j$, set

$$\boldsymbol{w}_t = \boldsymbol{u}_i$$

- otherwise draw random sign $s \in \{-1, 1\}$ and set

$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$



Sampling done right

Sparse sampling

- sample **two** indices

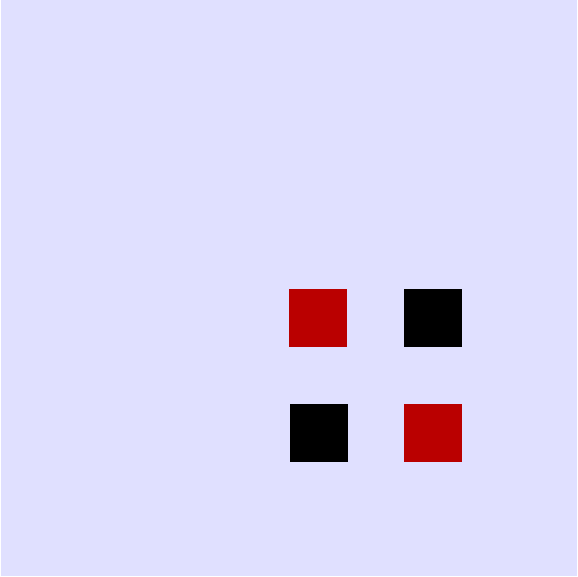
$$i, j \sim \lambda$$

- if $i = j$, set

$$\boldsymbol{w}_t = \boldsymbol{u}_i$$

- otherwise draw random sign $s \in \{-1,1\}$ and set

$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$



Sampling done right

Sparse sampling

- sample **two** indices

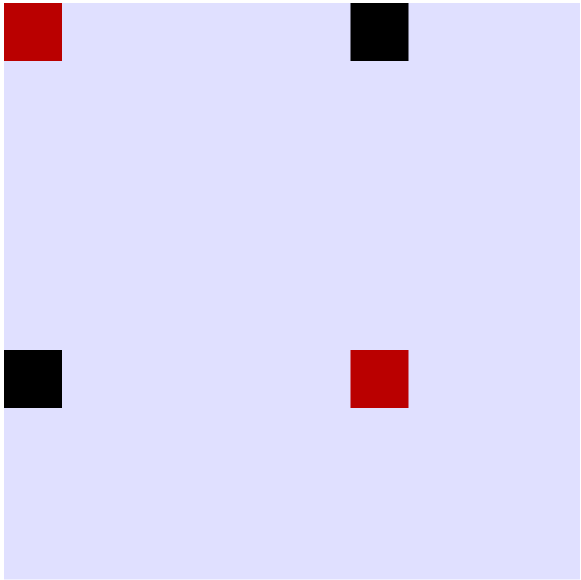
$$i, j \sim \lambda$$

- if $i = j$, set

$$\boldsymbol{w}_t = \boldsymbol{u}_i$$

- otherwise draw random sign $s \in \{-1, 1\}$ and set

$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$



Sampling done right

Sparse sampling

- sample **two** indices

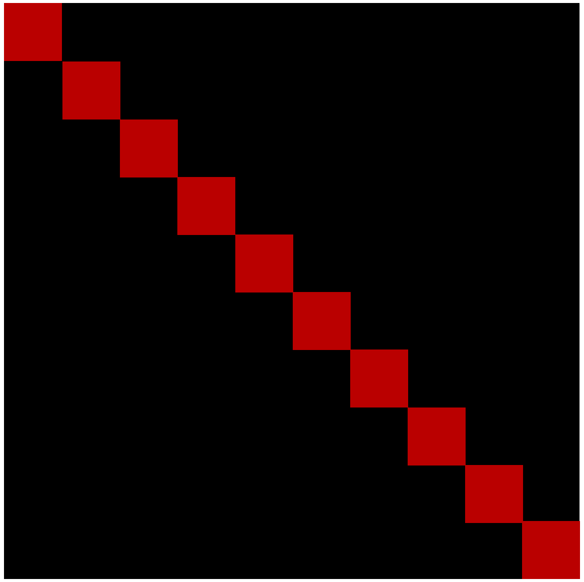
$$i, j \sim \lambda$$

- if $i = j$, set

$$\boldsymbol{w}_t = \boldsymbol{u}_i$$

- otherwise draw random sign $s \in \{-1, 1\}$ and set

$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$



Sampling and loss estimation done right

Sparse sampling

- sample **two** indices
$$i, j \sim \lambda$$
- if $i = j$, set
$$\boldsymbol{w}_t = \boldsymbol{u}_i$$
- otherwise draw random sign $s \in \{-1,1\}$ and set
$$\boldsymbol{w}_t = \frac{1}{\sqrt{2}}\left(\boldsymbol{u}_i + s\boldsymbol{u}_j\right)$$

