What is the best algorithm for online PCA?

Jiazhong Nie ¹
Wojciech Kotłowski ²
Manfred K. Warmuth ¹

¹University of California - Santa Cruz

²Poznań University of Technology, Poland

October 23, 2013
Overview of this talk

Our setting:

- **Online PCA**: Compress the data of \( \mathbb{R}^n \) into \( k \) dimensional subspace in an online manner almost as good as the best \( k \)-dimensional subspace in hindsight.

We analyze two algorithms:

- **Additive update**: Gradient Descent (GD)
- **Multiplicative update**: Exponentiated Gradient (EG)
Overview of this talk

Our contributions:

1. We show that (surprisingly) both GD and EG essentially optimal for the worst-case data sequence.

2. However, when there is a good $k$ subspace approximating the data well, EG remains optimal, while GD becomes suboptimal.

3. In the “full rank” extension of PCA, EG remains optimal, GD is suboptimal already for the worst-case data sequence.
Our contributions:

1. We show that (surprisingly) both GD and EG essentially optimal for the worst-case data sequence.

2. However, when there is a good $k$ subspace approximating the data well, EG remains optimal, while GD becomes suboptimal.

3. In the “full rank” extension of PCA, EG remains optimal, GD is suboptimal already for the worst-case data sequence.

Take-home message:

- Use EG for PCA
Outline

1. Introduction to online PCA
2. Algorithms and regret bounds
3. Generalization and open problems
Outline

1. Introduction to online PCA
2. Algorithms and regret bounds
3. Generalization and open problems
Introduction to online PCA

Principle Component Analysis (PCA)

- Given data in unit ball of $\mathbb{R}^n$, find the $k$ dimension subspace with the least compression loss

$$\inf_{\text{projection matrix } P \text{ of rank } k} \sum_{t=1}^{T} \left\| x_t - Px_t \right\|^2$$

Nie, Kotłowski & Warmuth (UCSC & PUT)

What is the best algorithm for online PCA?
Principle Component Analysis (PCA)

Given data in unit ball of $\mathbb{R}^n$, finds the $k$ dimension subspace with the least compression loss

$$\inf_{\text{projection matrix } P} \sum_{t=1}^{T} \left\| x_t - Px_t \right\|^2$$

- $P^*$: $k$ largest eigen-direction of data matrix $\sum_t x_t x_t^\top$

Nie, Kotłowski & Warmuth (UCSC & PUT)
Online PCA

- Data points produced on-line and change over time
- Online protocol
  - Before each point $x_t$, learner predicts a projection $P_t$ probabilistically
  - Suffers expected compression loss $\mathbb{E} \left[ \|x_t - P_tx_t\|^2 \right]$
- Goal: small regret

$$R = \sum_{t=1}^{T} \mathbb{E} \left[ \|x_t - P_tx_t\|^2 \right] - \sum_{t=1}^{T} \|x_t - P^*x_t\|^2$$

- total expected loss of learner
- loss of best $k$ subspace

Nie, Kotłowski & Warmuth (UCSC & PUT) What is the best algorithm for online PCA?
Online PCA

- Data points produced on-line and change over time
- Online protocol
  - Before each point $x_t$, learner predicts a projection $P_t$ probabilistically
  - Suffers expected compression loss $\mathbb{E} \left[ \|x_t - P_t x_t\|^2 \right]$
- Goal: small regret

$$R = \sum_{t=1}^{T} \mathbb{E} \left[ \|x_t - P_t x_t\|^2 \right] - \sum_{t=1}^{T} \|x_t - P^* x_t\|^2$$

- total expected loss of learner
- loss of best $k$ subspace

$\approx$ sum of $n - k$ smallest eigenvalue of $\sum_t x_t x_t^T$
Simplify the expected loss of learner

- Compression loss is a dot product
  \[
  \|x - Px\|^2 = \text{tr} \left( (I - P) (xx^T) \right) = (I - P) \cdot xx^T
  \]
  \[
  = \text{matrix dot product}
  \]

- Expected compression loss (w.r.t randomized prediction \(P\))
  \[
  \mathbb{E} \left[ (I - P) \cdot X \right] = \mathbb{E} \left[ I - P \right] \cdot X
  \]

- For any randomized prediction \(P\), \(\mathbb{E} \left[ I - P \right]\) determines the loss
Use $\mathbb{E}[I - P]$ as learner’s parameter

At each trial $t$, the learner

- Determines a parameter matrix $W_t$
- Sample a random rank $k$ projection matrix $P_t$ with $\mathbb{E}[I - P_t] = W_t$
- Predict with $P_t$ and suffer loss $W_t \cdot X_t$
Use $\mathbb{E} [ I - P ]$ as learner’s parameter

At each trial $t$, the learner

- Determines a parameter matrix $W_t$
- Sample a random rank $k$ projection matrix $P_t$ with $\mathbb{E} [ I - P_t ] = W_t$
- Predict with $P_t$ and suffer loss $W_t \cdot X_t$

A simple dot product loss function!

