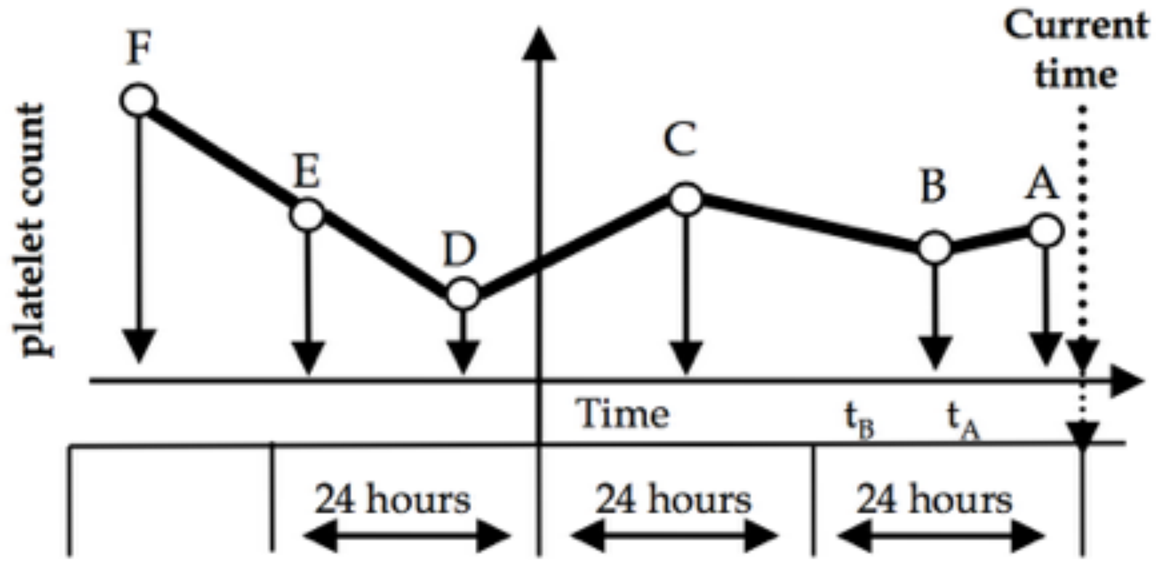
10 YEARS ROAD TO SQUEAK

SequeL, Inria Lille - Nord Europe

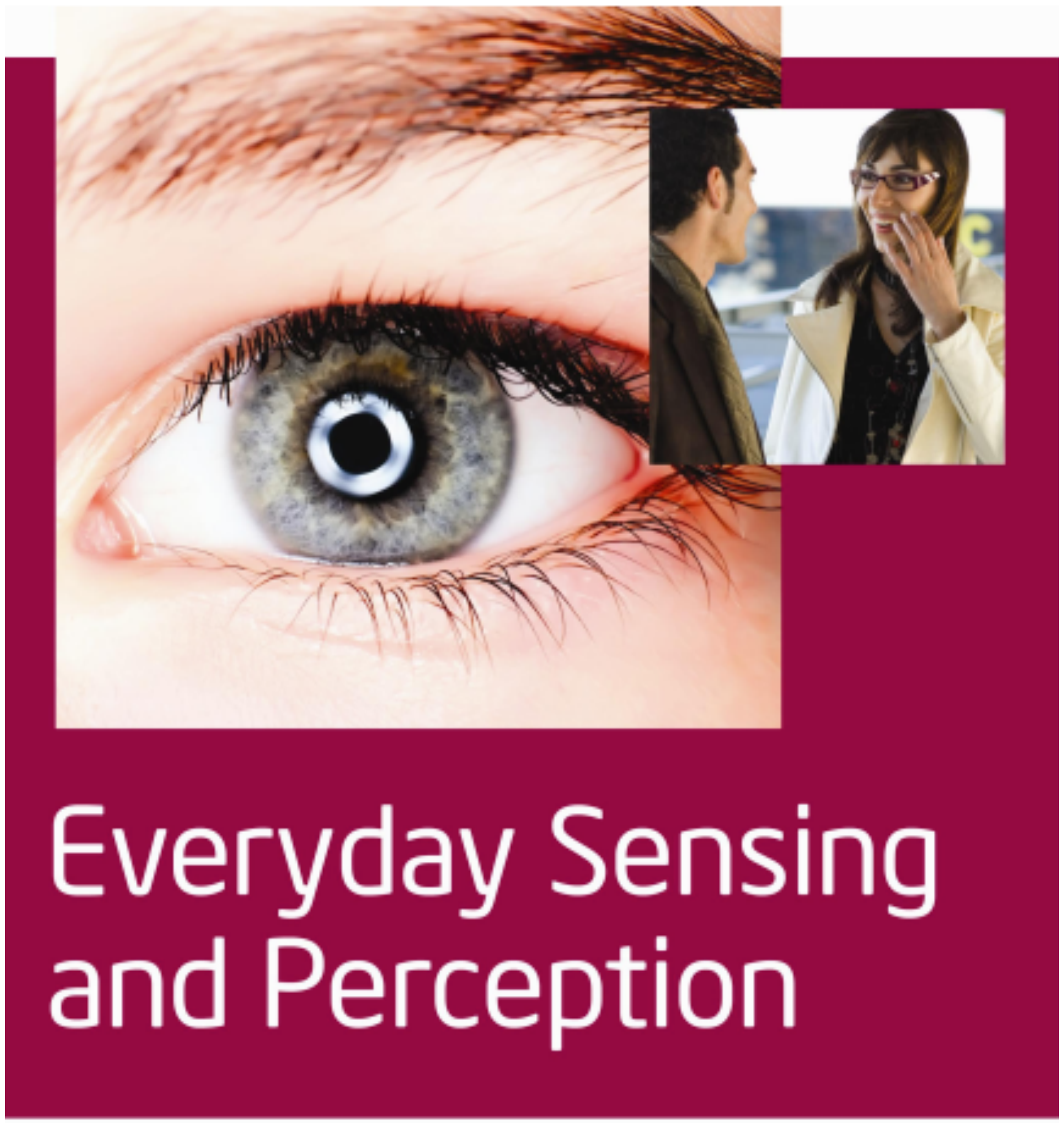
10 YEARS ROAD TO SQUEAK AND QUADRATIC BARRIER

ONLINE GRAPH-BASED ANOMALY DETECTION

- medical data
- graph on patient states
- labels are the medical action
- goal: online detection of anomalous data



EVERYDAY SENSING AND PERCEPTION



EVERYDAY SENSING AND PERCEPTION

informatics mathematics Inria

Intel Research Berkeley

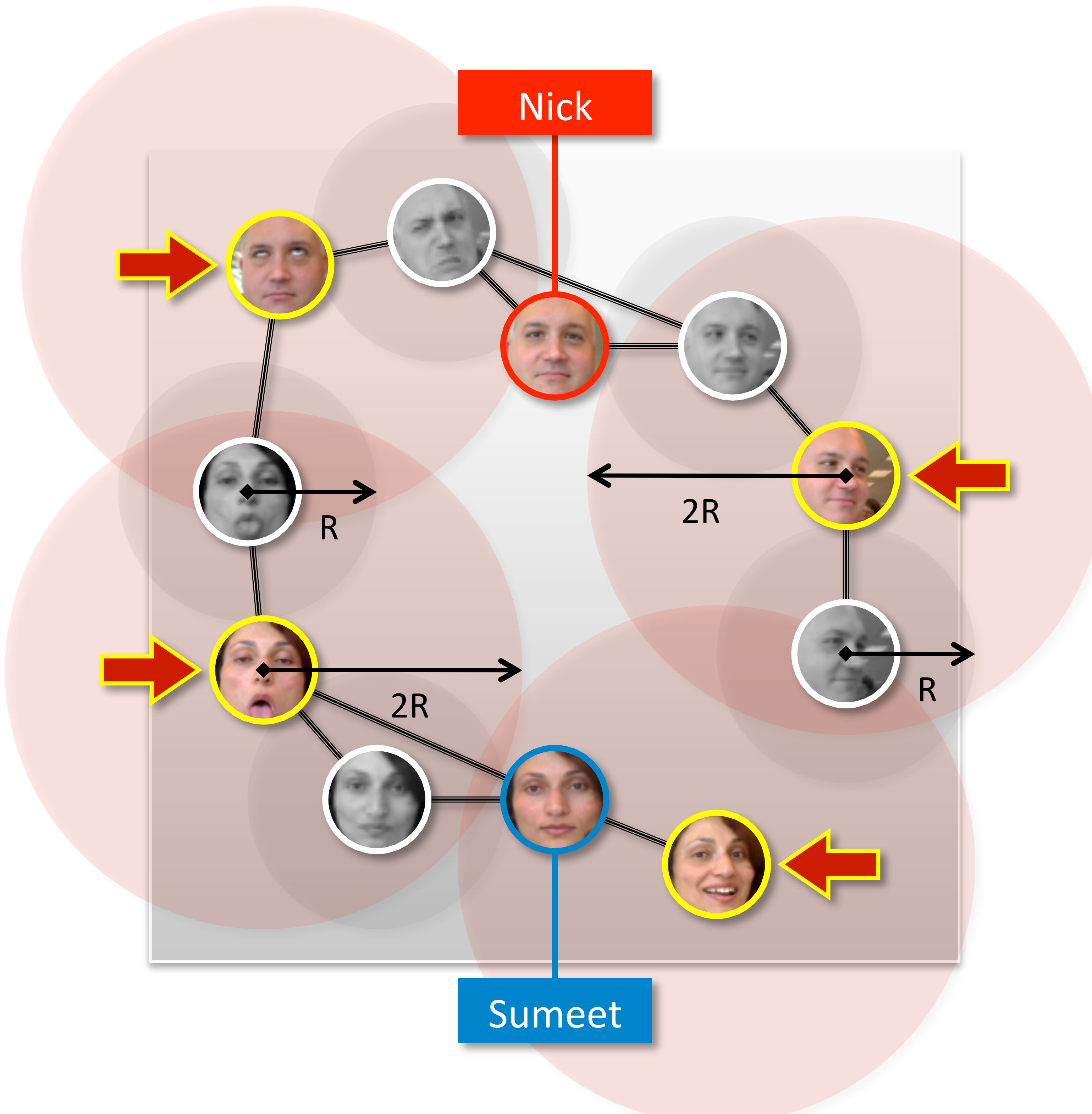
intel

WWW.INTEL.COM INTEL RESEARCH INTEL RESEARCH BERKELEY EVERYDAY SENSING & PERCEPTION

Online Semi-Supervised Learning and Face Recognition

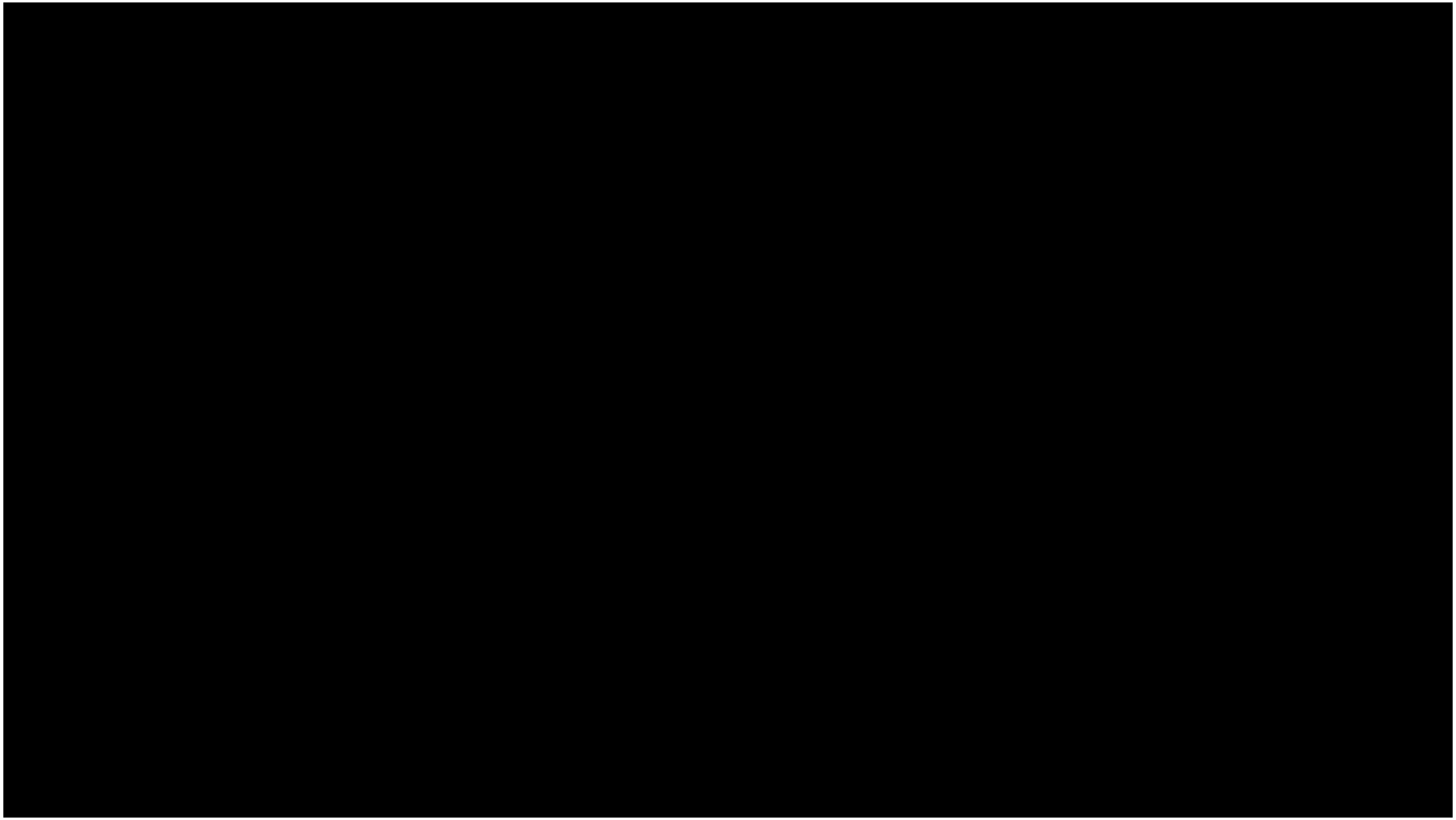
This project focuses on real-time learning without explicit feedback. This work combines the ideas of semi-supervised learning on approximate graphs and online learning. In particular, we develop algorithms that iteratively build a graphical representation of the world and update it on-the-fly with observed examples (both labeled and unlabeled). We proved regret bounds of the solutions, demonstrated that the system can recognize faces in real-time even in a resource constraint environment and can take advantage of the manifold structure to outperform existing methods. The following videos show how online semi-supervised learning can be used to train a robust face recognizer of a person from just a single frontal image:

ONLINE K-CENTER CLUSTERING



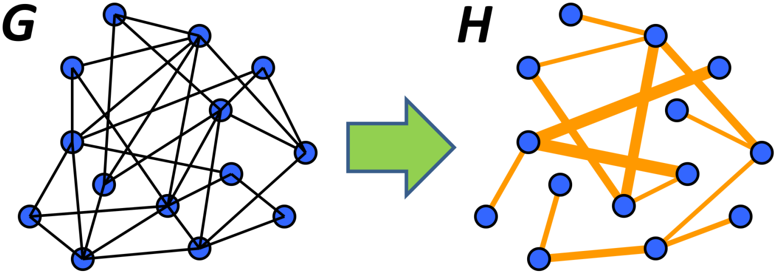
13

INTEL AD FOR THE ONLINE FACE RECO



Graph Sparsification

Goal: Get graph G and find sparse H



Graph Sparsification: What is sparse?

What does sparse graph mean?

- average degree < 10 is pretty sparse

Graph Sparsification: What is sparse?

What does sparse graph mean?

- average degree < 10 is pretty sparse
- for billion nodes even 100 should be ok

Graph Sparsification: What is sparse?

What does **sparse** graph mean?

- average degree < 10 is pretty sparse
- for billion nodes even 100 should be ok
- in general: average degree $<$ polylog n

Graph Sparsification: What is sparse?

What does **sparse** graph mean?

- average degree < 10 is pretty sparse
- for billion nodes even 100 should be ok
- in general: average degree $<$ polylog n

Graph Sparsification: What is sparse?

What does sparse graph mean?

- average degree < 10 is pretty sparse
- for billion nodes even 100 should be ok
- in general: average degree $<$ polylog n

Are all edges important?

Graph Sparsification: What is sparse?

What does sparse graph mean?

- average degree < 10 is pretty sparse
- for billion nodes even 100 should be ok
- in general: average degree $<$ polylog n

Are all edges important?

in a tree — sure, in a dense graph perhaps not

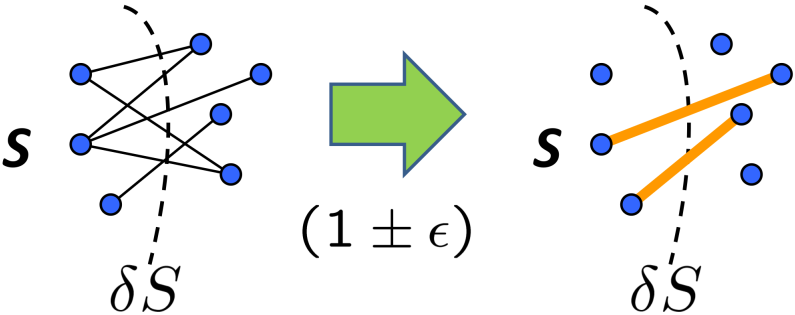
Graph Sparsification: What is good sparse?

Graph Sparsification: What is **good** sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

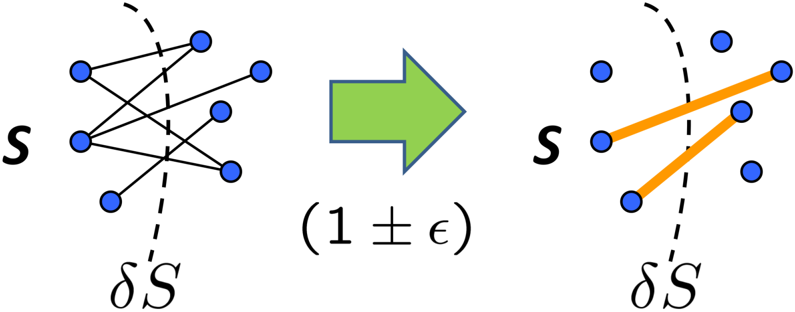
Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!



Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!



H approximates G well iff $\forall S \subset V$, sum of edges on δS remains

δS = edges leaving S

https://math.berkeley.edu/~nikhil/

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care?

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Recall what is a cut: $\text{cut}_G(S) =$

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Recall what is a cut: $\text{cut}_G(S) = \sum_{i \in S, j \in \overline{S}} w_{i,j}$

Graph Sparsification: What is **good** sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Recall what is a cut: $\text{cut}_G(S) = \sum_{i \in S, j \in \overline{S}} w_{i,j}$

Define G and H are $(1 \pm \varepsilon)$-**cut similar** when $\forall S$

$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Recall what is a cut: $\text{cut}_G(S) = \sum_{i \in S, j \in \overline{S}} w_{i,j}$

Define G and H are $(1 \pm \varepsilon)$**-cut similar** when $\forall S$

$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

Is this always possible?