Loss estimation

- let $\ell = \boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t$
- if $i = j$, set
$$\hat{\boldsymbol{L}}_t = \frac{\ell}{\lambda_i^2} \boldsymbol{u}_i \boldsymbol{u}_i^\top$$
- otherwise set
$$\hat{\boldsymbol{L}}_t = \frac{s\ell}{\lambda_i \lambda_j}\left(\boldsymbol{u}_j \boldsymbol{u}_i^\top + \boldsymbol{u}_i \boldsymbol{u}_j^\top\right)$$

Sampling and loss estimation done right

Sparse sampling

- sample **two** indices
$$i, j \sim \lambda$$
- if $i = j$, set
$$\boldsymbol{w}_t = \boldsymbol{u}_i$$
- otherwise draw random sign $s \in \{-1,1\}$ and set

Lemma:
$$\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$$

Loss estimation

- let $\ell = \boldsymbol{w}_t^\top \boldsymbol{L}_t \boldsymbol{w}_t$
- if $i = j$, set
$$\hat{\boldsymbol{L}}_t = \frac{\ell}{\lambda_i^2} \boldsymbol{u}_i \boldsymbol{u}_i^\top$$
- otherwise set
$$\hat{\boldsymbol{L}}_t = \frac{s\ell}{\quad} \left(\quad + \boldsymbol{u}_i \boldsymbol{u}_j^\top \right)$$

Lemma:
$$\mathbb{E}\left[\hat{\boldsymbol{L}}_t\right] = \boldsymbol{L}_t$$

divergence $D = ?$

What divergence?

First thought:

the usual quantum relative entropy

$$D(\boldsymbol{W} \| \boldsymbol{U}) = \boldsymbol{W} \log(\boldsymbol{W}\boldsymbol{U}^{-1})$$

induced by the quantum entropy $R(\boldsymbol{W}) = \boldsymbol{W} \log \boldsymbol{W}$

What divergence?

First thought:

the usual quantum relative entropy

$$D(\boldsymbol{W} \| \boldsymbol{U}) = \boldsymbol{W} \log(\boldsymbol{W}\boldsymbol{U}^{-1})$$

induced by the quantum entropy $R(\boldsymbol{W}) = \boldsymbol{W} \log \boldsymbol{W}$

a.k.a. “Matrix Hedge”

Warmuth and Kuzmin (2006)

Matrix Hedge for Bandit PCA does not work?

W.K.

June 25, 2018

Consider the adversarial $k = 1$ PCA with bandit feedback. In each trial, the algorithm plays with a rank-one matrix $\boldsymbol{w}_t \boldsymbol{w}_t^\top$ with $\boldsymbol{w}_t \in \mathbb{R}^d$, $\|\boldsymbol{w}_t\| = 1$. Then, nature chooses a symmetric loss matrix $\boldsymbol{L}_t \in \mathbb{R}^{d \times d}$ with eigenvalues bounded in $[0, 1]$, and the algorithm receives and observes loss $\ell_t = \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$.

We start with a standard bound on the Matrix Hedge algorithm: for any loss sequence $\widetilde{\boldsymbol{L}}_1, \ldots, \widetilde{\boldsymbol{L}}_T$ such that each $\widetilde{\boldsymbol{L}}_t$ has eigenvalues in the range $[-a, \infty)$, the sequence of density matrices $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_T$ produced by Matrix Hedge with fixed learning rate η has regret against a comparator density matrix $\boldsymbol{U}$ upper-bounded by:

$$\mathrm{regret}_T(\boldsymbol{U}) = \sum_{t=1}^{T} \mathrm{tr}((\boldsymbol{W}_t - \boldsymbol{U})\widetilde{\boldsymbol{L}}_t) \leq \frac{\ln d}{\eta} + \kappa(\eta a)\eta \sum_{t=1}^{T} \mathrm{tr}(\boldsymbol{W}_t \widetilde{\boldsymbol{L}}_t^2),$$

where $\kappa(x) = \frac{e^x - x - 1}{x^2}$. The trick is now to use this bound in the bandit case as follows: in each trial $t = 1, \ldots, T$, the algorithm probabilistically chooses $\boldsymbol{w}_t \boldsymbol{w}_t^\top$ such that $\mathbb{E}_t[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$ (where $\mathbb{E}_t[\cdot]$ denotes the conditional expectation with respect the randomness at trial t, conditioned on all the past); then, the algorithm observes ℓ_t and produced an estimate $\widetilde{\boldsymbol{L}}_t$ of the loss matrix $\boldsymbol{L}_t$, with eigenvalues in $[-a, \infty]$, such that $\mathbb{E}_t[\widetilde{\boldsymbol{L}}_t] = \boldsymbol{L}_t + c_t \boldsymbol{I}$ (the estimate is allowed to be biased by a multiplicity of identity matrix!). The expected regret of the algorithm is given by:

Matrix Hedge for Bandit PCA does not work?

W.K.

June 25, 2018

Doesn't work indeed ☹☹☹

... In each trial, the algorithm plays with a ... e chooses a symmetric loss matrix $\boldsymbol{L}_t \in$ ... ives and observes loss $\ell_t = \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$.