Same as the Expert setting
Use $\mathbb{E}[I - P]$ as learner’s parameter

At each trial $t$, the learner

- Determines a parameter matrix $W_t$
- Sample a random rank $k$ projection matrix $P_t$ with $\mathbb{E}[I - P_t] = W_t$
- Predict with $P_t$ and suffer loss $W_t \cdot X_t$

A simple dot product loss function!

Same as the Expert setting

What is the learner’s parameter set?

$$\mathcal{W} = \{ W \in R^{n \times n} \mid 0 \preceq W \preceq I \quad \text{tr}(W) = m \}, \text{ where } m = n - k$$
Connection to other settings with “dot loss”

<table>
<thead>
<tr>
<th>Learning problem</th>
<th>Action per trial</th>
<th>Parameter space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert setting</td>
<td>choose one expert from $n$ experts</td>
<td>$\mathbf{w} \in \mathbb{R}^n$ s.t. $\sum w_i = 1$, $0 \leq w_i$</td>
</tr>
<tr>
<td>$m$-set problem</td>
<td>choose $m$ expert from $n$ experts</td>
<td>$\mathbf{w} \in \mathbb{R}^n$ s.t. $\sum w_i = m$, $0 \leq w_i \leq 1$</td>
</tr>
<tr>
<td>Online PCA</td>
<td>choose $m$ dim. subspace from $n$ dim. space</td>
<td>$\mathbf{W} \in \mathbb{R}^{n \times n}$ s.t. $\text{tr}(\mathbf{W}) = m$, $0 \preceq \mathbf{W} \preceq I$</td>
</tr>
</tbody>
</table>

Expert setting is a **special case** of $m$-set problem when $m = 1$ ($k = n - 1$).

$m$-set problem is a **special case** of online PCA if data is orthogonal.
Outline

1. Introduction to online PCA
2. Algorithms and regret bounds
3. Generalization and open problems
Algorithm descent (GD) algorithm

- At trial $t$, an algorithm (learner) determines $\mathbf{W}_t$ based on $\mathbf{X}_{1...t-1}$
- GD algorithm update $\mathbf{W}_t$ as

\[
\mathbf{W}_t = \inf_{\mathbf{W} \in \mathcal{W}} \eta \mathbf{W} \cdot \mathbf{X}_{t-1} + \| \mathbf{W} - \mathbf{W}_{t-1} \|^2_F,
\]

where $\| \mathbf{W} \|_F$ is the Frobenius norm and $\eta$ is learning rate
- Equivalent to

Descent step: $\hat{\mathbf{W}}_t = \mathbf{W}_{t-1} - \eta \mathbf{X}_{t-1}$

Projection step: $\mathbf{W}_t = \inf_{\mathbf{W} \in \mathcal{W}} \| \mathbf{W} - \hat{\mathbf{W}}_t \|^2_F$
Matrix exponential gradient (MEG) algorithm

- MEG algorithm updates $W_t$ as

  $$W_t = \inf_{W \in \mathcal{W}} \eta W \cdot X_{t-1} + \Delta(W, W_{t-1})$$

- $\Delta(W, W_t)$ is Quantum Relative Entropy.

  $$\Delta(W, W_t) = W \cdot (\log W - \log W_t)$$

  where $\log$ is matrix version of logarithm

- Belongs to the same exponential gradient family as weighted majority alg., winnow alg., etc.
Worst case regret upper bounds \[\text{[KW97, WK08]}\]

For any data seq. the regrets of GD and MEG are upper bounded by

\[
\begin{align*}
R_{GD} & \leq \sqrt{2\frac{m}{n}kT} \\
R_{MEG} & \leq \sqrt{2m \log \frac{n}{m}} \text{ loss of best k subspace } + m \log \frac{n}{m}
\end{align*}
\]
For any data seq. the regrets of GD and MEG are upper bounded by

\[ R_{GD} \leq \sqrt{\frac{2m}{n} kT} \quad R_{MEG} \leq \sqrt{2m \log \frac{n}{m}} \frac{\text{loss of best } k \text{ subspace}}{m} + m \log \frac{n}{m} \]

loss of best 

\[ k \text{ subspace} = \text{sum of } m \text{ smallest eigenvalues of } \sum_t X_t \leq \frac{m}{n} \left( \text{sum of all eigenvalues of } \sum_t X_t \right) \]