Graph Sparsification: What is good sparse?

Good sparse by Benczúr and Karger (1996) = **cut preserving**!

Why did they care? faster mincut/maxflow

Recall what is a cut: $\text{cut}_G(S) = \sum_{i \in S, j \in \overline{S}} w_{i,j}$

Define G and H are $(1 \pm \varepsilon)$-**cut similar** when $\forall S$

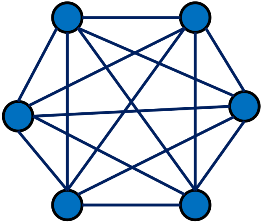
$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

Is this always possible? Benczúr and Karger (1996): Yes!

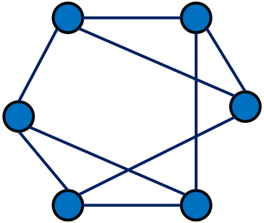
$\forall \varepsilon \; \exists \; (1+\varepsilon)$-cut similar $\widetilde{G}$ with $\mathcal{O}(n \log n / \varepsilon^2)$ edges s.t. $E_H \subseteq E$ and computable in $\mathcal{O}(m \log^3 n + m \log n / \varepsilon^2)$ time n nodes, m edges

Graph Sparsification: What is **good** sparse?

$G = K_n$

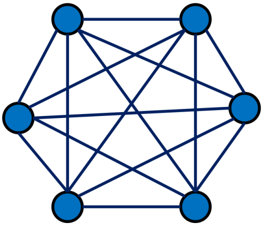


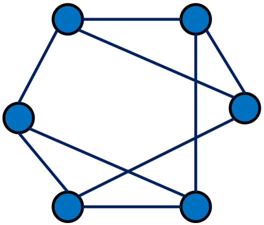
$H = d$-**regular** (random)



Graph Sparsification: What is **good** sparse?

$G = K_n$ $H =$ d-**regular** (random)

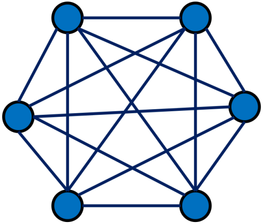




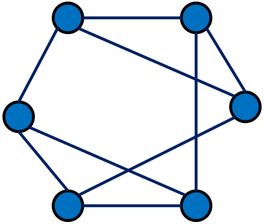
How many edges?

Graph Sparsification: What is good sparse?

$G = K_n$



$H = d$-**regular** (random)

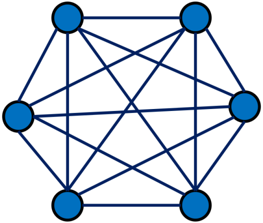


How many edges?

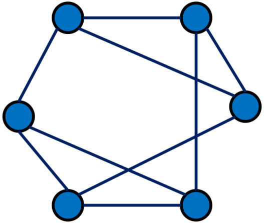
$$|E_G| = \mathcal{O}(n^2)$$

Graph Sparsification: What is **good** sparse?

$G = K_n$



$H =$ d-**regular** (random)



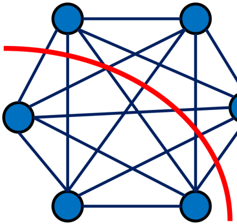
How many edges?

$|E_G| = \mathcal{O}(n^2)$

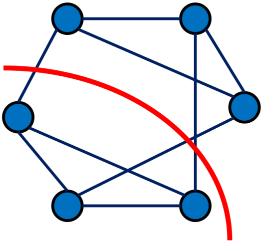
$|E_H| = \mathcal{O}(dn)$

Graph Sparsification: What is good sparse?

$G = K_n$

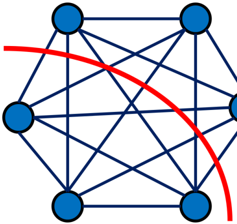


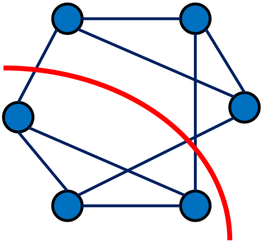
$H = d$-**regular** (random)



Graph Sparsification: What is good sparse?

$G = K_n$ $\qquad$ $H = d$-**regular** (random)

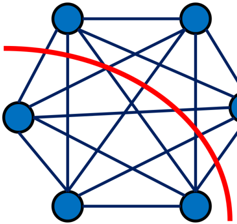




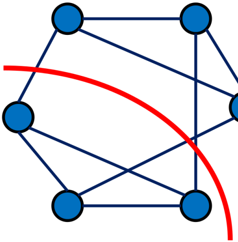
What are the cut weights for any S?

Graph Sparsification: What is **good** sparse?

$G = K_n$



$H = d$-**regular** (random)

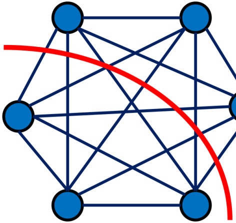


What are the cut weights for any S?

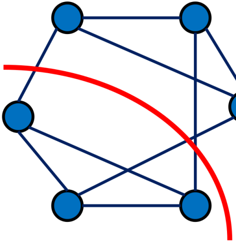
$$w_G(\delta S) = |S| \cdot |\overline{S}|$$

Graph Sparsification: What is **good** sparse?

$G = K_n$



$H =$ d-**regular** (random)



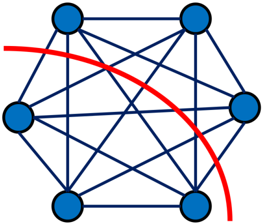
What are the cut weights for any S?

$$w_G(\delta S) = |S| \cdot |\overline{S}| \qquad\qquad w_H(\delta S) \approx \tfrac{d}{n} \cdot |S| \cdot |\overline{S}|$$

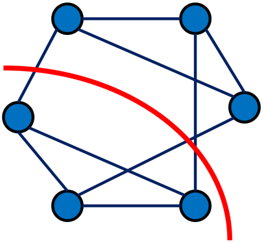
$$\forall S \subset V : \frac{w_G(\delta S)}{w_H(\delta S)} \approx \frac{n}{d}$$

Graph Sparsification: What is good sparse?

$G = K_n$



$H =$ d-**regular** (random)



What are the cut weights for any S?

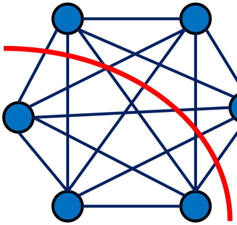
$w_G(\delta S) = |S| \cdot |\overline{S}|$ $\quad\quad$ $w_H(\delta S) \approx \frac{d}{n} \cdot |S| \cdot |\overline{S}|$

$$\forall S \subset V : \frac{w_G(\delta S)}{w_H(\delta S)} \approx \frac{n}{d}$$

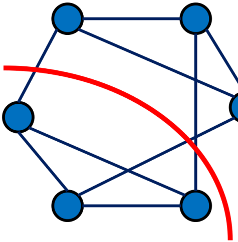
Could be large :(

Graph Sparsification: What is good sparse?

$G = K_n$



$H = d$-**regular** (random)



What are the cut weights for any S?

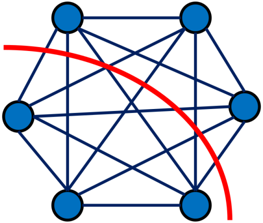
$w_G(\delta S) = |S| \cdot |\overline{S}|$ $\qquad$ $w_H(\delta S) \approx \frac{d}{n} \cdot |S| \cdot |\overline{S}|$

$$\forall S \subset V : \frac{w_G(\delta S)}{w_H(\delta S)} \approx \frac{n}{d}$$

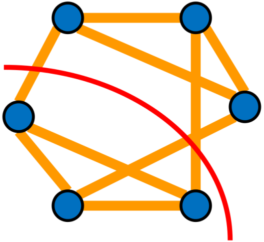
Could be large :($\quad$ What to do?

Graph Sparsification: What is good sparse?

$G = K_n$

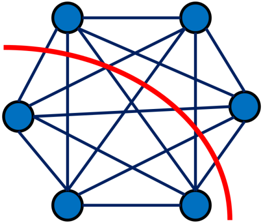


$H = d$-**regular** (random)

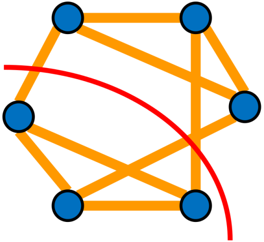


Graph Sparsification: What is good sparse?

$G = K_n$



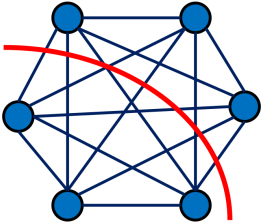
$H = d$-**regular** (random)



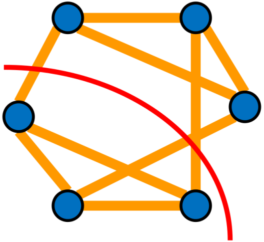
What are the cut weights for any S?

Graph Sparsification: What is **good** sparse?

$G = K_n$



$H =$ *d*-**regular** (random)

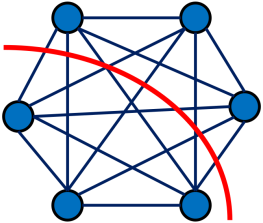


What are the cut weights for any S?

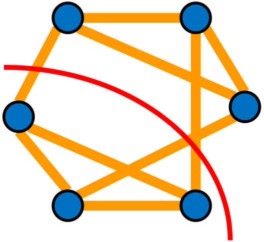
$$w_G(\delta S) = |S| \cdot |\overline{S}|$$

Graph Sparsification: What is good sparse?

$G = K_n$



$H = d$-**regular** (random)



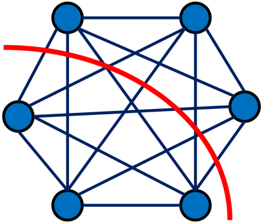
What are the cut weights for any S?

$$w_G(\delta S) = |S| \cdot |\overline{S}| \qquad w_H(\delta S) \approx \tfrac{d}{n} \cdot \tfrac{n}{d} \cdot |S| \cdot |\overline{S}|$$

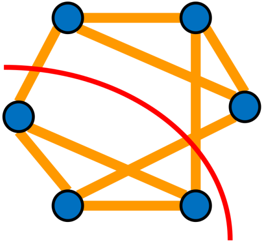
$$\forall S \subset V : \frac{w_G(\delta S)}{w_H(\delta S)} \approx 1$$

Graph Sparsification: What is **good** sparse?

$G = K_n$



$H = d$-**regular** (random)



What are the cut weights for any S?

$w_G(\delta S) = |S| \cdot |\overline{S}|$ $\qquad$ $w_H(\delta S) \approx \frac{d}{n} \cdot \frac{n}{d} \cdot |S| \cdot |\overline{S}|$

$$\forall S \subset V : \frac{w_G(\delta S)}{w_H(\delta S)} \approx 1$$

Benczúr & Karger: Can find such H quickly for any G!

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} =$

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \text{cut}_G(S)$

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \mathrm{cut}_G(S)$

$$(1-\varepsilon)\mathrm{cut}_H(S) \leq \mathrm{cut}_G(S) \leq (1+\varepsilon)\mathrm{cut}_H(S)$$

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \mathrm{cut}_G(S)$

$$(1-\varepsilon)\mathrm{cut}_H(S) \leq \mathrm{cut}_G(S) \leq (1+\varepsilon)\mathrm{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \text{cut}_G(S)$

$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

If we ask this only for $\mathbf{f} \in \{0,1\}^n \rightarrow (1+\varepsilon)$-cut similar combinatorial
Benczúr & Karger (1996)

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \text{cut}_G(S)$

$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

If we ask this only for $\mathbf{f} \in \{0,1\}^n \rightarrow (1+\varepsilon)$-cut similar combinatorial
Benczúr & Karger (1996)

If we ask this for all $\mathbf{f} \in \mathbb{R}^n \rightarrow$

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \mathrm{cut}_G(S)$

$$(1-\varepsilon)\mathrm{cut}_H(S) \leq \mathrm{cut}_G(S) \leq (1+\varepsilon)\mathrm{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

If we ask this only for $\mathbf{f} \in \{0,1\}^n \rightarrow (1+\varepsilon)$-cut similar combinatorial
Benczúr & Karger (1996)

If we ask this for all $\mathbf{f} \in \mathbb{R}^n \rightarrow (1+\varepsilon)$-**spectrally similar**
Spielman & Teng (2004)

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \text{cut}_G(S)$

$$(1-\varepsilon)\text{cut}_H(S) \leq \text{cut}_G(S) \leq (1+\varepsilon)\text{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

If we ask this only for $\mathbf{f} \in \{0,1\}^n \rightarrow (1+\varepsilon)$-cut similar combinatorial
Benczúr & Karger (1996)

If we ask this for all $\mathbf{f} \in \mathbb{R}^n \rightarrow (1+\varepsilon)$-**spectrally similar**
Spielman & Teng (2004)

Spectral sparsifiers are stronger!

Graph Sparsification: What is good sparse?

Recall if $\mathbf{f} \in \{0,1\}^n$ represents S then $\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} = \mathrm{cut}_G(S)$

$$(1-\varepsilon)\mathrm{cut}_H(S) \leq \mathrm{cut}_G(S) \leq (1+\varepsilon)\mathrm{cut}_H(S)$$

becomes

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f}$$

If we ask this only for $\mathbf{f} \in \{0,1\}^n \rightarrow (1+\varepsilon)$-cut similar combinatorial
Benczúr & Karger (1996)

If we ask this for all $\mathbf{f} \in \mathbb{R}^n \rightarrow (1+\varepsilon)$-**spectrally similar**
Spielman & Teng (2004)

Spectral sparsifiers are stronger!

but checking for spectral similarity is easier

Spectral Graph Sparsification

Rayleigh-Ritz gives:

Spectral Graph Sparsification

Rayleigh-Ritz gives:

$$\lambda_{\min} = \min \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \quad \text{and} \quad \lambda_{\max} = \max \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}}$$

Spectral Graph Sparsification

Rayleigh-Ritz gives:

$$\lambda_{\min} = \min \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \quad \text{and} \quad \lambda_{\max} = \max \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}}$$

What can we say about $\lambda_i(G)$ and $\lambda_i(H)$?

Spectral Graph Sparsification

Rayleigh-Ritz gives:

$$\lambda_{\min} = \min \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \quad \text{and} \quad \lambda_{\max} = \max \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}}$$

What can we say about $\lambda_i(G)$ and $\lambda_i(H)$?

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \le \mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \le (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f}$$

Spectral Graph Sparsification

Rayleigh-Ritz gives:

$$\lambda_{\min} = \min \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \quad \text{and} \quad \lambda_{\max} = \max \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}}$$

What can we say about $\lambda_i(G)$ and $\lambda_i(H)$?

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \leq \mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \leq (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f}$$

Eigenvalues are approximated well!

$$(1-\varepsilon)\lambda_i(G) \leq \lambda_i(H) \leq (1+\varepsilon)\lambda_i(G)$$

Using matrix ordering notation $(1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$

Spectral Graph Sparsification

Rayleigh-Ritz gives:

$$\lambda_{\min} = \min \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \quad \text{and} \quad \lambda_{\max} = \max \frac{\mathbf{x}^\mathsf{T}\mathbf{L}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}}$$

What can we say about $\lambda_i(G)$ and $\lambda_i(H)$?

$$(1-\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f} \le \mathbf{f}^\mathsf{T}\mathbf{L}_H\mathbf{f} \le (1+\varepsilon)\mathbf{f}^\mathsf{T}\mathbf{L}_G\mathbf{f}$$

Eigenvalues are approximated well!

$$(1-\varepsilon)\lambda_i(G) \le \lambda_i(H) \le (1+\varepsilon)\lambda_i(G)$$

Using matrix ordering notation $(1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$

As a consequence, $\arg\min_{\mathbf{x}} \|\mathbf{L}_H\mathbf{x} - \mathbf{b}\| \approx \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}\|$

Spectral Graph Sparsification

Let us consider unweighted graphs: $w_{ij} \in \{0, 1\}$

$$\mathbf{L}_G = \sum_{ij} w_{ij} \mathbf{L}_{ij} = \sum_{ij \in E} \mathbf{L}_{ij}$$

Spectral Graph Sparsification

Let us consider unweighted graphs: $w_{ij} \in \{0,1\}$

$$\mathbf{L}_G = \sum_{ij} w_{ij} \mathbf{L}_{ij} = \sum_{ij \in E} \mathbf{L}_{ij} = \sum_{ij \in E} (\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)(\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)^\mathsf{T}$$

Spectral Graph Sparsification

Let us consider unweighted graphs: $w_{ij} \in \{0,1\}$

$$\mathbf{L}_G = \sum_{ij} w_{ij}\mathbf{L}_{ij} = \sum_{ij\in E}\mathbf{L}_{ij} = \sum_{ij\in E}(\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)(\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)^\mathsf{T} = \sum_{e\in E}\mathbf{b}_e\mathbf{b}_e^\mathsf{T}$$

Spectral Graph Sparsification

Let us consider unweighted graphs: $w_{ij} \in \{0, 1\}$

$$\mathbf{L}_G = \sum_{ij} w_{ij}\mathbf{L}_{ij} = \sum_{ij \in E} \mathbf{L}_{ij} = \sum_{ij \in E} (\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)(\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)^\mathsf{T} = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T}$$

We look for a **subgraph** H

$$\mathbf{L}_H = \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T}$$

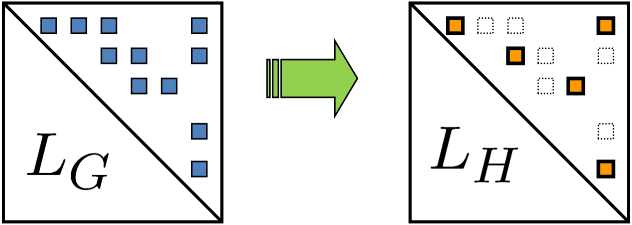
Spectral Graph Sparsification

Let us consider unweighted graphs: $w_{ij} \in \{0, 1\}$

$$\mathbf{L}_G = \sum_{ij} w_{ij}\mathbf{L}_{ij} = \sum_{ij \in E} \mathbf{L}_{ij} = \sum_{ij \in E} (\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)(\boldsymbol{\delta}_i - \boldsymbol{\delta}_j)^\mathsf{T} = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T}$$

We look for a **subgraph** H

$$\mathbf{L}_H = \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \quad \text{where } s_e \text{ is a new weight of edge e}$$



https://math.berkeley.edu/~nikhil/

Spectral Graph Sparsification

We want $\quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

Equivalent, given $\mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^{\mathsf{T}}$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

Equivalent, given $\mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T}$ find $\mathbf{s}$, s.t. $\mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$

Forget $\mathbf{L}$, given $\mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^\mathsf{T}$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$$

$$\text{Forget } \mathbf{L}, \text{ given } \mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{A} \preceq \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{A}$$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^{\mathsf{T}} \text{ find } \mathbf{s}\text{, s.t. } \mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^{\mathsf{T}} \preceq \kappa \cdot \mathbf{L}_G$$

$$\text{Forget } \mathbf{L}\text{, given } \mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^{\mathsf{T}} \text{ find } \mathbf{s}\text{, s.t. } \mathbf{A} \preceq \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^{\mathsf{T}} \preceq \kappa \cdot \mathbf{A}$$

$$\text{Same as, given } \mathbf{I} = \sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^{\mathsf{T}}$$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$$

$$\text{Forget } \mathbf{L}, \text{ given } \mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{A} \preceq \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{A}$$

$$\text{Same as, given } \mathbf{I} = \sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{I} \preceq \sum_{e \in E} s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{I}$$

Spectral Graph Sparsification

$$\text{We want}\quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e\in E} \mathbf{b}_e\mathbf{b}_e^\mathsf{T} \text{ find } \mathbf{s}\text{, s.t. } \mathbf{L}_G \preceq \sum_{e\in E} s_e\mathbf{b}_e\mathbf{b}_e^\mathsf{T} \preceq \kappa\cdot\mathbf{L}_G$$

$$\text{Forget } \mathbf{L}\text{, given } \mathbf{A} = \sum_{e\in E} \mathbf{a}_e\mathbf{a}_e^\mathsf{T} \text{ find } \mathbf{s}\text{, s.t. } \mathbf{A} \preceq \sum_{e\in E} s_e\mathbf{a}_e\mathbf{a}_e^\mathsf{T} \preceq \kappa\cdot\mathbf{A}$$

$$\text{Same as, given } \mathbf{I} = \sum_{e\in E} \mathbf{v}_e\mathbf{v}_e^\mathsf{T} \text{ find } \mathbf{s}\text{, s.t. } \mathbf{I} \preceq \sum_{e\in E} s_e\mathbf{v}_e\mathbf{v}_e^\mathsf{T} \preceq \kappa\cdot\mathbf{I}$$

How to get it?