We start with a standard bound on the Matrix Hedge algorithm: for any loss sequence $\widetilde{\boldsymbol{L}}_1, \ldots, \widetilde{\boldsymbol{L}}_T$ such that each $\widetilde{\boldsymbol{L}}_t$ has eigenvalues in the range $[-a, \infty)$, the sequence of density matrices $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_T$ produced by Matrix Hedge with fixed learning rate η has regret against a comparator density matrix $\boldsymbol{U}$ upper-bounded by:

$$\mathrm{regret}_T(\boldsymbol{U}) = \sum_{t=1}^{T} \mathrm{tr}((\boldsymbol{W}_t - \boldsymbol{U})\widetilde{\boldsymbol{L}}_t) \leq \frac{\ln d}{\eta} + \kappa(\eta a)\eta \sum_{t=1}^{T} \mathrm{tr}(\boldsymbol{W}_t \widetilde{\boldsymbol{L}}_t^2),$$

where $\kappa(x) = \frac{e^x - x - 1}{x^2}$. The trick is now to use this bound in the bandit case as follows: in each trial $t = 1, \ldots, T$, the algorithm probabilistically chooses $\boldsymbol{w}_t \boldsymbol{w}_t^\top$ such that $\mathbb{E}_t[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$ (where $\mathbb{E}_t[\cdot]$ denotes the conditional expectation with respect the randomness at trial t, conditioned on all the past); then, the algorithm observes ℓ_t and produced an estimate $\widetilde{\boldsymbol{L}}_t$ of the loss matrix $\boldsymbol{L}_t$, with eigenvalues in $[-a, \infty]$, such that $\mathbb{E}_t[\widetilde{\boldsymbol{L}}_t] = \boldsymbol{L}_t + c_t \boldsymbol{I}$ (the estimate is allowed to be biased by a multiplicity of identity matrix!). The expected regret of the algorithm is given by:

Matrix Hedge for Bandit PCA does not work?

W.K.

June 25, 2018

Doesn't work indeed ☹☹☹

In each trial, the algorithm plays with a … e chooses a symmetric loss matrix $\boldsymbol{L}_t \in$ … ives and observes loss $\ell_t = \mathrm{tr}(\boldsymbol{w}_t \boldsymbol{w}_t^\top \boldsymbol{L}_t)$.

We start with a standard bound on the Matrix Hedge algorithm: for any loss sequence $\widetilde{\boldsymbol{L}}_1, \dots, \widetilde{\boldsymbol{L}}_T$ such that each $\widetilde{\boldsymbol{L}}_t$ has eigenvalues in the range $[-a, \infty)$, the sequence of density matrices $\boldsymbol{W}_1, \dots, \boldsymbol{W}_T$ produced by Matrix Hedge with fixed learning rate η has regret against a comparator density matrix $\boldsymbol{U}$

$$\mathrm{regret}_T(\boldsymbol{U}) = \sum_{t=1}^{T} \mathrm{tr}((\boldsymbol{W}_t - \boldsymbol{U})\widetilde{\boldsymbol{L}}_t) \leq \frac{\ln d}{\eta} + \kappa(\eta a)\eta \sum_{t=1}^{T} \mathrm{tr}(\boldsymbol{W}_t \widetilde{\boldsymbol{L}}_t^2),$$

where $\kappa(x) = \frac{e^x - x - 1}{x^2}$. The trick is now to use th… l in the bandit case as follows: in each trial $t = 1, \dots, T$, the algorithm probabilistically … $\mathbb{E}_t[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$ (where $\mathbb{E}_t[\cdot]$ denotes the conditional expectation with res… past); then, the algorithm observes ℓ_t and produce… ues in $[-a, \infty)$ such that $\mathbb{E}_t[\widetilde{\boldsymbol{L}}_t] = \boldsymbol{L}_t + c_t \boldsymbol{I}$ (the e… entity matrix … pected regret of the algorit…

This bound is virtually useless

(for complicated reasons)

'zmeg

The right divergence

$$D(\boldsymbol{W} \| \boldsymbol{U}) = \operatorname{tr}(\boldsymbol{W}\boldsymbol{U}^{-1}) - \log\det(\boldsymbol{W}\boldsymbol{U}^{-1}) - d$$

The Bregman divergence induced by

$$R(\boldsymbol{W}) = -\log\det\boldsymbol{W}$$

a.k.a. Stein's loss (James and Stein, 1967)

The right divergence

$$D(\boldsymbol{W} \| \boldsymbol{U}) = \mathrm{tr}(\boldsymbol{W}\boldsymbol{U}^{-1}) - \log\det(\boldsymbol{W}\boldsymbol{U}^{-1}) - d$$

The Bregman divergence induced by

$$R(\boldsymbol{W}) = -\log\det \boldsymbol{W}$$

a.k.a. Stein's loss (James and Stein, 1967)

The matrix generalization of the trendy "log-barrier" regularizer $-\sum_i \log p_i$

(Foster et al., 2016, Agarwal et al., 2017, Bubeck et al. 2018, Wei and Luo, 2018, Luo et al., 2018, …)

Online Mirror Descent for bandit PCA

loss estimate $\hat{\boldsymbol{L}}_t = ?$

divergence $D = ?$

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W} \in \mathcal{S}} \{\eta \langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t)\}$$

+

sampling scheme?