\[ = \frac{m}{n} \text{tr} (\sum_t X_t) = \text{tr}(x_t x_t^\top) \leq 1 \leq \frac{m}{n} T \]
For any data seq. the regrets of GD and MEG are upper bounded by

$$R_{GD} \leq \sqrt{2 \frac{m}{n} kT} \quad \quad \quad R_{MEG} \leq \sqrt{2m \log \frac{n}{m}} \left( \text{loss of best } k \text{ subspace} \right) + m \log \frac{n}{m}$$

loss of best $k$ subspace = sum of $m$ smallest eigenvalues of $\sum_t X_t \leq \frac{m}{n} \left( \text{sum of all eigenvalues of } \sum_t X_t \right)$

$$= \frac{m}{n} \text{tr} (\sum_t X_t) \leq \frac{m}{n} T$$

Both bounds are $O(\sqrt{T})$ but dependence on $n$ and $k$ may differ
Comparing dependence on $n$ and $k$

Regret bounds:

\[ \mathcal{R}_{GD} \leq \sqrt{2 \frac{m}{n} k T} \]
\[ \mathcal{R}_{MEG} \leq \sqrt{2 \frac{m}{n} m \log \frac{n}{m} T + o(\sqrt{T})} \]

• Recall in online PCA, $n = m + k$ and $n \gg k$ which mean $\frac{m}{n} \approx 1$
Comparing dependence on $n$ and $k$

Regret bounds:

$$\mathcal{R}_{GD} \leq \sqrt{2 \frac{m}{n} k T} \quad \mathcal{R}_{MEG} \leq \sqrt{2 \frac{m}{n} m \log \frac{n}{m} T + o(\sqrt{T})}$$

- Recall in online PCA, $n = m + k$ and $n \gg k$ which mean $\frac{m}{n} \approx 1$
- **Constant** v.s. **logarithmic** dependence on $n$? GD better than EG?
Comparing dependence on $n$ and $k$

Regret bounds:

$$R_{GD} \leq \sqrt{2 \frac{m}{n} k T} \quad \quad R_{MEG} \leq \sqrt{2 \frac{m}{n} m \log \frac{n}{m} T + o(\sqrt{T})}$$

- Recall in online PCA, $n = m + k$ and $n \gg k$ which mean $\frac{m}{n} \approx 1$
- **Constant** v.s. **logarithmic** dependence on $n$? GD better than EG?
- No.

$$m \log \frac{n}{m} = m \log(1 + \frac{k}{m}) \leq m \frac{k}{m} = k$$
Comparing dependence on $n$ and $k$

Regret bounds:

\[ R_{GD} \leq \sqrt{2 \frac{m}{n} k} \quad T \]
\[ R_{MEG} \leq \sqrt{2 \frac{m}{n} m \log \frac{n}{m}} \quad T + o(\sqrt{T}) \]

- Recall in online PCA, $n = m + k$ and $n \gg k$ which mean $\frac{m}{n} \approx 1$
- **Constant** v.s. **logarithmic** denominator on $n$? GD better than EG?
- No.

\[ m \log \frac{n}{m} = m \log(1 + \frac{k}{m}) \leq m \frac{k}{m} = k \]

**Asymptotically the same! End of story?**
Comparing dependence on $n$ and $k$

Regret bounds:

\[ R_{GD} \leq \sqrt{2 \frac{m}{n} k} \ T \quad \quad R_{MEG} \leq \sqrt{2 \frac{m}{n} m \log \frac{n}{m}} \ T + o(\sqrt{T}) \]

- Recall in online PCA, $n = m + k$ and $n \gg k$ which mean $\frac{m}{n} \approx 1$
- **Constant** v.s. **logarithmic** dependence on $n$? GD better than EG?
- No.

\[ m \log \frac{n}{m} = m \log \left(1 + \frac{k}{m} \right) \leq m \frac{k}{m} = k \]

Asymptotically the same! End of story? No!
Loss dependent regret bound

- Recall the original MEG regret bound

$$\mathcal{R}_{MEG} \leq \sqrt{2m \log \frac{n}{m}} \text{ loss of best } k \text{ subspace } + m \log \frac{n}{m} \leq \sqrt{2k \text{ loss of best } k \text{ subspace}} + k$$
Loss dependent regret bound

- Recall the original MEG regret bound

\[
R_{\text{MEG}} \leq \sqrt{2m \log \frac{n}{m}} \text{ loss of best } k \text{ subspace } + m \log \frac{n}{m} \leq \sqrt{2k} \text{ loss of best } k \text{ subspace } + k
\]

- Loss dependent bound are more interesting. In real online PCA, there is usually a \( k \) subspace which approximates the data well

loss of best \( k \) subspace \( \ll \frac{m}{n}T \)

What is GD’s loss dependent bound?
Loss dependent regret bound

- Recall the original MEG regret bound

\[ R_{MEG} \leq \sqrt{2m \log \frac{n}{m}} \text{ loss of best } k \text{ subspace } + m \log \frac{n}{m} \leq \sqrt{2k \text{ loss of best } k \text{ subspace } + k} \]

- Loss dependent bound are more interesting. In real online PCA, there is usually a \( k \) subspace which approximates the data well

\[ \text{loss of best } k \text{ subspace } \ll \frac{m}{n} T \]

- What is GD’s loss dependent bound?