Spectral Graph Sparsification

We want $(1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$

Equivalent, given $\mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T}$ find $\mathbf{s}$, s.t. $\mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$

Forget $\mathbf{L}$, given $\mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^\mathsf{T}$ find $\mathbf{s}$, s.t. $\mathbf{A} \preceq \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{A}$

Same as, given $\mathbf{I} = \sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$ find $\mathbf{s}$, s.t. $\mathbf{I} \preceq \sum_{e \in E} s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{I}$

How to get it? $\mathbf{v}_e \leftarrow \mathbf{A}^{-1/2}\mathbf{a}_e$

Spectral Graph Sparsification

$$\text{We want} \quad (1-\varepsilon)\mathbf{L}_G \preceq \mathbf{L}_H \preceq (1+\varepsilon)\mathbf{L}_G$$

$$\text{Equivalent, given } \mathbf{L}_G = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{L}_G \preceq \sum_{e \in E} s_e \mathbf{b}_e \mathbf{b}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{L}_G$$

$$\text{Forget } \mathbf{L}, \text{ given } \mathbf{A} = \sum_{e \in E} \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{A} \preceq \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{A}$$

$$\text{Same as, given } \mathbf{I} = \sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \text{ find } \mathbf{s}, \text{ s.t. } \mathbf{I} \preceq \sum_{e \in E} s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \preceq \kappa \cdot \mathbf{I}$$

How to get it? $\mathbf{v}_e \leftarrow \mathbf{A}^{-1/2}\mathbf{a}_e$

$$\text{Then } \textstyle\sum_{e \in E} s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \approx \mathbf{I} \iff \sum_{e \in E} s_e \mathbf{a}_e \mathbf{a}_e^\mathsf{T} \approx \mathbf{A}$$

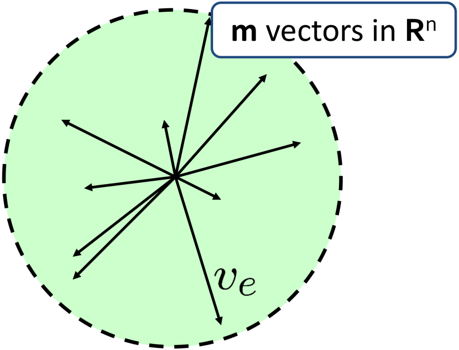
multiplying by $\mathbf{A}^{1/2}$ on both sides

Spectral Graph Sparsification: Intuition

How does $\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$ look like geometrically?

Spectral Graph Sparsification: Intuition

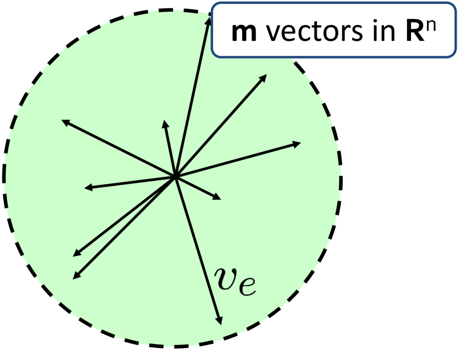
How does $\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$ look like geometrically?



Decomposition of identity: $\forall \mathbf{u}$ (unit vector): $\sum_{e \in E} (\mathbf{u}^\mathsf{T} \mathbf{v}_e)^2 = 1$

Spectral Graph Sparsification: Intuition

How does $\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$ look like geometrically?



Decomposition of identity: $\forall \mathbf{u}$ (unit vector): $\sum_{e \in E} (\mathbf{u}^\mathsf{T} \mathbf{v}_e)^2 = 1$

moment ellipse is a sphere

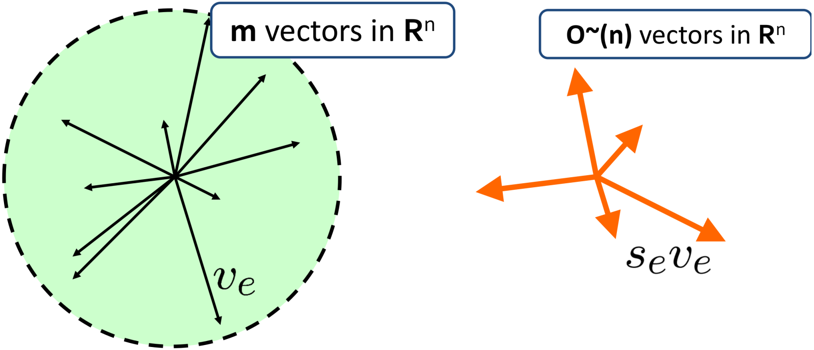
https://math.berkeley.edu/~nikhil/

Spectral Graph Sparsification: Intuition

What are we doing by choosing H?

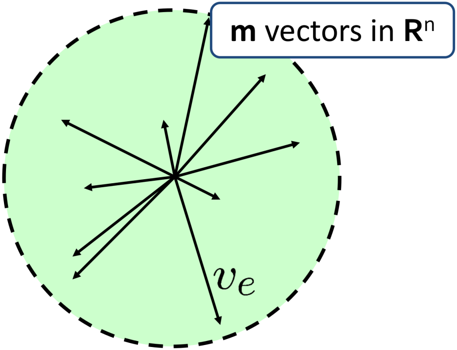
Spectral Graph Sparsification: Intuition

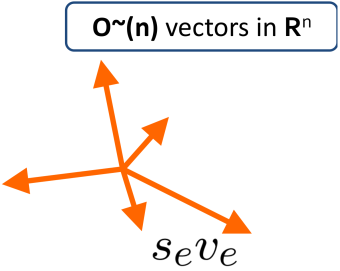
What are we doing by choosing H?



Spectral Graph Sparsification: Intuition

What are we doing by choosing H?





We take a subset of these $\mathbf{e}_e$s and scale them!

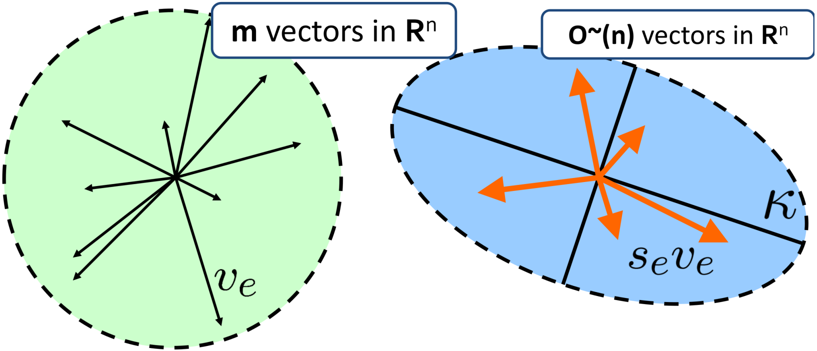
https://math.berkeley.edu/~nikhil/

Spectral Graph Sparsification: Intuition

What kind of scaling go we want?

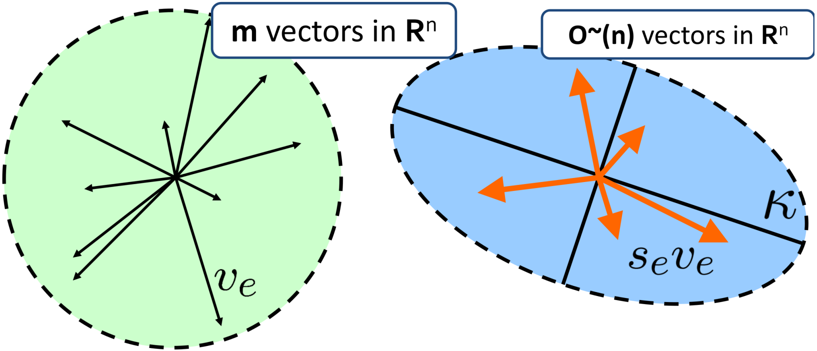
Spectral Graph Sparsification: Intuition

What kind of scaling go we want?



Spectral Graph Sparsification: Intuition

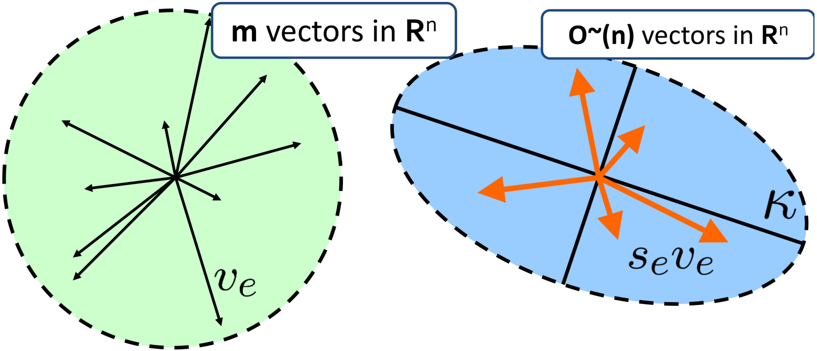
What kind of scaling go we want?



Such that the blue ellipsoid looks like identity!

Spectral Graph Sparsification: Intuition

What kind of scaling go we want?



Such that the blue ellipsoid looks like identity!

the blue eigenvalues are between 1 and κ

https://math.berkeley.edu/~nikhil/

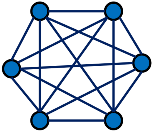
Spectral Graph Sparsification: Intuition

Example: What happens with K_n?

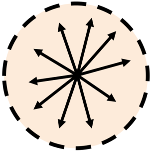
Spectral Graph Sparsification: Intuition

Example: What happens with K_n?

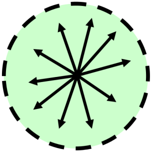
K_n graph



$\sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



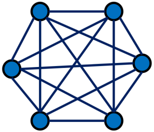
$\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$



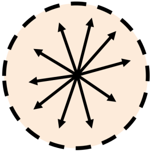
Spectral Graph Sparsification: Intuition

Example: What happens with K_n?

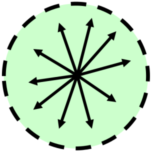
K_n graph



$\sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



$\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$

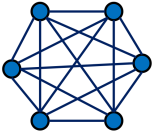


It is already isotropic! (looks like a sphere)

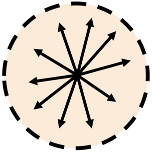
Spectral Graph Sparsification: Intuition

Example: What happens with K_n?

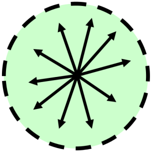
K_n graph



$\sum_{e\in E} \mathbf{b}_e\mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



$\sum_{e\in E} \mathbf{v}_e\mathbf{v}_e^\mathsf{T} = \mathbf{I}$



It is already isotropic! (looks like a sphere)

rescaling $\mathbf{v}_e = \mathbf{L}^{-1/2}\mathbf{b}_e$ does not change the shape

https://math.berkeley.edu/~nikhil/

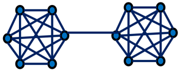
Spectral Graph Sparsification: Intuition

Example: What happens with a dumbbell?

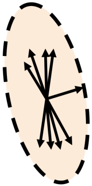
Spectral Graph Sparsification: Intuition

Example: What happens with a dumbbell?

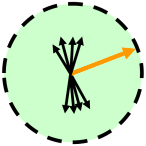
K_n graph



$\sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



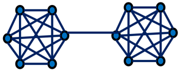
$\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$



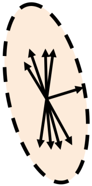
Spectral Graph Sparsification: Intuition

Example: What happens with a dumbbell?

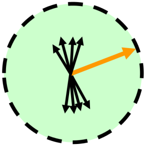
K_n graph



$\sum_{e\in E} \mathbf{b}_e\mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



$\sum_{e\in E} \mathbf{v}_e\mathbf{v}_e^\mathsf{T} = \mathbf{I}$

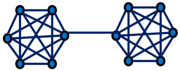


The vector corresponding to the link gets stretched!

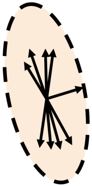
Spectral Graph Sparsification: Intuition

Example: What happens with a dumbbell?

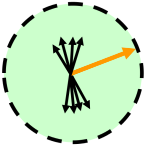
K_n graph



$\sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^\mathsf{T} = \mathbf{L}_G$



$\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^\mathsf{T} = \mathbf{I}$



The vector corresponding to the link gets stretched!

because this transformation makes all the directions important

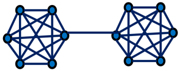
Spectral Graph Sparsification: Intuition

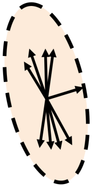
Example: What happens with a dumbbell?

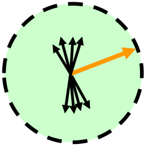
K_n graph

$\sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^{\mathsf{T}} = \mathbf{L}_G$

$\sum_{e \in E} \mathbf{v}_e \mathbf{v}_e^{\mathsf{T}} = \mathbf{I}$







The vector corresponding to the link gets stretched!

because this transformation makes all the directions important

rescaling reveals the vectors that are critical

https://math.berkeley.edu/~nikhil/

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2$$

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\left\|\mathbf{v}_e\right\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2$$

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^\mathsf{T}\mathbf{L}_G^{-1}\mathbf{b}_e$$

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^{\mathsf{T}}\mathbf{L}_G^{-1}\mathbf{b}_e = R_{\text{eff}}(e)$$

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^\mathsf{T}\mathbf{L}_G^{-1}\mathbf{b}_e = R_{\text{eff}}(e)$$

reminder $R_{\text{eff}}(e)$ is the potential difference between the nodes when injecting a unit current

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^{\mathsf{T}}\mathbf{L}_G^{-1}\mathbf{b}_e = R_{\text{eff}}(e)$$

reminder $R_{\text{eff}}(e)$ is the potential difference between the nodes when injecting a unit current

In other words: $R_{\text{eff}}(e)$ is related to the edge importance!

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\|\mathbf{v}_e\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^\mathsf{T}\mathbf{L}_G^{-1}\mathbf{b}_e = R_{\text{eff}}(e)$$

reminder $R_{\text{eff}}(e)$ is the potential difference between the nodes when injecting a unit current

In other words: $R_{\text{eff}}(e)$ is related to the edge importance!

Electrical intuition: We want to find an electrically similar H and the importance of the edge is its effective resistance $R_{\text{eff}}(e)$.

Spectral Graph Sparsification: Intuition

What it this rescaling $\mathbf{v}_e = \mathbf{L}_G^{-1/2}\mathbf{b}_e$ doing to the norm?

$$\left\|\mathbf{v}_e\right\|^2 = \left\|\mathbf{L}_G^{-1/2}\mathbf{b}_e\right\|^2 = \mathbf{b}_e^\mathsf{T}\mathbf{L}_G^{-1}\mathbf{b}_e = R_{\text{eff}}(e)$$

reminder $R_{\text{eff}}(e)$ is the potential difference between the nodes when injecting a unit current

In other words: $R_{\text{eff}}(e)$ is related to the edge importance!

Electrical intuition: We want to find an electrically similar H and the importance of the edge is its effective resistance $R_{\text{eff}}(e)$.

Edges with higher R_{eff} are more **electrically significant**!

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n / \varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n / \varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n/\varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Does this work?

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n/\varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Does this work?

Application of Matrix Chernoff Bound by Rudelson (1999)

$$1 - \varepsilon \prec \lambda \left(\sum_e s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \right) \prec 1 + \varepsilon$$

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n/\varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Does this work?

Application of Matrix Chernoff Bound by Rudelson (1999)

$$1 - \varepsilon \prec \lambda \left(\sum_e s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \right) \prec 1 + \varepsilon$$

finer bounds now available

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n / \varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Does this work?

Application of Matrix Chernoff Bound by Rudelson (1999)

$$1 - \varepsilon \prec \lambda \left(\sum_e s_e \mathbf{v}_e \mathbf{v}_e^\mathsf{T} \right) \prec 1 + \varepsilon$$

finer bounds now available

What is the the biggest problem here?

Spectral Graph Sparsification

Todo: Given $\mathbf{I} = \sum_e \mathbf{v}_e\mathbf{v}_e^\mathsf{T}$, find a sparse reweighting.

Randomized algorithm that finds $\mathbf{s}$:

- Sample $n \log n/\varepsilon^2$ with replacement $p_i \propto \|\mathbf{v}_e\|^2$ (resistances)
- Reweigh: $s_i = 1/p_i$ (to be unbiased)

Does this work?

Application of Matrix Chernoff Bound by Rudelson (1999)

$$1 - \varepsilon \prec \lambda\left(\sum_e s_e\mathbf{v}_e\mathbf{v}_e^\mathsf{T}\right) \prec 1 + \varepsilon$$

finer bounds now available

What is the the biggest problem here? Getting the p_is!

Spectral Graph Sparsification

We want to make this algorithm fast.

Spectral Graph Sparsification

We want to make this algorithm fast.

How can we compute the effective resistances?

Spectral Graph Sparsification

We want to make this algorithm fast.
How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^\mathsf{T}\widehat{\mathbf{x}}$

Spectral Graph Sparsification

We want to make this algorithm fast.
How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^{\mathsf{T}}\widehat{\mathbf{x}}$

Gaussian Elimination $\qquad \mathcal{O}(n^3)$

Spectral Graph Sparsification

We want to make this algorithm fast.

How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G \mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^{\mathsf{T}} \widehat{\mathbf{x}}$

Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$

Spectral Graph Sparsification

We want to make this algorithm fast.
How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^{\mathsf{T}}\widehat{\mathbf{x}}$

Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$
Spielman & Teng (2004)	$\mathcal{O}(m \log^{30} n)$

Spectral Graph Sparsification

We want to make this algorithm fast.
How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G \mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^{\mathsf{T}} \widehat{\mathbf{x}}$

Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$
Spielman & Teng (2004)	$\mathcal{O}(m \log^{30} n)$
Koutis, Miller, and Peng (2010)	$\mathcal{O}(m \log n)$

Spectral Graph Sparsification

We want to make this algorithm fast.