Sample $\boldsymbol{w}_t$ so that $\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$

Online Mirror Descent for bandit PCA

loss estimate $\hat{\boldsymbol{L}}_t = ?$

divergence $D = ?$

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W} \in \mathcal{S}} \{\eta \langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t)\}$$

+

sampling scheme?

Sample $\boldsymbol{w}_t$ so that

$$\mathbb{E}[\boldsymbol{w}_t \boldsymbol{w}_t^\top] = \boldsymbol{W}_t$$

Main course

Algorithm

Main results

Main result #1: upper bounds

Theorem

$$\text{regret}_T \le \frac{d \log T}{\eta} + \eta d \sum_{t=1}^{T} \|\boldsymbol{L}_t\|_F^2$$

For rank-1 losses:

$$\text{regret}_T = \mathcal{O}\left(d\sqrt{T \log T}\right)$$

In general:

$$\text{regret}_T = \mathcal{O}\left(d^{3/2}\sqrt{T \log T}\right)$$

Main result #2: lower bound

Theorem

There is a problem instance on which any algorithm will suffer

$$\mathrm{regret}_T = \Omega\left(d\sqrt{T/\log T}\right)$$

Dessert | Fast implementation

Implementing the update

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

Implementing the update

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

= by the classic decomposition

$$\widetilde{\boldsymbol{W}}_{t+1} = \arg\min_{\boldsymbol{W}} \left\{ \eta \left\langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

$$\boldsymbol{W}_{t+1} = \arg\min_{W\in\mathcal{S}} D\left(\boldsymbol{W} \| \widetilde{\boldsymbol{W}}_{t+1}\right)$$

Implementing the update

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W} \in \mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

= by the classic decomposition

$$\widetilde{\boldsymbol{W}}_{t+1} = \left(\boldsymbol{W}_t^{-1} + \eta \hat{\boldsymbol{L}}_t \right)^{-1}$$

$$\boldsymbol{W}_{t+1} = \text{renormalize}\left(\widetilde{\boldsymbol{W}}_{t+1} \right)$$

Implementing the update

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}} \left\{ \eta \left\langle \boldsymbol{W}, \widehat{\boldsymbol{L}}_t \right\rangle + D(\boldsymbol{W} \| \boldsymbol{W}_t) \right\}$$

= by the classic decomposition

$$\widetilde{\boldsymbol{W}}_{t+1} = \left(\boldsymbol{W}_t^{-1} + \eta \widehat{\boldsymbol{L}}_t \right)^{-1}$$
$$\boldsymbol{W}_{t+1} = \text{renormalize}\left(\widetilde{\boldsymbol{W}}_{t+1} \right)$$

takes $\mathcal{O}(d^3)$ time in general

Implementing the update

$$\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W}\in\mathcal{S}}\left\{\eta\langle \boldsymbol{W}, \hat{\boldsymbol{L}}_t\rangle + D(\boldsymbol{W}\|\boldsymbol{W}_t)\right\}$$

= by the classic decomposition

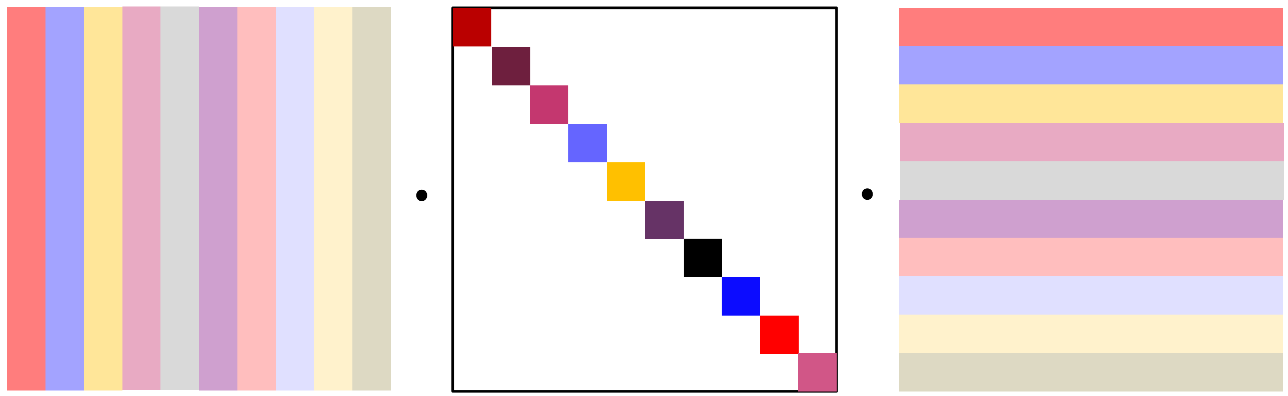
$$\widetilde{\boldsymbol{W}}_{t+1} = \left(\boldsymbol{W}_t^{-1} + \eta\hat{\boldsymbol{L}}_t\right)^{-1}$$
$$\boldsymbol{W}_{t+1} = \text{renormalize}\left(\widetilde{\boldsymbol{W}}_{t+1}\right)$$