- [Thm. 2] gives the following lower bound for GD

\[ R_{GD} \geq k\sqrt{\text{loss of best } k \text{ subspace}} \]
Loss dependent regret bound

- Recall the original MEG regret bound

\[ R_{MEG} \leq \sqrt{2m \log \frac{n}{m}} \text{ loss of best } k \text{ subspace } + m \log \frac{n}{m} \leq \sqrt{2k} \text{ loss of best } k \text{ subspace } + k \]

- Loss dependent bound are more interesting. In real online PCA, there is usually a \( k \) subspace which approximates the data well

\[ \text{loss of best } k \text{ subspace } \ll \frac{m}{n} T \]

What is GD’s loss dependent bound?

- [Thm. 2] gives the following lower bound for GD

\[ R_{GD} \geq k \sqrt{\text{loss of best } k \text{ subspace}} \]

MEG is better than GD by a factor of \( \sqrt{k} \)
Regret lower bound for any algorithm

- **[Thm. 3]**: Given $n$, $k \leq \frac{n}{2}$ and any algorithm for online PCA, there are data sequence incurring regret $\Omega(\sqrt{kT})$ and $\Omega(\sqrt{k} \text{ loss of best } k \text{ subspace})$

- Proved with the special case, $m$-set problem
  A problem is harder than its special case

- Matches MEG upper bounds up to constant factors
  MEG is asymptotically optimal in both types of bounds
Outline

1. Introduction to online PCA
2. Algorithms and regret bounds
3. Generalization and open problems
Generalization: data matrix with full rank

- In online PCA, data matrix $X_t = x_t x_t^T$ has rank one
- What if $X_t$ has full rank with max eigenvalue 1?
  Loss dependent regret upper bound still hold
  
  $$\text{loss of best } k \text{ subspace} \leq mT$$

- Plugging into loss dependent bound gives $m \sqrt{\log \frac{n}{mT}}$. Also optimal?
In online PCA, data matrix $X_t = x_t x_t^\top$ has rank one

What if $X_t$ has full rank with max eigenvalue 1?

Loss dependent regret upper bound still hold

$$\text{loss of best } k \text{ subspace } \leq mT$$

Plugging into loss dependent bound gives $m \sqrt{\log \frac{n}{m} T}$. Also optimal? No. When $k \ll n$, flipped MEG gives smaller regret [Thm. 1]

$$k \sqrt{\log \frac{n}{k} T}$$

Flipped MEG: regularized with $\Delta_{flipped}(W, U) = \Delta(I - W, I - U)$
## Summary

- **For time dependent bound**

<table>
<thead>
<tr>
<th>Rank one data mat.</th>
<th>Optimal Bound</th>
<th>Optimal Alg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m \sqrt{\log \frac{n}{m} \frac{T}{n}}$</td>
<td>MEG</td>
</tr>
<tr>
<td></td>
<td>$= \Theta(\sqrt{kT})$ when $k \ll n$ (Online PCA)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Full rank data mat.</th>
<th>Optimal Bound</th>
<th>Optimal Alg</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k \geq \frac{n}{2}$:</td>
<td>$m \sqrt{\log \frac{n}{m} T}$</td>
<td>MEG</td>
</tr>
<tr>
<td>$k \leq \frac{n}{2}$:</td>
<td>$k \sqrt{\log \frac{n}{k} T}$</td>
<td>flipped MEG</td>
</tr>
</tbody>
</table>

- **For loss dependent bound**, MEG is always optimal with bound

$$\mathcal{R}_{MEG} = \Theta\left(\sqrt{m \log \frac{n}{m} \text{ loss of best } k \text{ subspace}}\right)$$
Open questions

- We posed the question whether "non-forgetting" GD is also suboptimal in loss dependent bound
  Later found: yes, it is also suboptimal (not shown in the paper)
- Can GD with variable learning rate (e.g. Nesterovs Accelerated GD) achieve the optimum regret?
  OR
  Lower bounds against predicting with linear combination of data
- Both GD and EG are slow: need full eigen-decomposition of data covariance matrix in each trial even $k \ll n$
  Is there any faster algorithm? e.g. Follow the perturbed leader?