How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^{\mathsf{T}}\widehat{\mathbf{x}}$

Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$
Spielman & Teng (2004)	$\mathcal{O}(m \log^{30} n)$
Koutis, Miller, and Peng (2010)	$\mathcal{O}(m \log n)$

- Fast solvers for SDD systems:

Spectral Graph Sparsification

We want to make this algorithm fast.
How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^\mathsf{T}\widehat{\mathbf{x}}$

Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$
Spielman & Teng (2004)	$\mathcal{O}(m \log^{30} n)$
Koutis, Miller, and Peng (2010)	$\mathcal{O}(m \log n)$

- Fast solvers for SDD systems:
  - ↳ use sparsification internally

all the way until you hit the turtles

Spectral Graph Sparsification

We want to make this algorithm fast.

How can we compute the effective resistances?

Solve a linear system $\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{L}_G\mathbf{x} - \mathbf{b}_e\|$ and then $R_{\text{eff}} = \mathbf{b}_e^\mathsf{T}\widehat{\mathbf{x}}$

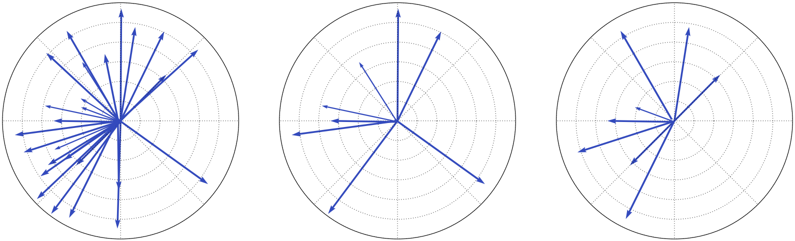
Gaussian Elimination	$\mathcal{O}(n^3)$
Fast Matrix Multiplication	$\mathcal{O}(n^{2.37})$
Spielman & Teng (2004)	$\mathcal{O}(m\log^{30} n)$
Koutis, Miller, and Peng (2010)	$\mathcal{O}(m\log n)$

- Fast solvers for SDD systems:
  - ↳ use sparsification internally

    all the way until you hit the turtles

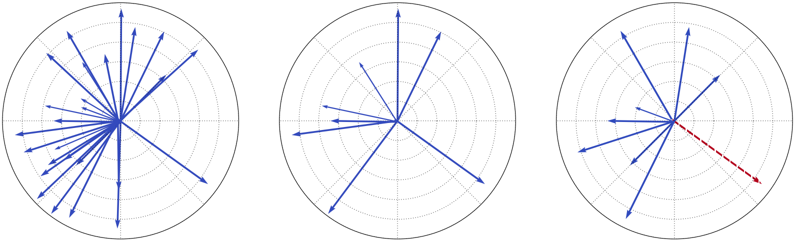
    still unfeasible when m is large

Efficient Sequential Learning
in Structured and Constrained Environments



Without losing information

Efficient Sequential Learning
in Structured and Constrained Environments

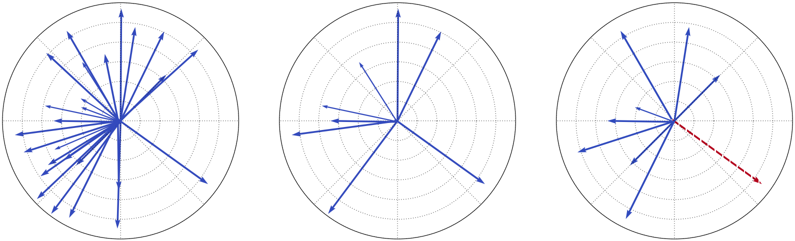


Without losing information

data-oblivious methods (e.g., uniform sampling)

↳ efficient but inaccurate [Bach, 2013]

Efficient Sequential Learning
in Structured and Constrained Environments



Without losing information

- data-oblivious methods (e.g., uniform sampling)
  ↳ efficient but inaccurate [Bach, 2013]
- data-adaptive methods (e.g. eigenvectors, leverage score sampling)
  ↳ accurate but too expensive [Alaoui and Mahoney, 2015]

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 1: find a small, provably accurate dictionary in near-linear time

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 1: find a small, provably accurate dictionary in near-linear time

Contribution: Two new single-pass sequential algorithms
KORS[Calandriello et al., 2017c]
SQUEAK[Calandriello et al., 2017a] (first part of the talk)

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 1: find a small, provably accurate dictionary in near-linear time

Contribution: Two new single-pass sequential algorithms

KORS[Calandriello et al., 2017c]

SQUEAK[Calandriello et al., 2017a] (first part of the talk)

variant of Nyström sampling

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 1: find a small, provably accurate dictionary in near-linear time

Contribution: Two new single-pass sequential algorithms
KORS[Calandriello et al., 2017c]
SQUEAK[Calandriello et al., 2017a] (first part of the talk)

variant of Nyström sampling

chooses samples using ridge leverage scores

↳ new ridge leverage score estimator

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 1: find a small, provably accurate dictionary in near-linear time

Contribution: Two new single-pass sequential algorithms

KORS[Calandriello et al., 2017c]

SQUEAK[Calandriello et al., 2017a] (first part of the talk)

variant of Nyström sampling

chooses samples using ridge leverage scores

↳ new ridge leverage score estimator

new sequential importance sampling approach

↳ analysis for non i.i.d. matrix sampling

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Efficient Sequential Learning

in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Contribution: two approximate second-order optimization algorithms
SKETCHED-KONS [Calandriello et al., 2017c]
PROS-N-KONS [Calandriello et al., 2017b] (second part of the talk)

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Contribution: two approximate second-order optimization algorithms
SKETCHED-KONS [Calandriello et al., 2017c]
PROS-N-KONS [Calandriello et al., 2017b] (second part of the talk)
approximate kernelized online Newton step

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Contribution: two approximate second-order optimization algorithms
SKETCHED-KONS [Calandriello et al., 2017c]
PROS-N-KONS [Calandriello et al., 2017b] (second part of the talk)

approximate kernelized online Newton step

constant per-step cost using Nyström embedding
↳ adaptive embedding based on KORS dictionary

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Contribution: two approximate second-order optimization algorithms
SKETCHED-KONS [Calandriello et al., 2017c]
PROS-N-KONS [Calandriello et al., 2017b] (second part of the talk)

- approximate kernelized online Newton step
- constant per-step cost using Nyström embedding
  ↳ adaptive embedding based on KORS dictionary
- preserve fast rates of exact online Newton step
  ↳ new adaptive restart strategy

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

not in this talk: provably accurate solutions in near-linear time

Kernel PCA [Musco and Musco, 2017]

Kernel Regression [Alaoui and Mahoney, 2015; Bach, 2013; Rudi et al., 2015]

Kernel K-Means [Musco and Musco, 2017]

Graph Semi-Supervised Learning [Calandriello et al., 2015]

Graph Sparsification [Calandriello et al., 2016]

Outline

(1) Dictionary learning

- ▷ Nyström sampling
- ▷ ridge leverage scores and effective dimension
- ▷ SQUEAK: sequential RLS importance sampling
  - ↳ analysis for non i.i.d. matrix sampling

(2) Online Kernel Learning

- ▷ online kernel learning and kernelized online Newton step
- ▷ PROS-N-KONS: adaptive Nyström embedding for online kernel learning
- ▷ adaptive restarts
- ▷ regression and classification experiments

Setting

Samples: $\mathbf{x}_i \in \mathcal{X}$ (e.g. $\mathbb{R}^d$)

Feature map: $\varphi(\mathbf{x}_i) : \mathcal{X} \to \mathcal{H} = \phi_i$

Dataset: $\mathcal{D}_n = \{\phi_i\}_{i=1}^n$, $\Phi_n = [\phi_1, \phi_2, \ldots, \phi_n]$

Empirical Kernel Matrix: $\Phi_n^\mathsf{T}\Phi_n = \mathbf{K}_n \in \mathbb{R}^{n \times n}$

Covariance operator: $\Phi_n\Phi_n^\mathsf{T} = \sum_{i=1}^n \phi_i\phi_i^\mathsf{T}$

Setting

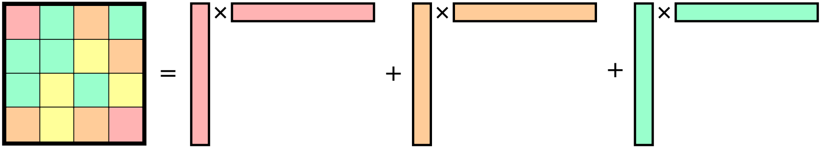
Samples: $\mathbf{x}_i \in \mathcal{X}$ (e.g. $\mathbb{R}^d$)

Feature map: $\varphi(\mathbf{x}_i) : \mathcal{X} \to \mathcal{H} = \phi_i$

Dataset: $\mathcal{D}_n = \{\phi_i\}_{i=1}^n$, $\Phi_n = [\phi_1, \phi_2, \ldots, \phi_n]$

Empirical Kernel Matrix: $\Phi_n^\mathsf{T}\Phi_n = \mathbf{K}_n \in \mathbb{R}^{n\times n}$

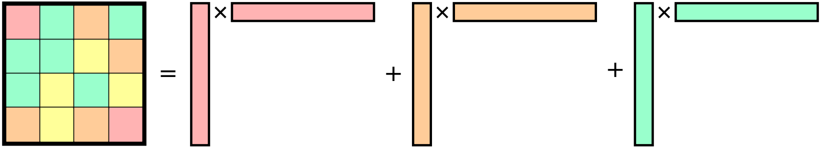
Covariance operator: $\Phi_n\Phi_n^\mathsf{T} = \sum_{i=1}^n \phi_i\phi_i^\mathsf{T}$



Dictionary Learning

What is Dictionary Learning (DL)?

Representation/Unsupervised learning:

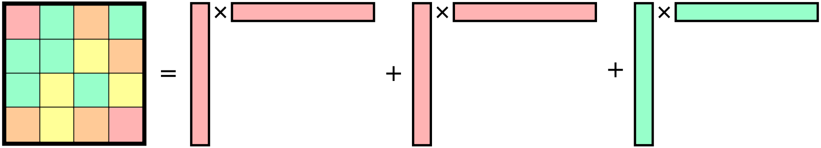


finding an accurate representation of the input data as a linear combination of a small set of basic elements (atoms)

Dictionary Learning

What is Dictionary Learning (DL)?

Representation/Unsupervised learning:

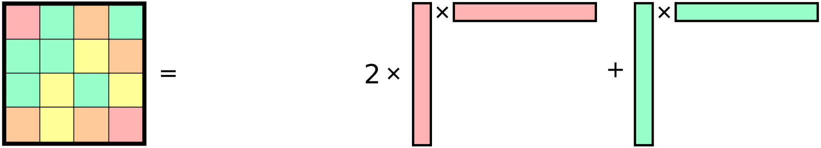


finding an accurate representation of the input data as a linear combination of a small set of basic elements (atoms)

Dictionary Learning

What is Dictionary Learning (DL)?

Representation/Unsupervised learning:

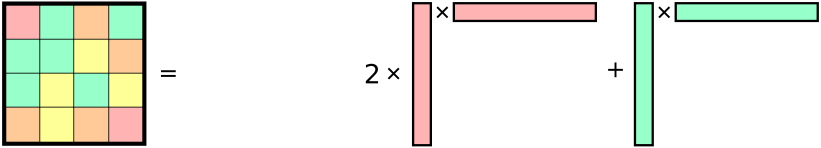


finding an accurate representation of the input data as a linear combination of a small set of basic elements (atoms)

Dictionary Learning

What is Dictionary Learning (DL)?

Representation/Unsupervised learning:



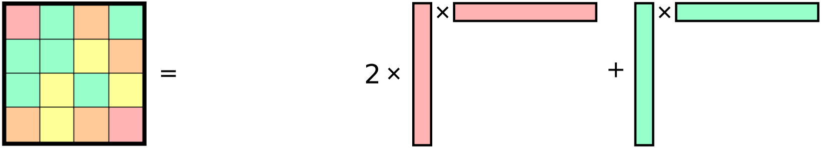
finding an accurate representation of the input data as a linear combination of a small set of basic elements (atoms)

Dictionary $\mathcal{I} = \{(w_j, \phi_j)\}_{j=1}^{m}$

Dictionary Learning

What is Dictionary Learning (DL)?

Representation/Unsupervised learning:



finding an accurate representation of the input data as a linear combination of a small set of basic elements (atoms)

Dictionary $\mathcal{I} = \{(w_j, \phi_j)\}_{j=1}^{m}$

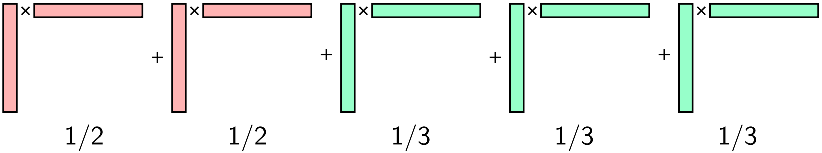
$$\sum_{i=1}^{m} w_i \phi_i \phi_i^{\mathsf{T}} = \sum_{i=1}^{m} (\sqrt{w_i}\phi_i)(\sqrt{w_i}\phi_i)^{\mathsf{T}} = \Phi_n \mathbf{S}_n \mathbf{S}_n^{\mathsf{T}} \Phi_n^{\mathsf{T}}$$

Dictionary Learning

(1) which to pick?   (2) how many to pick?   (3) how to build $\mathcal{I}$?

Dictionary Learning

(1) which to pick? (2) how many to pick? (3) how to build $\mathcal{I}$?

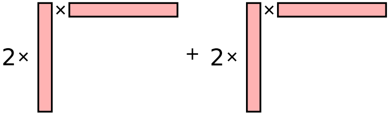


Dictionary Learning

(1) which to pick? (2) how many to pick? (3) how to build $\mathcal{I}$?

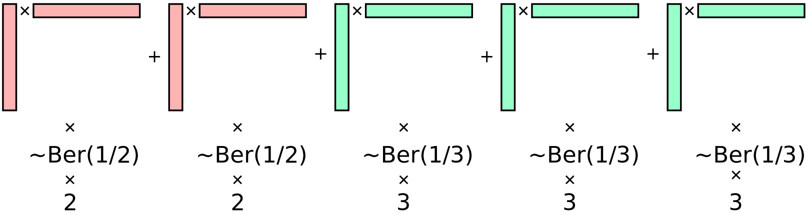
Dictionary Learning

(1) which to pick? (2) how many to pick? (3) how to build $\mathcal{I}$?



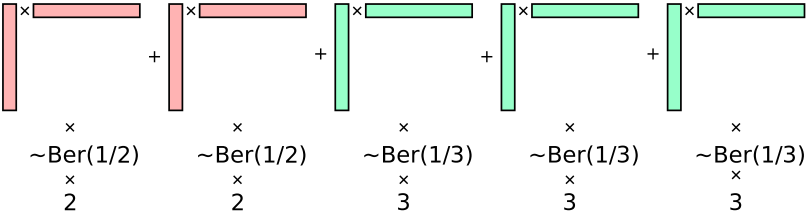
Dictionary Learning

(1) which to pick? (2) how many to pick? (3) how to build $\mathcal{I}$?



Dictionary Learning

(1) which to pick? (2) how many to pick? (3) how to build $\mathcal{I}$?



Nyström sampling: unbiased estimator

$$\Phi_n \mathbf{S}_n \mathbf{S}_n^{\mathsf{T}} \Phi_n^{\mathsf{T}} = \sum_{i=1}^{n} \sum_{j=1}^{\overline{q}} \frac{1}{p_i} \frac{z_{i,j}}{\overline{q}} \phi_i \phi_i^{\mathsf{T}}$$

Ridge Leverage Scores

Intuitively, RLS capture orthogonality

$$\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_n^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} = \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i$$

Ridge Leverage Scores

Intuitively, RLS capture orthogonality

$$\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_n^\mathsf{T}(\mathbf{K}_n + \gamma \mathbf{I}_n)^{-1}\mathbf{e}_{n,i} = \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma \mathbf{I})^{-1}\phi_i$$

If all ϕ_i are orthogonal, we have

$$\tau_{n,i} = \phi_i^\mathsf{T}(\phi_i\phi_i^\mathsf{T} + \gamma \mathbf{I})^{-1}\phi_i = \frac{\phi_i^\mathsf{T}\phi_i}{\phi_i^\mathsf{T}\phi_i + \gamma} \sim \mathbf{1}$$

Ridge Leverage Scores

Intuitively, RLS capture orthogonality

$$\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_n^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} = \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i$$

If all ϕ_i are orthogonal, we have

$$\tau_{n,i} = \phi_i^\mathsf{T}(\phi_i\phi_i^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i = \frac{\phi_i^\mathsf{T}\phi_i}{\phi_i^\mathsf{T}\phi_i + \gamma} \sim \mathbf{1}$$

If all ϕ_i are identical (collinear), we have

$$\tau_{n,i} = \phi_i^\mathsf{T}(n\phi_i\phi_i^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i = \frac{\phi_i^\mathsf{T}\phi_i}{n\phi_i^\mathsf{T}\phi_i + \gamma} \sim \frac{1}{n}$$

Ridge Leverage Scores

Intuitively, RLS capture orthogonality

$$\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_n^{\mathsf{T}}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} = \phi_i^{\mathsf{T}}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\phi_i$$

If all ϕ_i are orthogonal, we have

$$\tau_{n,i} = \phi_i^{\mathsf{T}}(\phi_i\phi_i^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\phi_i = \frac{\phi_i^{\mathsf{T}}\phi_i}{\phi_i^{\mathsf{T}}\phi_i + \gamma} \sim \mathbf{1}$$

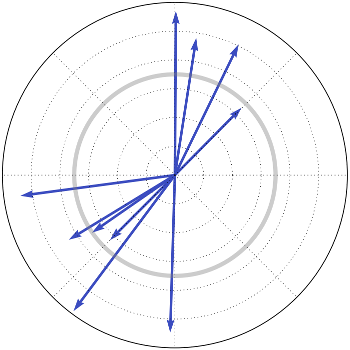
If all ϕ_i are identical (collinear), we have

$$\tau_{n,i} = \phi_i^{\mathsf{T}}(n\phi_i\phi_i^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\phi_i = \frac{\phi_i^{\mathsf{T}}\phi_i}{n\phi_i^{\mathsf{T}}\phi_i + \gamma} \sim \frac{1}{n}$$