takes $\mathcal{O}(d^3)$ time in general

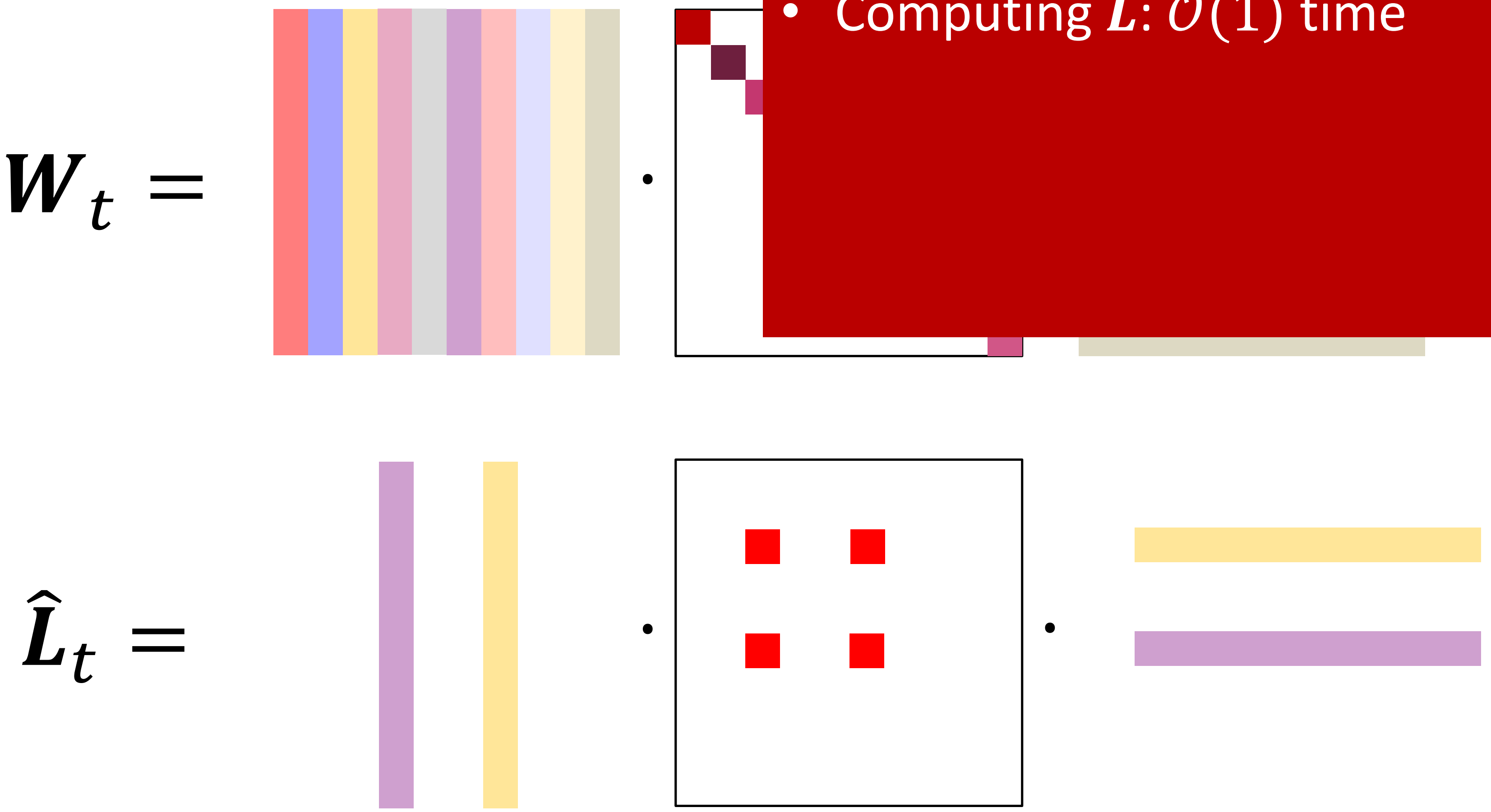
BUT ONLY $\mathcal{O}(d)$ TIME IN OUR CASE!!