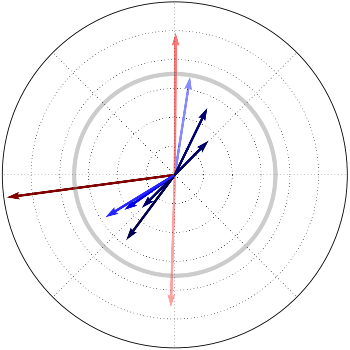
Given Φ_{t-1}, adding a new column to it can only reduce the RLS of columns already in Φ_{t-1}

$$\boldsymbol{\tau_{t,i} \leq \tau_{t-1,i}}$$

Ridge Leverage Scores

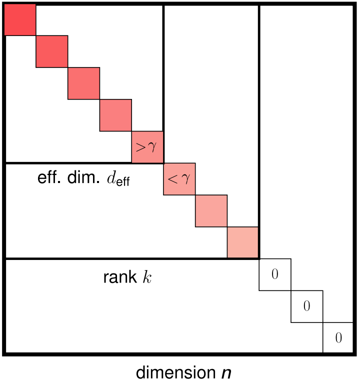


Ridge Leverage Scores



Effective Dimension

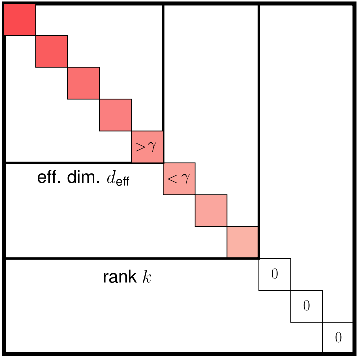
Intuitively, the effective dimension is the number of relevant directions in the data



$$d_{\text{eff}}^{n}(\gamma) = \sum_{i=1}^{n} \tau_{n,i} = \operatorname{Tr}\left(\mathbf{K}_n(\mathbf{K}_n + \gamma \mathbf{I}_n)^{-1}\right) = \sum_{i=1}^{n} \frac{\lambda_i(\mathbf{K}_n)}{\lambda_i(\mathbf{K}_n) + \gamma} \leq \operatorname{Rank}(\mathbf{K}_n)$$

Effective Dimension

Intuitively, the effective dimension is the number of relevant directions in the data



Given $d_{\text{eff}}^{t-1}(\gamma)$, adding a new column to Φ_{t-1} can only increase $d_{\text{eff}}^{t}(\gamma)$

$$\mathbf{d}_{\text{eff}}^{\mathbf{t}}(\gamma) \geq \mathbf{d}_{\text{eff}}^{\mathbf{t-1}}(\gamma)$$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\Phi \mathbf{S}\mathbf{S}^{\mathsf{T}} \Phi^{\mathsf{T}}$$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^\mathsf{T}}^{\text{multiplicative error}} \preceq \Phi \mathbf{S}\mathbf{S}^\mathsf{T}\Phi^\mathsf{T} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^\mathsf{T}}^{\text{multiplicative error}}$$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Low-rank PSD matrix approximation

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Low-rank PSD matrix approximation

Projection $\mathbf{\Pi}_{\mathcal{I}} = \Phi\mathbf{S}(\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}\Phi\mathbf{S})\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}$ on dictionary span

↳ Nyström approx. $\widetilde{\mathbf{K}} = \Phi^{\mathsf{T}}\mathbf{\Pi}_{\mathcal{I}}\Phi$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Low-rank PSD matrix approximation

Projection $\mathbf{\Pi}_{\mathcal{I}} = \Phi\mathbf{S}(\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}\Phi\mathbf{S})\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}$ on dictionary span

↳ Nyström approx. $\widetilde{\mathbf{K}} = \Phi^{\mathsf{T}}\mathbf{\Pi}_{\mathcal{I}}\Phi$

↳ $\mathbf{K} - \frac{\varepsilon}{1-\varepsilon}\gamma\mathbf{I}_n \preceq \widetilde{\mathbf{K}} \preceq \mathbf{K}$

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^\mathsf{T}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^\mathsf{T}\Phi^\mathsf{T} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^\mathsf{T}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Low-rank PSD matrix approximation

Projection $\mathbf{\Pi}_\mathcal{I} = \Phi\mathbf{S}(\mathbf{S}^\mathsf{T}\Phi^\mathsf{T}\Phi\mathbf{S})\mathbf{S}^\mathsf{T}\Phi^\mathsf{T}$ on dictionary span

↳ Nyström approx. $\widetilde{\mathbf{K}} = \Phi^\mathsf{T}\mathbf{\Pi}_\mathcal{I}\Phi$

↳ $\mathbf{K} - \frac{\varepsilon}{1-\varepsilon}\gamma\mathbf{I}_n \preceq \widetilde{\mathbf{K}} \preceq \mathbf{K}$

Graph sparsification (not in this talk)

Reconstruction guarantees

An (ε, γ)-accurate dictionary $\mathcal{I}$ satisfies

$$\overbrace{(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} - \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq \overbrace{(1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}}}^{\text{multiplicative error}} + \overbrace{\varepsilon\gamma\mathbf{I}}^{\text{additive error}}$$

Low-rank PSD matrix approximation

Projection $\mathbf{\Pi}_{\mathcal{I}} = \Phi\mathbf{S}(\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}\Phi\mathbf{S})\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}}$ on dictionary span

↳ Nyström approx. $\widetilde{\mathbf{K}} = \Phi^{\mathsf{T}}\mathbf{\Pi}_{\mathcal{I}}\Phi$

↳ $\mathbf{K} - \frac{\varepsilon}{1-\varepsilon}\gamma\mathbf{I}_n \preceq \widetilde{\mathbf{K}} \preceq \mathbf{K}$

Graph sparsification (not in this talk)

In graph problems dictionary $\mathcal{I}$ is subset of reweighted edges

↳ $(1-\varepsilon)\mathbf{L}_{\mathcal{G}} \preceq \mathbf{L}_{\mathcal{I}} \preceq (1+\varepsilon)\mathbf{L}_{\mathcal{G}}$

Oracle RLS Sampling

Theorem (Alaoui and Mahoney, 2015)

Given γ be the Nystrom regularization, ε the accuracy, δ the confidence. If the dictionary $\mathcal{I}_n$ is computed using the sampling distribution $p_{n,i} \propto \tau_{n,i}$ and using at least m columns

$$m \geq \left(\frac{2 d_{eff}^n(\gamma)}{\varepsilon^2} \right) \log \left(\frac{n}{\delta} \right),$$

then with probability $1 - \delta$ we have

$$(1-\varepsilon)\Phi_n \Phi_n^\mathsf{T} - \varepsilon\gamma\mathbf{I} \preceq \Phi \mathbf{S}\mathbf{S}^\mathsf{T} \Phi^\mathsf{T} \preceq (1+\varepsilon)\Phi_n \Phi_n^\mathsf{T} + \varepsilon\gamma\mathbf{I}$$

Oracle RLS Sampling

Theorem (Alaoui and Mahoney, 2015)

Given γ *be the Nystrom regularization,* ε *the accuracy,* δ *the confidence.*
If the dictionary $\mathcal{I}_n$ *is computed using the sampling distribution* $p_{n,i} \propto \tau_{n,i}$ *and using at least m columns*

$$m \geq \left(\frac{2 d_{eff}^{n}(\gamma)}{\varepsilon^{2}} \right) \log \left(\frac{n}{\delta} \right),$$

then with probability $1 - \delta$ *we have*

$$(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}} - \varepsilon\gamma\mathbf{I} \preceq \Phi\mathbf{S}\mathbf{S}^{\mathsf{T}}\Phi^{\mathsf{T}} \preceq (1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}} + \varepsilon\gamma\mathbf{I}$$

~~Goal 1: small and accurate dictionary~~ done!

Oracle RLS Sampling

Theorem (Alaoui and Mahoney, 2015)

Given γ be the Nystrom regularization, ε the accuracy, δ the confidence. If the dictionary $\mathcal{I}_n$ is computed using the sampling distribution $p_{n,i} \propto \tau_{n,i}$ and using at least m columns

$$m \geq \left(\frac{2 d_{eff}^n(\gamma)}{\varepsilon^2} \right) \log \left(\frac{n}{\delta} \right),$$

then with probability $1 - \delta$ we have

$$(1-\varepsilon)\Phi_n\Phi_n^\mathsf{T} - \varepsilon\gamma\mathbf{I} \preceq \Phi\mathbf{S}\mathbf{S}^\mathsf{T}\Phi^\mathsf{T} \preceq (1+\varepsilon)\Phi_n\Phi_n^\mathsf{T} + \varepsilon\gamma\mathbf{I}$$

~~Goal 1: small and accurate dictionary~~ done!

Goal 1: small and accurate dictionary in near-linear time

If someone gave us the RLS

Oracle RLS Sampling

Theorem (Alaoui and Mahoney, 2015)

Given γ be the Nystrom regularization, ε the accuracy, δ the confidence. If the dictionary $\mathcal{I}_n$ is computed using the sampling distribution $p_{n,i} \propto \tau_{n,i}$ and using at least m columns

$$m \geq \left(\frac{2d_{eff}^n(\gamma)}{\varepsilon^2} \right) \log \left(\frac{n}{\delta} \right),$$

then with probability $1 - \delta$ we have

$$(1-\varepsilon)\Phi_n\Phi_n^\mathsf{T} - \varepsilon\gamma\mathbf{I} \preceq \Phi\mathbf{S}\mathbf{S}^\mathsf{T}\Phi^\mathsf{T} \preceq (1+\varepsilon)\Phi_n\Phi_n^\mathsf{T} + \varepsilon\gamma\mathbf{I}$$

~~Goal 1: small and accurate dictionary~~ done!

Goal 1: small and accurate dictionary in near-linear time

If someone gave us the RLS

Computing $\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_n^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i}$ also requires storing and inverting the full $\mathbf{K}_n$

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i}$ $\Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$\tau_{n,i} = \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i}$$

- $\widetilde{\tau}_{n,i} = \mathbf{e}_i^\mathsf{T}\widetilde{\mathbf{K}}_\mathbf{n}(\widetilde{\mathbf{K}}_\mathbf{t} + \gamma\mathbf{I})^{-1}\mathbf{e}_i$

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$\begin{aligned}\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i}\\ &= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i,\end{aligned}$$

- $\widetilde{\tau}_{n,i} = \mathbf{e}_i^\mathsf{T}\widetilde{\mathbf{K}}_\mathbf{n}(\widetilde{\mathbf{K}}_\mathbf{t} + \gamma\mathbf{I})^{-1}\mathbf{e}_i$
- Instead, approximate $\tau_{n,i}$ directly in $\mathcal{H}$

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$
\begin{aligned}
\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} \\
&= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i, \\
\widetilde{\tau}_{n,i} &= \phi_i^\mathsf{T}(\Phi_n\mathbf{S}_n\mathbf{S}_n^\mathsf{T}\Phi^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i
\end{aligned}
$$

- $\widetilde{\tau}_{n,i} = \mathbf{e}_i^\mathsf{T}\widetilde{\mathbf{K}}_\mathbf{n}(\widetilde{\mathbf{K}}_\mathbf{t} + \gamma\mathbf{I})^{-1}\mathbf{e}_i$
- Instead, approximate $\tau_{n,i}$ directly in $\mathcal{H}$

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$\begin{aligned}\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i}\\ &= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i,\\ \widetilde{\tau}_{n,i} &= \phi_i^\mathsf{T}(\Phi_n\mathbf{S}_n\mathbf{S}_n^\mathsf{T}\Phi^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i\\ &= \frac{1+\varepsilon}{\alpha\gamma}\left(k_{i,i} - \mathbf{k}_{n,i}\mathbf{S}_n\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}\mathbf{S}_n^\mathsf{T}\mathbf{k}_{n,i}\right).\end{aligned}$$

- $\widetilde{\tau}_{n,i} = \mathbf{e}_i^\mathsf{T}\widetilde{\mathbf{K}}_\mathbf{n}(\widetilde{\mathbf{K}}_\mathbf{t} + \gamma\mathbf{I})^{-1}\mathbf{e}_i$
- Instead, approximate $\tau_{n,i}$ directly in $\mathcal{H}$, and then use kernel trick

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$\begin{aligned}
\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} \\
&= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i, \\
\widetilde{\tau}_{n,i} &= \phi_i^\mathsf{T}(\Phi_n\mathbf{S}_n\mathbf{S}_n^\mathsf{T}\Phi^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i \\
&= \frac{1+\varepsilon}{\alpha\gamma}\left(k_{i,i} - \mathbf{k}_{n,i}\mathbf{S}_n\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}\mathbf{S}_n^\mathsf{T}\mathbf{k}_{n,i}\right).
\end{aligned}$$

- $\widetilde{\tau}_{n,i} = \mathbf{e}_i^\mathsf{T}\widetilde{\mathbf{K}}_\mathbf{n}(\widetilde{\mathbf{K}}_\mathbf{t} + \gamma\mathbf{I})^{-1}\mathbf{e}_i$
- Instead, approximate $\tau_{n,i}$ directly in $\mathcal{H}$, and then use kernel trick
- If $\mathcal{I}$ (ε,γ)-accurate $\Rightarrow \tau_{n,i}(\gamma)/\left(\frac{1+3\varepsilon}{1-\varepsilon}\right) \leq \widetilde{\tau}_{n,i} \leq \tau_{n,i}(\gamma)$
  [Calandriello et al., 2017a]

Estimating RLS

Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

$$\begin{aligned}\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i} \\ &= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i, \\ \widetilde{\tau}_{n,i} &= \phi_i^\mathsf{T}(\Phi_n\mathbf{S}_n\mathbf{S}_n^\mathsf{T}\Phi^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i \\ &= \frac{1+\varepsilon}{\alpha\gamma}\left(k_{i,i} - \mathbf{k}_{n,i}\mathbf{S}_n\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}\mathbf{S}_n^\mathsf{T}\mathbf{k}_{n,i}\right).\end{aligned}$$

- $\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}$ is a $J \times J$ matrix
  ↳ $\widetilde{\tau}_{n,i}$ can be computed in $\mathcal{O}(J^2)$ space and $\mathcal{O}(J^3)$ time

Estimating RLS

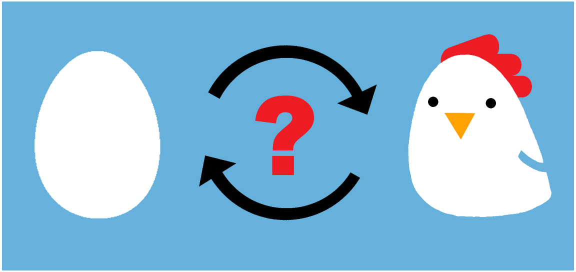
Good news 1: given accurate $\widetilde{\tau}_{n,i} \Rightarrow$ compute accurate dictionary
Good news 2: given accurate dictionary $\Rightarrow$ compute accurate $\widetilde{\tau}_{n,i}$

Given dictionary $\mathcal{I}_n$ with $|\mathcal{I}_n| = J$ atoms

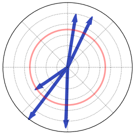
$$\begin{aligned}\tau_{n,i} &= \mathbf{e}_{n,i}\mathbf{K}_t^\mathsf{T}(\mathbf{K}_n + \gamma\mathbf{I}_n)^{-1}\mathbf{e}_{n,i}\\ &= \phi_i^\mathsf{T}(\Phi_n\Phi_n^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i,\\ \widetilde{\tau}_{n,i} &= \phi_i^\mathsf{T}(\Phi_n\mathbf{S}_n\mathbf{S}_n^\mathsf{T}\Phi^\mathsf{T} + \gamma\mathbf{I})^{-1}\phi_i\\ &= \frac{1+\varepsilon}{\alpha\gamma}\left(k_{i,i} - \mathbf{k}_{n,i}\mathbf{S}_n\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}\mathbf{S}_n^\mathsf{T}\mathbf{k}_{n,i}\right).\end{aligned}$$

- $\left(\mathbf{S}_n^\mathsf{T}\mathbf{K}_t\mathbf{S}_n + \gamma\mathbf{I}\right)^{-1}$ is a $J \times J$ matrix
  ↳ $\widetilde{\tau}_{n,i}$ can be computed in $\mathcal{O}(J^2)$ space and $\mathcal{O}(J^3)$ time
- $\widetilde{\tau}_{n,i}$ for $i \in \mathcal{I}_n$ can be computed using only samples contained in $\mathcal{I}_n$.

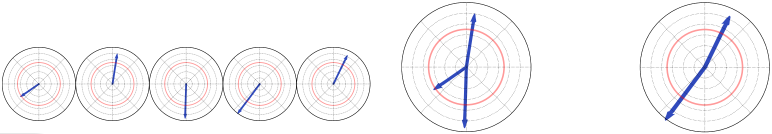
Chicken and egg problem



SQUEAK- Sequential RLS sampling

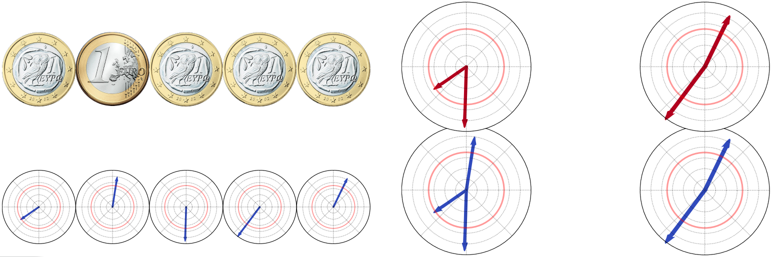


SQUEAK- Sequential RLS sampling



SQUEAK- Sequential RLS sampling

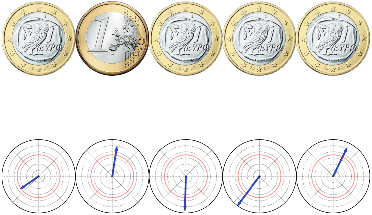
$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i}$,
$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

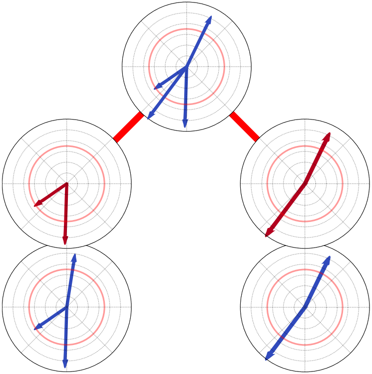


SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i},$

$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$





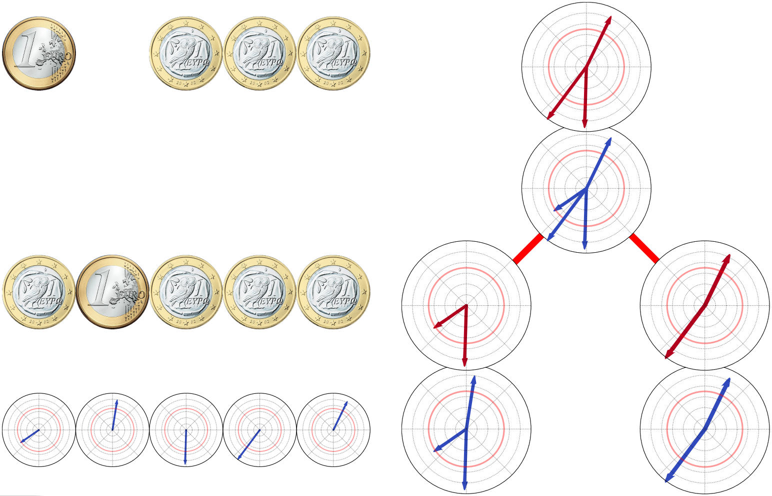
SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i}$,

$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$

$z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

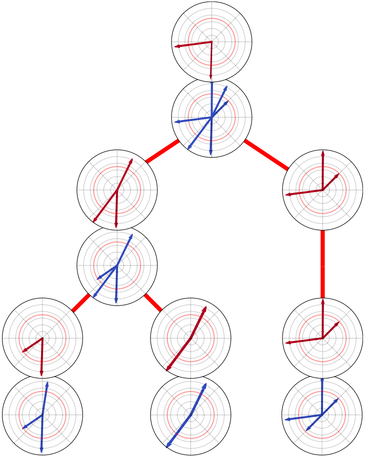


SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i}$, $\quad z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$ $\quad z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i}$ $\quad z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{3,i}}\right)\right\} z_{2,i}$



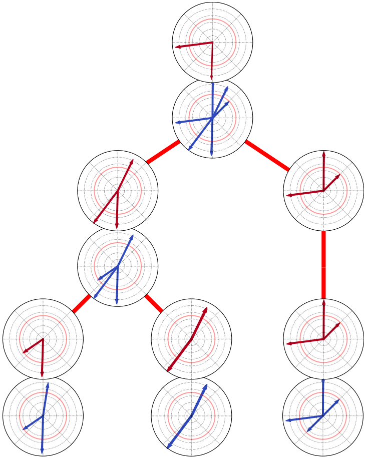
SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i}$, $z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$, $z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i}$, $z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{3,i}}\right)\right\} z_{2,i}$

- Store points directly in $\mathcal{I}$
  - ↳ single pass over the dataset

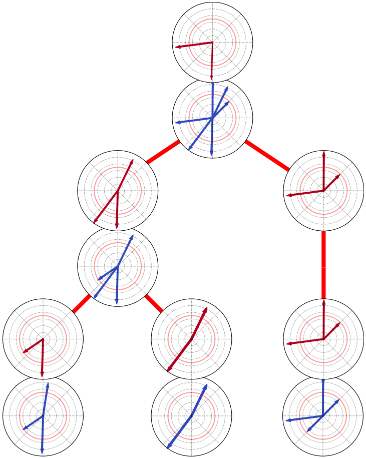


SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i},$
$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$
$z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i}$
$z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{3,i}}\right)\right\} z_{2,i}$



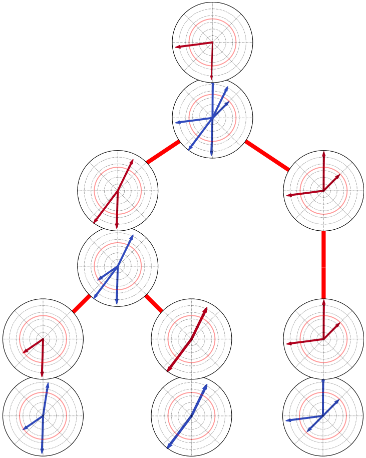
- Store points directly in $\mathcal{I}$
  - ↳ single pass over the dataset
- Unnormalized $\widetilde{p}_{t,i}$
  - ↳ no need for approximate $d_{\text{eff}}(\gamma)_t$

SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i},$
$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$
$z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i}$
$z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{3,i}}\right)\right\} z_{2,i}$



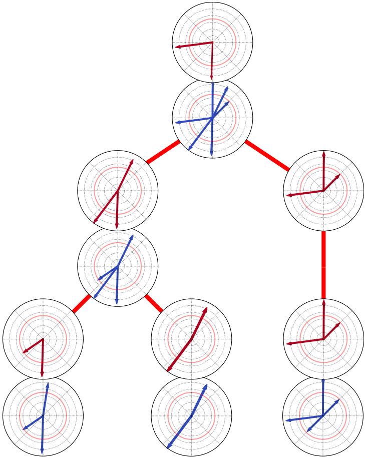
- Store points directly in $\mathcal{I}$
  - ↳ single pass over the dataset
- Unnormalized $\widetilde{p}_{t,i}$
  - ↳ no need for approximate $d_{\text{eff}}(\gamma)_t$
- Never recompute $\widetilde{\tau}_{t,i}$ after dropping
  - ↳ never construct the whole $\mathbf{K}_n$

SQUEAK- Sequential RLS sampling

$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i}$,
$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$

$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i}$
$z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$

$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i}$
$z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{3,i}}\right)\right\} z_{2,i}$



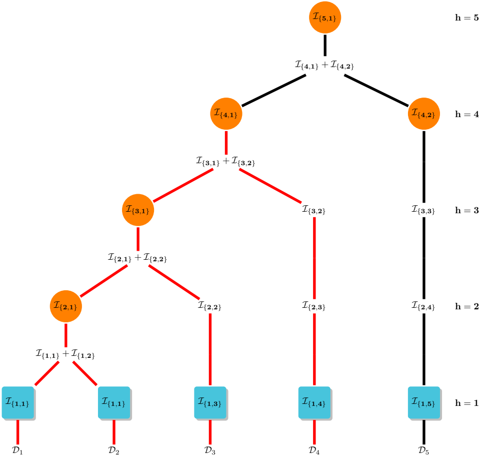
- Store points directly in $\mathcal{I}$
  - ↳ single pass over the dataset
- Unnormalized $\widetilde{p}_{t,i}$
  - ↳ no need for approximate $d_{\text{eff}}(\gamma)_t$
- Never recompute $\widetilde{\tau}_{t,i}$ after dropping
  - ↳ never construct the whole $\mathbf{K}_n$
- Runtime depends on merge tree

SQUEAK- Sequential RLS sampling

$\mathcal{I}$ with $|\mathcal{I}| = J$ atoms, space: $\mathcal{O}(J^2)$, Runtime: single merge $\mathcal{O}(J^3)$

SQUEAK- Sequential RLS sampling

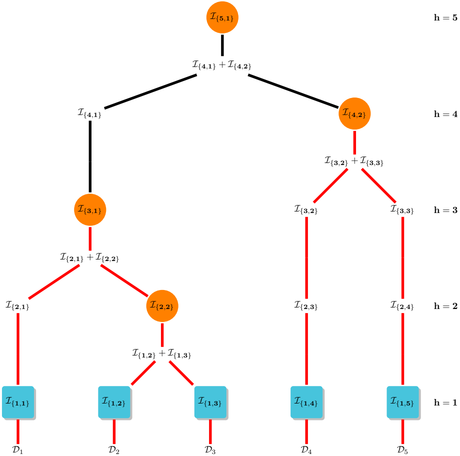
SQUEAK - fully unbalanced tree: $\widetilde{\mathcal{O}}(nJ^3)$



$\mathcal{I}$ with $|\mathcal{I}| = J$ atoms, space: $\mathcal{O}(J^2)$, Runtime: single merge $\mathcal{O}(J^3)$

DISQUEAK- Distributed sequential RLS sampling

DISQUEAK - fully balanced tree: $\widetilde{\mathcal{O}}(\log(n)J^3)$



$\mathcal{I}$ with $|\mathcal{I}| = J$ atoms, space: $\mathcal{O}(J^2)$, Runtime: single merge $\mathcal{O}(J^3)$

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = \left(\frac{1+2\varepsilon}{1-2\varepsilon}\right)$ *and* $\gamma > 1$. *For any* $0 \leq \varepsilon \leq 1$, *and* $0 \leq \delta \leq 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \geq \frac{\mathbf{26}\alpha}{\varepsilon^2} \log(\frac{\mathbf{n}}{\delta})$, *then w.p.* $1-\delta$, *for all nodes* $\{h, l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε, γ)*-accurate*.

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \leq \mathcal{O}(\overline{q} d_{eff}(\gamma)_{\{h,l\}}) \leq \mathcal{O}(\frac{\alpha}{\varepsilon^2} d_{eff}^n(\gamma) \log(\frac{n}{\delta}))$.

- Accuracy/dictionary size match oracle RLS-sampling at any time
  ↳ no free lunch: space/time scale with $|\mathcal{I}| \leq d_{\text{eff}}^n(\gamma)$

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = \left(\frac{1+2\varepsilon}{1-2\varepsilon}\right)$ *and* $\gamma > 1$. *For any* $0 \leq \varepsilon \leq 1$, *and* $0 \leq \delta \leq 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \geq \frac{\mathbf{26}\alpha}{\varepsilon^{\mathbf{2}}} \log(\frac{\mathbf{n}}{\delta}))$, *then w.p.* $1 - \delta$, *for all nodes* $\{h, l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε, γ)*-accurate.*

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \leq \mathcal{O}(\overline{q} d_{eff}(\gamma)_{\{h,l\}}) \leq \mathcal{O}(\frac{\alpha}{\varepsilon^2} d_{eff}^n(\gamma) \log(\frac{n}{\delta}))).$

- Accuracy/dictionary size match oracle RLS-sampling at any time
  ↳ no free lunch: space/time scale with $|\mathcal{I}| \leq d_{\text{eff}}^n(\gamma)$
- $\widetilde{\mathcal{O}}(\mathbf{d}_{\text{eff}}^{\mathbf{n}}(\gamma)^{\mathbf{2}} + \mathbf{d}_{\text{eff}}^{\mathbf{n}}(\gamma)\mathbf{d})$ space constant in n

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = (\frac{1+2\varepsilon}{1-2\varepsilon})$ *and* $\gamma > 1$. *For any* $0 \leq \varepsilon \leq 1$, *and* $0 \leq \delta \leq 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \geq \frac{26\alpha}{\varepsilon^2} \log(\frac{\mathbf{n}}{\delta})$), *then w.p.* $1-\delta$, *for all nodes* $\{h, l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε, γ)*-accurate.*

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \leq \mathcal{O}(\overline{q} d_{\mathit{eff}}(\gamma)_{\{h,l\}}) \leq \mathcal{O}(\frac{\alpha}{\varepsilon^2} d^n_{\mathit{eff}}(\gamma) \log(\frac{n}{\delta}))$).

- Accuracy/dictionary size match oracle RLS-sampling at any time
  - ↳ no free lunch: space/time scale with $|\mathcal{I}| \leq d^n_{\text{eff}}(\gamma)$
- $\widetilde{\mathcal{O}}(\mathbf{d}^{\mathbf{n}}_{\text{eff}}(\gamma)^2 + \mathbf{d}^{\mathbf{n}}_{\text{eff}}(\gamma)\mathbf{d})$ space constant in n
- Merge tree fixed in advance

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = (\frac{1+2\varepsilon}{1-2\varepsilon})$ *and* $\gamma > 1$. *For any* $0 \leq \varepsilon \leq 1$, *and* $0 \leq \delta \leq 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \geq \frac{\mathbf{26}\alpha}{\varepsilon^{\mathbf{2}}} \log(\frac{\mathbf{n}}{\delta}))$, *then w.p.* $1-\delta$, *for all nodes* $\{h, l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε, γ)*-accurate*.

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \leq \mathcal{O}(\overline{q} d_{\text{eff}}(\gamma)_{\{h,l\}}) \leq \mathcal{O}(\frac{\alpha}{\varepsilon^2} d^n_{\text{eff}}(\gamma) \log(\frac{n}{\delta})))$.

- Runtime: single merge $\mathcal{O}(|\mathcal{I}_n|^3) \leq \widetilde{\mathcal{O}}(d^n_{\text{eff}}(\gamma)^3)$
  ↳ total depends on specific merge tree

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = \left(\frac{1+2\varepsilon}{1-2\varepsilon}\right)$ *and* $\gamma > 1$. *For any* $0 \leq \varepsilon \leq 1$, *and* $0 \leq \delta \leq 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \geq \frac{\mathbf{26}\alpha}{\varepsilon^{\mathbf{2}}} \log(\frac{\mathbf{n}}{\delta})$, *then w.p.* $1-\delta$, *for all nodes* $\{h, l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε, γ)*-accurate.*

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \leq \mathcal{O}(\overline{q} d_{\mathit{eff}}(\gamma)_{\{h,l\}}) \leq \mathcal{O}(\frac{\alpha}{\varepsilon^2} d^n_{\mathit{eff}}(\gamma) \log(\frac{n}{\delta}))$.

- Runtime: single merge $\mathcal{O}(|\mathcal{I}_n|^3) \leq \widetilde{\mathcal{O}}(d^n_{\text{eff}}(\gamma)^3)$
  ↳ total depends on specific merge tree
- Fully unbalanced tree: ~~$\mathcal{O}(n^3)$~~ $\Rightarrow \widetilde{\mathcal{O}}(n d^n_{\text{eff}}(\gamma)^3)$ on a single machine

DISQUEAK

Theorem (Calandriello et al., 2017a)

Let $\alpha = \left(\frac{1+2\varepsilon}{1-2\varepsilon}\right)$ *and* $\gamma > 1$. *For any* $0 \le \varepsilon \le 1$, *and* $0 \le \delta \le 1$, *if we run* DISQUEAK *with* $\overline{\mathbf{q}} \ge \frac{\mathbf{26}\alpha}{\varepsilon^2} \log\left(\frac{\mathbf{n}}{\delta}\right)$, *then w.p.* $1-\delta$, *for all nodes* $\{h,l\}$

(1) *The dictionary* $\mathcal{I}_{\{h,l\}}$ *is* (ε,γ)*-accurate.*

(2) $|\mathcal{I}_{\{\mathbf{h},\mathbf{l}\}}| \le \mathcal{O}(\overline{q} d_{\mathit{eff}}(\gamma)_{\{h,l\}}) \le \mathcal{O}\left(\frac{\alpha}{\varepsilon^2} d^n_{\mathit{eff}}(\gamma) \log\left(\frac{n}{\delta}\right)\right)$.

- Runtime: single merge $\mathcal{O}(|\mathcal{I}_n|^3) \le \widetilde{\mathcal{O}}(d^n_{\text{eff}}(\gamma)^3)$
  ↳ total depends on specific merge tree
- Fully unbalanced tree: $\mathcal{O}($~~n^3~~$)$ $\Rightarrow$ $\widetilde{\mathcal{O}}(n d^n_{\text{eff}}(\gamma)^3)$ on a single machine
- Fully balanced tree: $\widetilde{\mathcal{O}}(\log(n) d^n_{\text{eff}}(\gamma)^3)$ time, $\widetilde{\mathcal{O}}(n d^n_{\text{eff}}(\gamma)^3)$ work!

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1
Oracle RLS sampling	$n +$ 	$d_{\text{eff}}^{n}(\gamma)\log(n)$	Many

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1
Oracle RLS sampling	$n +$ 	$d_{\text{eff}}^{n}(\gamma)\log(n)$	Many
Exact RLS sampling	n^3	$d_{\text{eff}}^{n}(\gamma)\log(n)$	Many

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1
Oracle RLS sampling	$n +$ 	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Exact RLS sampling	n^3	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Alaoui and Mahoney, 2015	$n^3\mu(\gamma)^2$	$n\mu(\gamma) + d_{\text{eff}}^n(\gamma)\log(n)$	3

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1
Oracle RLS sampling	$n +$ 	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Exact RLS sampling	n^3	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Alaoui and Mahoney, 2015	$n^3\mu(\gamma)^2$	$n\mu(\gamma) + d_{\text{eff}}^n(\gamma)\log(n)$	3
SQUEAK/DISQUEAK Calandriello et al., 2017a	$(n/k)d_{\text{eff}}^n(\gamma)^3$	$d_{\text{eff}}^n(\gamma)\log(n)$	1

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

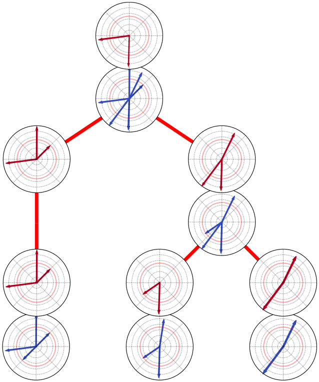
	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma) +$ 	$n\mu(\gamma)$	1
Oracle RLS sampling	$n +$ 	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Exact RLS sampling	n^3	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Alaoui and Mahoney, 2015	$n^3\mu(\gamma)^2$	$n\mu(\gamma) + d_{\text{eff}}^n(\gamma)\log(n)$	3
SQUEAK/DISQUEAK Calandriello et al., 2017a	$(n/k)d_{\text{eff}}^n(\gamma)^3$	$d_{\text{eff}}^n(\gamma)\log(n)$	1
KORS Calandriello et al., 2017c	$nd_{\text{eff}}^n(\gamma)^2$	$d_{\text{eff}}^n(\gamma)\log^2(n)$	1

Comparison

 = oracle, $\mu(\gamma) = \max_i \tau_{n,i}(\gamma) \leq 1/\gamma$ regularized coherence

	$\widetilde{\mathcal{O}}$(Runtime)	$\mathcal{O}(\lvert\mathcal{I}_n\rvert)$	Passes
Bach, 2013 (Uniform)	$n\mu(\gamma)$ + 	$n\mu(\gamma)$	1
Oracle RLS sampling	n + 	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Exact RLS sampling	n^3	$d_{\text{eff}}^n(\gamma)\log(n)$	Many
Alaoui and Mahoney, 2015	$n^3\mu(\gamma)^2$	$n\mu(\gamma) + d_{\text{eff}}^n(\gamma)\log(n)$	3
SQUEAK/DISQUEAK Calandriello et al., 2017a	$(n/k)d_{\text{eff}}^n(\gamma)^3$	$d_{\text{eff}}^n(\gamma)\log(n)$	1
KORS Calandriello et al., 2017c	$nd_{\text{eff}}^n(\gamma)^2$	$d_{\text{eff}}^n(\gamma)\log^2(n)$	1
Musco and Musco, 2017	$nd_{\text{eff}}^n(\gamma)^2$	$d_{\text{eff}}^n(\gamma)\log(n)$	$\log(n)$

Proof sketch



$$\widetilde{p}_{1,i} \propto \widetilde{\tau}_{1,i},$$
$$z_{1,i} = \mathbb{I}\{Ber(\widetilde{p}_{1,i})\}$$
$$\widetilde{p}_{2,i} \propto \widetilde{\tau}_{2,i},$$
$$z_{2,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{2,i}}{\widetilde{p}_{1,i}}\right)\right\} z_{1,i}$$
$$\widetilde{p}_{3,i} \propto \widetilde{\tau}_{3,i},$$
$$z_{3,i} = \mathbb{I}\left\{Ber\left(\frac{\widetilde{p}_{3,i}}{\widetilde{p}_{2,i}}\right)\right\} z_{2,i}$$

dependent chains

of dependent coin flip

Proof sketch

Similar to importance sampling. If the $\widetilde{p}_{t,i}$ were fixed in advance

$$\mathbb{P}(z_{t,i,j} = 1) = \mathbb{P}(\mathcal{B}(\widetilde{p}_{t,i}/\widetilde{p}_{t-1,i}) = 1)\mathbb{P}(z_{t-1,i,j} = 1)$$

Proof sketch

Need to bound

$$\mathbb{P}\left(\exists t \in \{1, \ldots, n\} : \|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \geq \varepsilon \cup |\mathcal{I}_t| \geq 3\overline{q} d_{\text{eff}}(\gamma)_t\right)$$

Proof sketch

Need to bound

$$\mathbb{P}\left(\exists t \in \{1,\ldots,n\} : \|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \geq \varepsilon \cup |\mathcal{I}_t| \geq 3\overline{q}d_{\text{eff}}(\gamma)_t\right)$$

After a union bound

$$\sum_{t=1}^{n} \mathbb{P}\left(\|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \geq \varepsilon\right)$$
$$+ \sum_{t=1}^{n} \mathbb{P}\left(|\mathcal{I}_t| \geq 3\overline{q}d_{\text{eff}}(\gamma)_t \cap \left\{\forall t' \in \{1,\ldots,t\} : \|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \leq \varepsilon\right\}\right)$$

Proof sketch

We start by bounding $\mathbb{P}\left(\|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \geq \varepsilon\right)$. Let

$$z_{s,i,j} = \mathbb{I}\left\{u_{s,i,j} \leq \frac{\widetilde{p}_{s,i}}{\widetilde{p}_{s-1,i}}\right\} z_{s-1,i,j}, \qquad \mathbf{v}_i = (\mathbf{K}_t + \gamma \mathbf{I})^{-1} \mathbf{K}_t^{1/2} \mathbf{e}_{t,i}$$

with $u_{s,i,j} \sim \mathcal{U}(0,1)$. Then

$$\mathbf{Y}_t = \mathbf{P}_t - \widetilde{\mathbf{P}}_t = \frac{1}{\overline{q}} \sum_{i=1}^{t} \sum_{j=1}^{\overline{q}} \left(1 - \frac{z_{t,i,j}}{\widetilde{p}_{t,i}}\right) \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}}$$

Proof sketch

We start by bounding $\mathbb{P}\left(\|\mathbf{P}_t - \widetilde{\mathbf{P}}_t\|_2 \geq \varepsilon\right)$. Let

$$z_{s,i,j} = \mathbb{I}\left\{u_{s,i,j} \leq \frac{\widetilde{p}_{s,i}}{\widetilde{p}_{s-1,i}}\right\} z_{s-1,i,j}, \qquad \mathbf{v}_i = (\mathbf{K}_t + \gamma\mathbf{I})^{-1}\mathbf{K}_t^{1/2}\mathbf{e}_{t,i}$$

with $u_{s,i,j} \sim \mathcal{U}(0,1)$. Then

$$\mathbf{Y}_t = \mathbf{P}_t - \widetilde{\mathbf{P}}_t = \frac{1}{\overline{q}}\sum_{i=1}^{t}\sum_{j=1}^{\overline{q}}\left(1 - \frac{z_{t,i,j}}{\widetilde{p}_{t,i}}\right)\mathbf{v}_i\mathbf{v}_i^{\mathsf{T}}$$

Cannot use concentrations for independent r.v., because $z_{t,i,j}$ and $z_{t,i',j'}$ are both dependent on $z_{t-1,i'',j''}$ through the estimates.