Updating in $\mathcal{O}(d)$ time

$\boldsymbol{W}_t =$



Updating in $\mathcal{O}(d)$ time



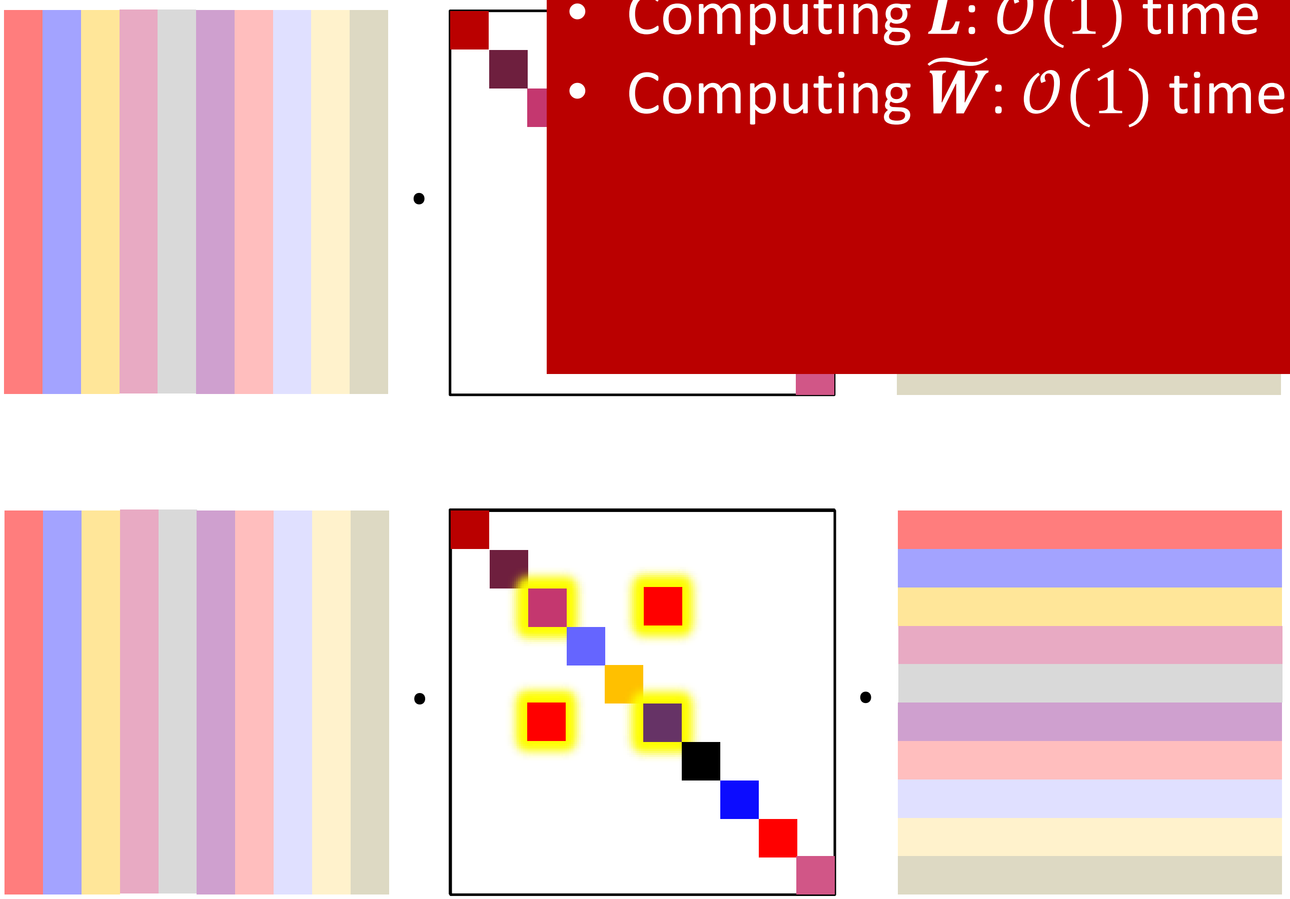
- Computing $\hat{\boldsymbol{L}}$: $\mathcal{O}(1)$ time

Updating in $\mathcal{O}(d)$ time

- Computing $\widehat{\boldsymbol{L}}$: $\mathcal{O}(1)$ time
- Computing $\widetilde{\boldsymbol{W}}$: $\mathcal{O}(1)$ time