Proof sketch

Build the martingale

$$\mathbf{X}_{\{s,i,j\}} = \left(\frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} - \frac{z_{t,i,j}}{\widetilde{p}_{s,i}} \right) \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}}$$

We can use variants of Bernstein's inequality for matrix martingales, we need a bound on the range

$$\begin{aligned}
\|\mathbf{X}_{\{s,i,j\}}\| &= \frac{1}{q} \left| \left(\frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} - \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right) \right| \|\mathbf{v}_i \mathbf{v}_i^{\mathsf{T}}\| \leq \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \|\mathbf{v}_i\|^2 \\
&\leq \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{v_i}^{\mathsf{T}} \mathbf{v_i} = \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{e_i}^{\mathsf{T}} \mathbf{K_t}^{1/2} (\mathbf{K_t} + \gamma \mathbf{I})^{-1} \mathbf{K_t}^{1/2} \mathbf{e_i} \\
&= \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{e_i}^{\mathsf{T}} \mathbf{P_t} \mathbf{e_i} = \frac{1}{q} \frac{\tau_{\mathbf{t,i}}}{\widetilde{p}_{s,i}} \leq \frac{\alpha}{q} \frac{\tau_{t,i}}{p_{s,i}} = \frac{\alpha}{q} \frac{\tau_{t,i}}{\tau_{s,i}} \leq \frac{\alpha}{q} := R,
\end{aligned}$$

Proof sketch

Build the martingale

$$\mathbf{X}_{\{s,i,j\}} = \left(\frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} - \frac{z_{t,i,j}}{\widetilde{p}_{s,i}} \right) \mathbf{v}_i \mathbf{v}_i^\mathsf{T}$$

We can use variants of Bernstein's inequality for matrix martingales, we need a bound on the range

$$\begin{aligned}
\|\mathbf{X}_{\{s,i,j\}}\| &= \frac{1}{q} \left| \left(\frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} - \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right) \right| \|\mathbf{v}_i \mathbf{v}_i^\mathsf{T}\| \leq \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \|\mathbf{v}_i\|^2 \\
&\leq \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{v_i^\mathsf{T} v_i} = \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{e_i^\mathsf{T} K_t^{1/2} (K_t + \gamma I)^{-1} K_t^{1/2} e_i} \\
&= \frac{1}{q} \frac{1}{\widetilde{p}_{s,i}} \mathbf{e_i^\mathsf{T} P_t e_i} = \frac{1}{q} \frac{\tau_{\mathbf{t,i}}}{\widetilde{p}_{s,i}} \leq \frac{\alpha}{q} \frac{\tau_{t,i}}{p_{s,i}} = \frac{\alpha}{q} \frac{\tau_{t,i}}{\tau_{s,i}} \leq \frac{\alpha}{q} := R,
\end{aligned}$$

RLS normalize our r.v.

Proof sketch

Now bound the total variation

$$\mathbf{W} = \sum \mathbb{E}\left[\mathbf{X}_{\{s,i,j\}}^2 \;\middle|\; \{\mathbf{X}_r\}_{r=0}^{\{s,i,j\}-1}\right]$$

$$= \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \sum_{s=1}^{t} \frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} \left(\frac{1}{\widetilde{p}_{s,i}} - \frac{1}{\widetilde{p}_{s-1,i}}\right) \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \mathbf{v}_i \mathbf{v}_i^\mathsf{T}$$

Proof sketch

Now bound the total variation

$$\mathbf{W} = \sum \mathbb{E}\left[\mathbf{X}^2_{\{s,i,j\}} \;\middle|\; \{\mathbf{X}_r\}_{r=0}^{\{s,i,j\}-1}\right]$$
$$= \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \sum_{s=1}^{t} \frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} \left(\frac{1}{\widetilde{p}_{s,i}} - \frac{1}{\widetilde{p}_{s-1,i}}\right) \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \mathbf{v}_i \mathbf{v}_i^\mathsf{T}$$

Deterministically

$$\|\mathbf{W}\| = \left\| \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \sum_{s=1}^{t} \frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} \left(\frac{1}{\widetilde{p}_{s,i}} - \frac{1}{\widetilde{p}_{s-1,i}}\right) \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \right\|$$
$$\leq \left\| \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \frac{\mathbf{v}_i^\mathsf{T} \mathbf{v}_i}{\widetilde{p}_{t,i}^2} \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \right\| \leq \left\| \frac{\alpha}{\overline{q}} \sum_{i=1}^{t} \frac{1}{\widetilde{p}_{t,i}} \mathbf{v}_i \mathbf{v}_i^\mathsf{T} \right\|$$
$$\leq \left\| \frac{\alpha^2}{\overline{q}} \sum_{i=1}^{t} \mathbf{I} \right\| = \frac{\alpha^2}{\overline{q}} t$$

Proof sketch

Now bound the total variation

$$\mathbf{W} = \sum \mathbb{E}\left[\mathbf{X}^2_{\{s,i,j\}} \,\middle|\, \{\mathbf{X}_r\}_{r=0}^{\{s,i,j\}-1}\right]$$

$$= \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \sum_{s=1}^{t} \frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} \left(\frac{1}{\widetilde{p}_{s,i}} - \frac{1}{\widetilde{p}_{s-1,i}}\right) \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}}$$

Deterministically

$$\|\mathbf{W}\| = \left\| \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \sum_{s=1}^{t} \frac{z_{s-1,i,j}}{\widetilde{p}_{s-1,i}} \left(\frac{1}{\widetilde{p}_{s,i}} - \frac{1}{\widetilde{p}_{s-1,i}}\right) \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \right\|$$

$$\leq \left\| \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \frac{\mathbf{v}_i^{\mathsf{T}} \mathbf{v}_i}{\widetilde{p}_{t,i}^2} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \right\| \leq \left\| \frac{\alpha}{\overline{q}} \sum_{i=1}^{t} \frac{1}{\widetilde{p}_{t,i}} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \right\|$$

$$\leq \left\| \frac{\alpha^2}{\overline{q}} \sum_{i=1}^{t} \mathbf{I} \right\| = \frac{\alpha^2}{\overline{q}} t$$

Deterministic bound on variance too large

Proof sketch

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero. We should take advantage of that.

Proof sketch

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero. We should take advantage of that.

We can use a finer concentration, Freedman's inequality, that treats $\mathbf{W}$ itself as a random variable.

$$\mathbb{P}\left(\|\mathbf{Y}_t\| \geq \varepsilon \ \cap \ \|\mathbf{W}\| \leq \sigma^2\right) \leq t \exp\{-\dots\}$$

Proof sketch

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero. We should take advantage of that.

We can use a finer concentration, Freedman's inequality, that treats $\mathbf{W}$ itself as a random variable.

$$\mathbb{P}\left(\|\mathbf{Y}_t\| \geq \varepsilon \ \cap \ \|\mathbf{W}\| \leq \sigma^2\right) \leq t \exp\{- \dots\}$$

Starting from an upper bound on $\mathbf{W}$ that is still a r.v.

$$\mathbf{W} \preceq \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{i=1}^{t} \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2} \right\} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}} \mathbf{v}_i \mathbf{v}_i^{\mathsf{T}}$$

Proof sketch

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero. We should take advantage of that.

We can use a finer concentration, Freedman's inequality, that treats $\mathbf{W}$ itself as a random variable.

$$\mathbb{P}\left(\|\mathbf{Y}_t\| \geq \varepsilon \ \cap \ \|\mathbf{W}\| \leq \sigma^2\right) \leq t\exp\{-\dots\}$$

Starting from an upper bound on $\mathbf{W}$ that is still a r.v.

$$\mathbf{W} \preceq \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{i=1}^{t} \max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}\mathbf{v}_i\mathbf{v}_i^\mathsf{T}\mathbf{v}_i\mathbf{v}_i^\mathsf{T}$$

This still has high variance: cannot simply apply martingale Bernstein

Proof sketch

$\max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2} \right\}$ is still hard to analyze, since it is the maximum of dependent variables

Proof sketch

$\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ is still hard to analyze, since it is the maximum of dependent variables

Moreover $\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ depends on $\max_{s=0}^{t-1}\left\{\frac{z_{s,i',j'}}{\widetilde{p}_{s,i'}^2}\right\}$

Proof sketch

$\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ is still hard to analyze, since it is the maximum of dependent variables

Moreover $\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ depends on $\max_{s=0}^{t-1}\left\{\frac{z_{s,i',j'}}{\widetilde{p}_{s,i'}^2}\right\}$

We will find another set of dominating r.v. $1/w_{i,j}$, indep. from each other
Then apply Bernstein for indep. r.v.

Proof sketch

$\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ is still hard to analyze, since it is the maximum of dependent variables

Moreover $\max_{s=0}^{t-1}\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}^2}\right\}$ depends on $\max_{s=0}^{t-1}\left\{\frac{z_{s,i',j'}}{\widetilde{p}_{s,i'}^2}\right\}$

We will find another set of dominating r.v. $1/w_{i,j}$, indep. from each other
Then apply Bernstein for indep. r.v.

Random variable A stochastically dominates random variable B, if for all values a the two equivalent conditions are verified

$$\mathbb{P}(A \geq a) \geq \mathbb{P}(B \geq a) \Leftrightarrow \mathbb{P}(A \leq a) \leq \mathbb{P}(B \leq a).$$

Proof sketch

Similar to importance sampling. If the $\widetilde{p}_{t,i}$ were fixed in advance

$$\begin{aligned}\mathbb{P}(z_{t,i,j} = 1) &= \mathbb{P}(\mathcal{B}(\widetilde{p}_{t,i}/\widetilde{p}_{t-1,i}) = 1)\mathbb{P}(z_{t-1,i,j} = 1) \\ &= \frac{\widetilde{p}_{t,i}}{\widetilde{p}_{t-1,i}}\mathbb{P}(z_{t-1,i,j} = 1)\end{aligned}$$

Proof sketch

Similar to importance sampling. If the $\widetilde{p}_{t,i}$ were fixed in advance

$$\begin{aligned}
\mathbb{P}(z_{t,i,j} = 1) &= \mathbb{P}(\mathcal{B}(\widetilde{p}_{t,i}/\widetilde{p}_{t-1,i}) = 1)\mathbb{P}(z_{t-1,i,j} = 1) \\
&= \frac{\widetilde{p}_{t,i}}{\widetilde{p}_{t-1,i}}\mathbb{P}(z_{t-1,i,j} = 1) \\
&= \frac{\widetilde{p}_{t,i}}{\widetilde{p}_{t-1,i}}\frac{\widetilde{p}_{t-1,i}}{\widetilde{p}_{t-2,i}}\cdots\frac{\widetilde{p}_{i+1,i}}{\widetilde{p}_{i,i}}\frac{\widetilde{p}_{i,i}}{1} = \widetilde{p}_{t,i}
\end{aligned}$$

Proof sketch

Similar to importance sampling. If the $\widetilde{p}_{t,i}$ were fixed in advance

$$
\begin{aligned}
\mathbb{P}(z_{t,i,j} = 1) &= \mathbb{P}(\mathcal{B}(\widetilde{p}_{t,i}/\widetilde{p}_{t-1,i}) = 1)\mathbb{P}(z_{t-1,i,j} = 1) \\
&= \frac{\widetilde{p}_{t,i}}{\widetilde{p}_{t-1,i}}\mathbb{P}(z_{t-1,i,j} = 1) \\
&= \frac{\widetilde{p}_{t,i}}{\widetilde{p}_{t-1,i}}\frac{\widetilde{p}_{t-1,i}}{\widetilde{p}_{t-2,i}}\cdots\frac{\widetilde{p}_{i+1,i}}{\widetilde{p}_{i,i}}\frac{\widetilde{p}_{i,i}}{1} = \widetilde{p}_{t,i}
\end{aligned}
$$

Weight increase along chain $\frac{z_{t-1,i,j}}{\widetilde{p}_{t-1,i}} \leq \frac{z_{t,i,j}}{\widetilde{p}_{t,i}}$ until $z_{t,i,j} = 0$ or $\frac{1}{\widetilde{p}_{n,i}} \lessapprox \frac{1}{\tau_{n,i}}$.

Proof sketch

Predictable quadratic variation $\mathbf{W}$ of a chain scales (roughly) with

$$\|\mathbf{W}\|_2^2 \sim \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\}$$

Proof sketch

Predictable quadratic variation $\mathbf{W}$ of a chain scales (roughly) with

$$\|\mathbf{W}\|_2^2 \sim \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\}$$

Cannot use concentrations for independent r.v.

Proof sketch

Predictable quadratic variation $\mathbf{W}$ of a chain scales (roughly) with

$$\|\mathbf{W}\|_2^2 \sim \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\}$$

Cannot use concentrations for independent r.v.

And in worst case $\max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\} \lessapprox \frac{1}{\tau_{t,i}} \leq t$

Proof sketch

Predictable quadratic variation $\mathbf{W}$ of a chain scales (roughly) with

$$\|\mathbf{W}\|_2^2 \sim \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\}$$

Cannot use concentrations for independent r.v.

And in worst case $\max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\} \lessapprox \frac{1}{\tau_{t,i}} \leq t$

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero.

Proof sketch

Predictable quadratic variation $\mathbf{W}$ of a chain scales (roughly) with

$$\|\mathbf{W}\|_2^2 \sim \max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\}$$

Cannot use concentrations for independent r.v.

And in worst case $\max_{s=0}^{t-1} \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\} \lessapprox \frac{1}{\tau_{t,i}} \leq t$

This looks too pessimistic. When $\frac{1}{\widetilde{p}_{s,i}}$ is large, $z_{s,i,j}$ should be zero.

We will find another set of dominating r.v. $\frac{1}{w_{i,j}}$, indep. from each other

$$\mathbb{P}\left(\max \left\{ \frac{z_{s,i,j}}{\widetilde{p}_{s,i}} \right\} \leq a \right) \geq \mathbb{P}\left(\frac{1}{w_{i,j}} \leq a \right)$$

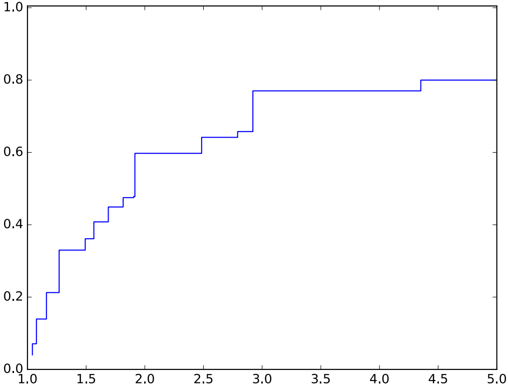
Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then

Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then

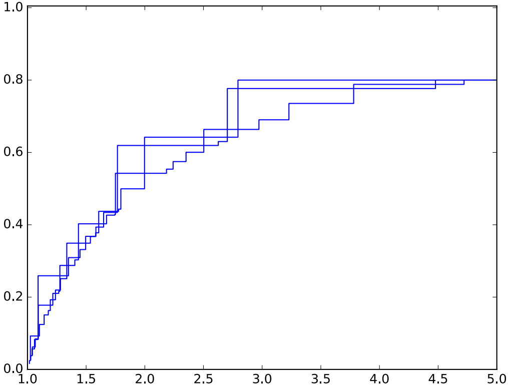
$$\mathbb{P}\left(\max\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}}\right\} \leq a\right) =$$



Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then

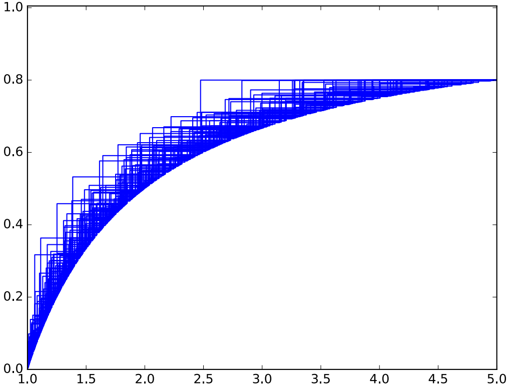
$$\mathbb{P}\left(\max\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}}\right\} \leq a\right) =$$



Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then

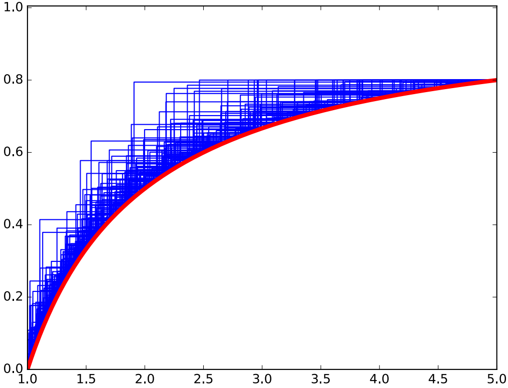
$$\mathbb{P}\left(\max\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}}\right\} \leq a\right) =$$



Proof sketch

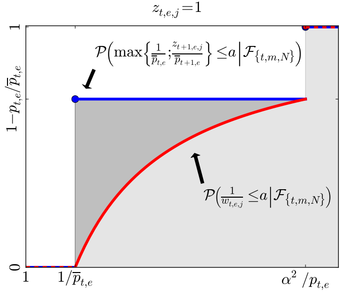
Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then

$$\mathbb{P}\left(\max\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}}\right\} \leq a\right) =$$



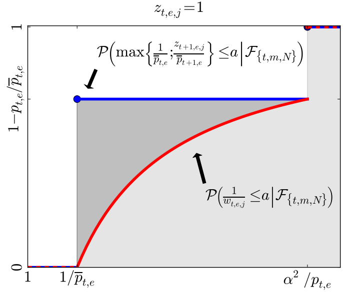
Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then