$\boldsymbol{W}_t =$

$\widetilde{\boldsymbol{W}}_{t+1} =$

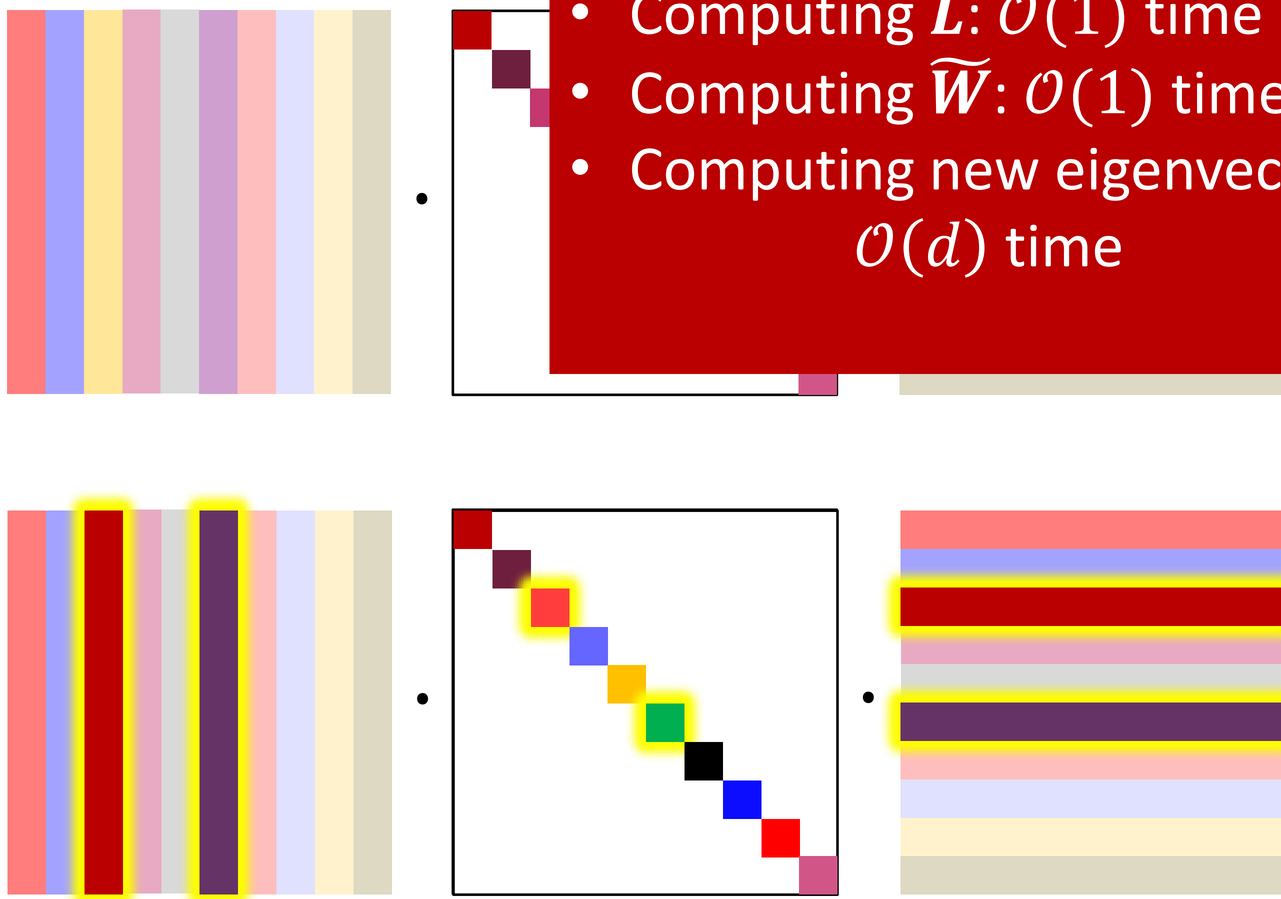


Updating in $\mathcal{O}(d)$ time

- Computing $\widehat{\boldsymbol{L}}$: $\mathcal{O}(1)$ time
- Computing $\widetilde{\boldsymbol{W}}$: $\mathcal{O}(1)$ time
- Computing new eigenvectors: $\mathcal{O}(d)$ time

$\boldsymbol{W}_t =$

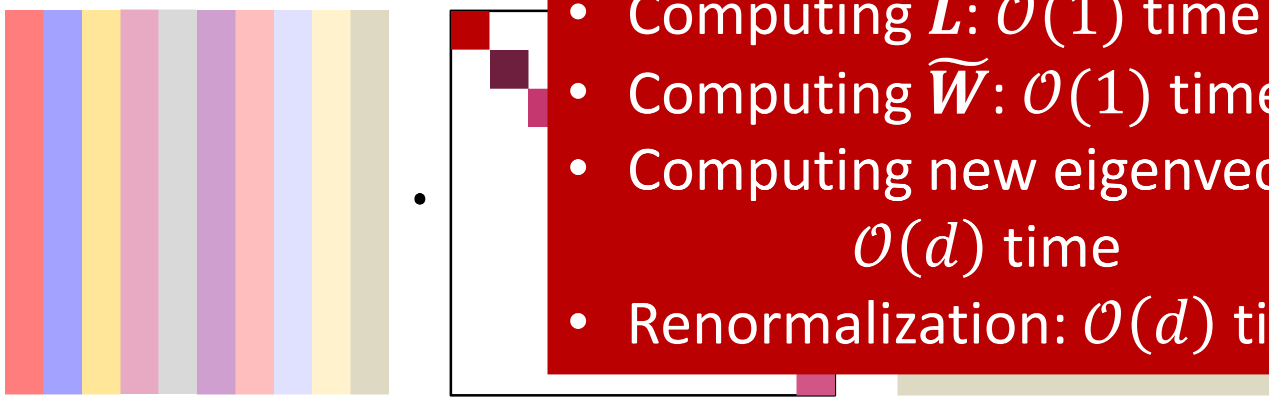
$\widetilde{\boldsymbol{W}}_{t+1} =$



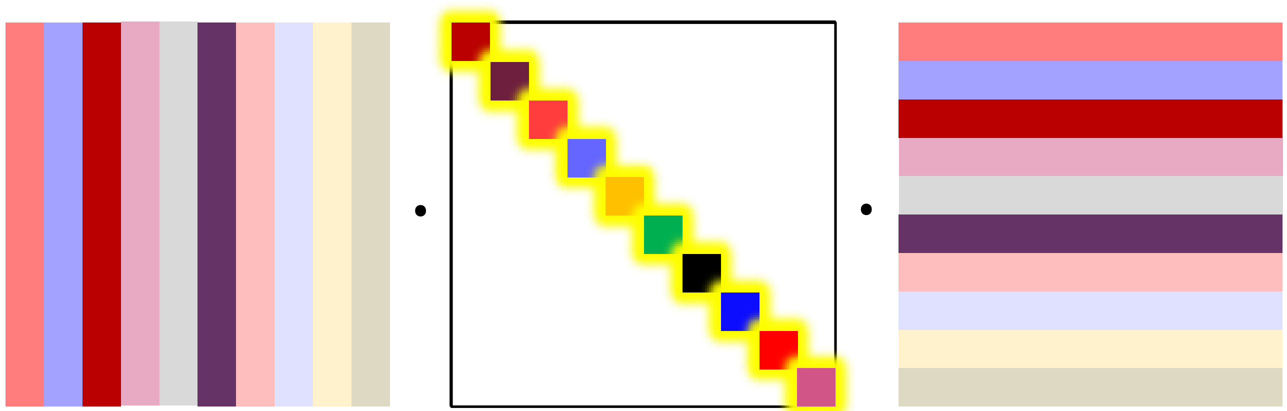
Updating in $\mathcal{O}(d)$ time

- Computing $\widehat{\boldsymbol{L}}$: $\mathcal{O}(1)$ time
- Computing $\widetilde{\boldsymbol{W}}$: $\mathcal{O}(1)$ time
- Computing new eigenvectors: $\mathcal{O}(d)$ time
- Renormalization: $\mathcal{O}(d)$ time

$\boldsymbol{W}_t =$



$\widetilde{\boldsymbol{W}}_{t+1} =$



Updating in $\mathcal{O}(d)$ time

1) time

(1) time

eigenvectors:

ne

$\mathcal{O}(d)$ time

$W_t =$

UPDATING TAKES LESS TIME THAN READING THE FULL LOSS MATRIX L_t!!!

$\widetilde{W}_{t+1} =$

Summary

	Previous best	Our work
Runtime	no polytime?	d
Upper bound	$d^2\sqrt{T}$	$d^{3/2}\sqrt{T}$
Lower bound	$\sqrt{dT}$	$d\sqrt{T}$

Summary

	Previous best	Our work
Runtime	no polytime?	d
Upper bound		$d^{3/2}\sqrt{T}$
Lower bound		$d\sqrt{T}$

Still a gap of $\sqrt{d}$

☹☹☹☹

Open problem: d or $d^{3/2}$?

d looks obvious, right?

Open problem: d or $d^{3/2}$?

d looks obvious, right?

- multi-armed bandits:
  d parameters to estimate $\Rightarrow \sqrt{dT}$ regret
- bandit PCA:
  d^2 parameters to estimate $\Rightarrow d\sqrt{T}$ regret?

Open problem: d or $d^{3/2}$?

d looks obvious, right?

- multi-armed bandits:
  d parameters to estimate $\Rightarrow \sqrt{dT}$ regret
- bandit PCA:
  d^2 parameters to estimate $\Rightarrow d\sqrt{T}$ regret?

NO:

Lemma:

For i.i.d. data, every non-adaptive algorithm will have error at least

$$\Omega\left(\frac{d^{3/2}}{\sqrt{T}}\right)$$

Open problem: d or $d^{3/2}$?

If true dependence is $\Theta(d)$:
First known case with a gap between non-adaptive and adaptive algorithms!!!

NO:

Lemma:
For i.i.d. data, every non-adaptive algorithm will have error at least

$$\Omega\left(\frac{d^{3/2}}{\sqrt{T}}\right)$$

Open problem: faster rates for phase retrieval

Our bound for PR:

$$O\left(\frac{d}{\sqrt{T}}\right)$$

SOTA for PR:

$$O\left(\frac{d}{T}\right)$$

Open problem: faster rates for phase retrieval

Our bound for PR:

$$O\left(\frac{d}{\sqrt{T}}\right)$$

SOTA for PR:

$$O\left(\frac{d}{T}\right)$$

Why such a big gap?

Open problem:
faster rates for phase retrieval

Our bound for PR:

$$O\left(\frac{d}{\sqrt{T}}\right)$$

SOTA for PR:

$$O\left(\frac{d}{T}\right)$$

Why such a big gap?

- i.i.d. assumption
- spiked covariance model

Can we exploit these to obtain even better rates?

Thanks!