Proof sketch

Imagine the $\widetilde{p}_{s,i}$ were fixed in advance. Then



$$\mathbb{P}\left(\max\left\{\frac{z_{s,i,j}}{\widetilde{p}_{s,i}}\right\} \leq a\right) \geq \mathbb{P}\left(\frac{1}{w_{0,i,j}} \leq a\right) = \begin{cases} 0 & \text{for} \quad a < 1 \\ 1 - \frac{1}{a} & \text{for} \quad 1 \leq a < \alpha/p_{t,i} \\ 1 & \text{for} \quad \alpha/p_{t,i} \leq a \end{cases},$$

SQUEAK- recap before application

Goal 1: find a small, provably accurate dictionary in near-linear time

SQUEAK and DISQUEAK

Sub-linear time using multiple machines

Final dictionary can be updated if new samples arrive

SQUEAK- recap before application

Goal 1: find a small, provably accurate dictionary in near-linear time

SQUEAK and DISQUEAK

- Sub-linear time using multiple machines
- Final dictionary can be updated if new samples arrive
- Novel analysis, potentially useful for general importance sampling

SQUEAK- recap before application

Goal 1: find a small, provably accurate dictionary in near-linear time

SQUEAK and DISQUEAK

- Sub-linear time using multiple machines
- Final dictionary can be updated if new samples arrive
- Novel analysis, potentially useful for general importance sampling

Future work

- Experiments
  - ↳ Easy to implement: distributed task queue
    Preliminary results promising, easily scales to 1M+ samples

SQUEAK- recap before application

Goal 1: find a small, provably accurate dictionary in near-linear time

SQUEAK and DISQUEAK

- Sub-linear time using multiple machines
- Final dictionary can be updated if new samples arrive
- Novel analysis, potentially useful for general importance sampling

Future work

- Experiments
  - ↳ Easy to implement: distributed task queue
    Preliminary results promising, easily scales to 1M+ samples
- Beyond passive processing: SQUEAK for active learning

Efficient Sequential Learning

in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Low-rank PSD matrix approximation

Kernel matrix $\mathbf{K}_n$

Kernel PCA

Kernel Regression

[Alaoui and Mahoney, 2015; Bach, 2013; Rudi et al., 2015]

Kernel K-Means

[Musco and Musco, 2017]

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Low-rank PSD matrix approximation

Kernel matrix $\mathbf{K}_n$

Kernel PCA

Kernel Regression

[Alaoui and Mahoney, 2015; Bach, 2013; Rudi et al., 2015]

Kernel K-Means

[Musco and Musco, 2017]

Graph Laplacians $\mathbf{L}_{\mathcal{G}}$

Graph Semi-Supervised Learning

[Calandriello et al., 2015]

Graph Sparsification

[Calandriello et al., 2016]

Efficient Sequential Learning

in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Low-rank PSD matrix approximation

Hessian (convex function)

Efficient Sequential Learning
in Structured and Constrained Environments

Goal 2: use dictionary to solve down-stream problems efficiently

Low-rank PSD matrix approximation

Hessian (convex function)

Batch Conjugate gradient

[Rudi et al., 2017]

Online Newton Step (second part of talk)

[Calandriello et al., 2017b; Calandriello et al., 2017c]

Outline

(1) Dictionary learning

- ▷ Nyström sampling
- ▷ ridge leverage scores and effective dimension
- ▷ SQUEAK: sequential RLS importance sampling
  - ↳ analysis for non i.i.d. matrix sampling

(2) Online Kernel Learning

- ▷ online kernel learning and kernelized online Newton step
- ▷ PROS-N-KONS: adaptive Nyström embedding for online kernel learning
- ▷ adaptive restarts
- ▷ regression and classification experiments

Online Kernel Learning (OKL)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Online Kernel Learning (OKL)

Online game between learner and adversary, at each round $t \in [T]$

1. the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$
2. the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,
3. the adversary reveals the curved loss ℓ_t,
4. the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

Kernel flexible but curse of kernelization

t parameters $\Rightarrow$ $\mathcal{O}(t)$ per-step prediction cost

$\mathbf{g}_t = \ell_t'(\phi_t^\mathsf{T}\mathbf{w}_t)\phi_t := \dot{g}_t\phi_t$

Online Kernel Learning (OKL)

Online game between learner and adversary, at each round $t \in [T]$

1 the adversary reveals a new point $\varphi(\mathbf{x}_t) = \phi_t \in \mathcal{H}$

2 the learner chooses a function $f_{\mathbf{w}_t}$ and predicts $f_{\mathbf{w}_t}(\mathbf{x}_t) = \varphi(\mathbf{x}_t)^\mathsf{T}\mathbf{w}_t$,

3 the adversary reveals the curved loss ℓ_t,

4 the learner suffers $\ell_t(\phi_t^\mathsf{T}\mathbf{w}_t)$ and observes the associated gradient $\mathbf{g}_t$.

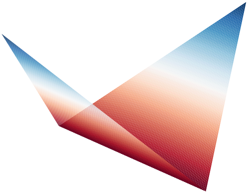
Kernel flexible but curse of kernelization

t parameters $\Rightarrow$ $\mathcal{O}(t)$ per-step prediction cost

$\mathbf{g}_t = \ell_t'(\phi_t^\mathsf{T}\mathbf{w}_t)\phi_t := \dot{g}_t\phi_t$

Learning to minimize regret $R(\mathbf{w}) = \sum_{t=1}^T \ell_t(\phi_t\mathbf{w}_t) - \ell_t(\phi_t\mathbf{w})$
and compete with best-in-hindsight $\mathbf{w}^* := \arg\min_{\mathbf{w}\in\mathcal{H}} \sum_{t=1}^T \ell_t(\phi_t\mathbf{w})$

OGD and losses

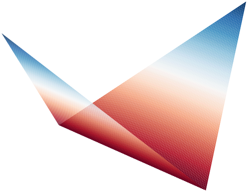


convex

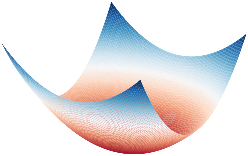
First order (GD) [Kivinen et al., 2004; Zinkevich, 2003]

$\sqrt{T}$ regret, $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step

OGD and losses



convex



strongly convex

First order (GD) [Kivinen et al., 2004; Zinkevich, 2003]
$\sqrt{T}$ regret, $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step

First order (GD) [Hazan et al., 2008]
$\log(T)$ regret,

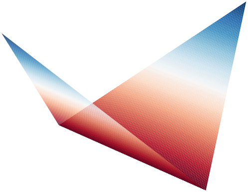
OGD and losses

convex **strongly convex**

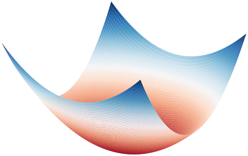
First order (GD) [Kivinen et al., 2004; Zinkevich, 2003]
$\sqrt{T}$ regret, $\mathcal{O}(d)/\mathcal{O}(t)$ time/space per-step

First order (GD) [Hazan et al., 2008]
$\log(T)$ regret, but often not satisfied in practice
↳(e.g. $(y_t - \phi_t^{\mathsf{T}} \mathbf{w}_t)^2$)

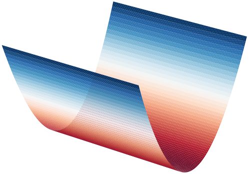
OGD and losses



convex



strongly convex

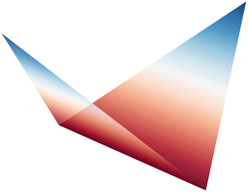


σ**-curved**

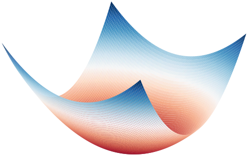
Second order (Newton-like) [Hazan et al., 2006; Zhdanov and Kalnishkan, 2010]

$\log(T)$ regret, $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

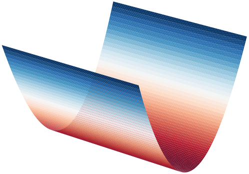
OGD and losses



convex



strongly convex



σ**-curved**

Second order (Newton-like) [Hazan et al., 2006; Zhdanov and Kalnishkan, 2010]
$\log(T)$ regret, $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

Weaker than strong convexity

OGD and losses

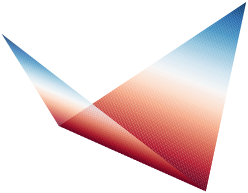
convex **strongly convex** **σ-curved**

Second order (Newton-like) [Hazan et al., 2006; Zhdanov and Kalnishkan, 2010]
$\log(T)$ regret, $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

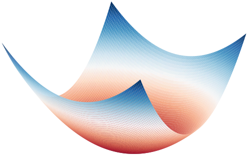
Weaker than strong convexity

Satisfied by exp-concave losses:
↳squared loss, squared hinge-loss, logistic loss

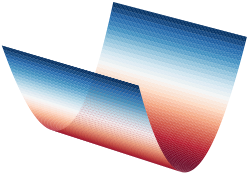
OGD and losses



convex



strongly convex



σ**-curved**

Second order (Newton-like) [Hazan et al., 2006; Zhdanov and Kalnishkan, 2010]
$\log(T)$ regret, $\mathcal{O}(d^2)/\mathcal{O}(t^2)$ time/space per-step

Weaker than strong convexity

Satisfied by exp-concave losses:
↳squared loss, squared hinge-loss, logistic loss

Assumptions:
ℓ_t are σ-curved and $|\ell_t'(z)| \leq L$ whenever $|z| \leq C$ (scalar Lipschitz)

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Regret [Hazan et al., 2006; Luo et al., 2016]

$$R(\mathbf{w}^*) \leq \overbrace{\alpha \|\mathbf{w}^* - \mathbf{w_0}\|_2^2}^{\text{initial error}} + \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} (\mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \mathbf{g}_t \right)$$

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Regret [Hazan et al., 2006; Luo et al., 2016]

$$R(\mathbf{w}^*) \leq \overbrace{\alpha \|\mathbf{w}^* - \mathbf{w}_0\|_2^2}^{\text{initial error}} + \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} (\mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \mathbf{g}_t \right)$$

$$\leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}\left(L \sum_{t=1}^{T} \phi_t^\mathsf{T} (\Phi_t \Phi_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \phi_t \right)$$

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Regret [Hazan et al., 2006; Luo et al., 2016]

$$R(\mathbf{w}^*) \leq \overbrace{\alpha \|\mathbf{w}^* - \mathbf{w}_0\|_2^2}^{\text{initial error}} + \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} (\mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \mathbf{g}_t \right)$$

$$\leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}\left(L \overbrace{\sum_{t=1}^{T} \phi_t^\mathsf{T} (\Phi_t \Phi_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \phi_t}^{\text{online effective dimension}} \right)$$

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Regret [Hazan et al., 2006; Luo et al., 2016]

$$R(\mathbf{w}^*) \leq \overbrace{\alpha \|\mathbf{w}^* - \mathbf{w_0}\|_2^2}^{\text{initial error}} + \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} (\mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \mathbf{g}_t \right)$$

$$\leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}\left(L \overbrace{\sum_{t=1}^{T} \phi_t^\mathsf{T} (\Phi_t \Phi_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \phi_t}^{\text{online effective dimension}} \right)$$

$$\leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(\log \operatorname{Det}(\mathbf{K}_T/\alpha + \mathbf{I}_n))$$

Second-Order OKL (Kernel Online Newton Step)

Second-Order Gradient Descent

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{A}_t^{-1}\mathbf{g}_t, \qquad \mathbf{A}_t = \sum_{s=1}^{t} \sigma \mathbf{g}_s \mathbf{g}_s^\mathsf{T} + \alpha \mathbf{I} = \mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I}$$

Regret [Hazan et al., 2006; Luo et al., 2016]

$$R(\mathbf{w}^*) \le \overbrace{\alpha \|\mathbf{w}^* - \mathbf{w_0}\|_2^2}^{\text{initial error}} + \mathcal{O}\left(\sum_{t=1}^{T} \mathbf{g}_t^\mathsf{T} (\mathbf{G}_t \mathbf{G}_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \mathbf{g}_t\right)$$

$$\le \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}\left(L \overbrace{\sum_{t=1}^{T} \phi_t^\mathsf{T} (\Phi_t \Phi_t^\mathsf{T} + \alpha \mathbf{I})^{-1} \phi_t}^{\text{online effective dimension}}\right)$$

$$\le \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(\log \operatorname{Det}(\mathbf{K}_T/\alpha + \mathbf{I}_n))$$

$$\le \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(d_{\text{eff}}^T(\alpha) \log(T))$$ [Calandriello et al., 2017c]

Effective Dimension in online learning

$$R(\mathbf{w}^*) \leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(d_{\text{eff}}^T(\alpha)\log(T))$$

$d_{\text{eff}}^T(\alpha)$ number of relevant orthogonal directions played by the adversary.

Every new orthogonal direction causes some regret.
↳ if it is played often enough (i.e., $\geq \alpha/(L\sigma)$)

Effective Dimension in online learning

$$R(\mathbf{w}^*) \le \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(d_{\text{eff}}^T(\alpha)\log(T))$$

$d_{\text{eff}}^T(\alpha)$ number of relevant orthogonal directions played by the adversary.

Every new orthogonal direction causes some regret.
↳if it is played often enough (i.e., $\geq \alpha/(L\sigma)$)

If all ϕ_t are orthogonal

$$d_{\text{eff}}^T(\sqrt{T}) \sim \sqrt{T}$$

and

$$R(\mathbf{w}^*) \le \sqrt{T} + \sqrt{T}\log(T) \sim \sqrt{T}$$

Effective Dimension in online learning

$$R(\mathbf{w}^*) \leq \alpha \|\mathbf{w}^* - \mathbf{w}_0\|^2 + \mathcal{O}(d_{\text{eff}}^T(\alpha) \log(T))$$

$d_{\text{eff}}^T(\alpha)$ number of relevant orthogonal directions played by the adversary.

Every new orthogonal direction causes some regret.
↳if it is played often enough (i.e., $\geq \alpha/(L\sigma)$)

If all ϕ_t are orthogonal

$$d_{\text{eff}}^T(\sqrt{T}) \sim \sqrt{T}$$

and

$$R(\mathbf{w}^*) \leq \sqrt{T} + \sqrt{T}\log(T) \sim \sqrt{T}$$

If ϕ_t from finite subspace

$$d_{\text{eff}}^T(1) \sim \mathcal{O}(1) \leq r$$

is constant in T and

$$R(\mathbf{w}^*) \leq \mathcal{O}(1) + \mathcal{O}(1)\log(T) \sim \log T$$

Approximating KONS

KONS: $d_{\text{eff}}^T(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow \mathcal{O}(t)$ prediction $\phi_t^\mathsf{T}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Approximating KONS

KONS: $d_{\text{eff}}^T(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow \mathcal{O}(t)$ prediction $\phi_t^\mathsf{T}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^T(\alpha)\log(T)$ regret, but prediction still depends on t

Approximating KONS

KONS: $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow \mathcal{O}(t)$ prediction $\phi_t^{\mathsf{T}}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret, but prediction still depends on t

Use exact second order updates in small approximate $\widetilde{\mathcal{H}}$

↳ replace φ with approximate map $\widetilde{\varphi}$ (random features, embeddings)

Approximating KONS

KONS: $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow \mathcal{O}(t)$ prediction $\phi_t^\mathsf{T}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^{T}(\alpha)\log(T)$ regret, but prediction still depends on t

Use exact second order updates in small approximate $\widetilde{\mathcal{H}}$

↳ replace φ with approximate map $\widetilde{\varphi}$ (random features, embeddings)

finite $\widetilde{\mathcal{H}} \Rightarrow$ constant per-step prediction/update cost

Approximating KONS

KONS: $d_{\text{eff}}^T(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow \mathcal{O}(t)$ prediction $\phi_t^{\mathsf{T}}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^T(\alpha)\log(T)$ regret, but prediction still depends on t

Use exact second order updates in small approximate $\widetilde{\mathcal{H}}$

↳ replace φ with approximate map $\widetilde{\varphi}$ (random features, embeddings)

finite $\widetilde{\mathcal{H}} \Rightarrow$ constant per-step prediction/update cost

$$\sum_{t=1}^{T} \ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\phi_t\mathbf{w}^*) = \sum_{t=1}^{T} \underbrace{\ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\widetilde{\phi}_t\overline{\mathbf{w}})}_{a} + \underbrace{\ell_t(\phi_t\overline{\mathbf{w}}) - \ell_t(\phi_t\mathbf{w}^*)}_{b}$$

Approximating KONS

KONS: $d_{\text{eff}}^T(\alpha)\log(T)$ regret

↳ large $\mathcal{H} \Rightarrow$ $\mathcal{O}(t)$ prediction $\phi_t^\mathsf{T}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^T(\alpha)\log(T)$ regret, but prediction still depends on t

Use exact second order updates in small approximate $\widetilde{\mathcal{H}}$

↳ replace φ with approximate map $\widetilde{\varphi}$ (random features, embeddings)

finite $\widetilde{\mathcal{H}} \Rightarrow$ constant per-step prediction/update cost

$$\sum_{t=1}^{T} \ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\phi_t\mathbf{w}^*) = \sum_{t=1}^{T} \underbrace{\ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\widetilde{\phi}_t\overline{\mathbf{w}})}_{a} + \underbrace{\ell_t(\phi_t\overline{\mathbf{w}}) - \ell_t(\phi_t\mathbf{w}^*)}_{b}$$

(a) Exact KONS in $\widetilde{\mathcal{H}}$: $d_{\text{eff}}^T(\alpha)\log(T)$

Approximating KONS

KONS: $d_{\text{eff}}^T(\alpha)\log(T)$ regret

↳ large $\mathcal{H}$ ⇒ $\mathcal{O}(t)$ prediction $\phi_t^\mathsf{T}\mathbf{w}_t$, $\mathcal{O}(t^2)$ updates $\mathbf{g}_t - \mathbf{A}_t^{-1}\mathbf{g}_t$

Use approximate second order updates in large $\mathcal{H}$ [Calandriello et al., 2017c]

↳ $d_{\text{eff}}^T(\alpha)\log(T)$ regret, but prediction still depends on t

Use exact second order updates in small approximate $\widetilde{\mathcal{H}}$

↳ replace φ with approximate map $\widetilde{\varphi}$ (random features, embeddings)

finite $\widetilde{\mathcal{H}}$ ⇒ constant per-step prediction/update cost

$$\sum_{t=1}^T \ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\phi_t\mathbf{w}^*) = \sum_{t=1}^T \underbrace{\ell_t(\widetilde{\phi}_t\widetilde{\mathbf{w}}_t) - \ell_t(\widetilde{\phi}_t\overline{\mathbf{w}})}_{a} + \underbrace{\ell_t(\phi_t\overline{\mathbf{w}}) - \ell_t(\phi_t\mathbf{w}^*)}_{b}$$

(a) Exact KONS in $\widetilde{\mathcal{H}}$: $d_{\text{eff}}^T(\alpha)\log(T)$

(b) error between $\overline{\mathbf{w}}$ best in $\widetilde{\mathcal{H}}$ and $\mathbf{w}^*$ best in $\mathcal{H}$: bound how?

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this
same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

$\widetilde{\mathcal{H}}_t = \text{Span}(\mathcal{I}_t)$ defined using m_t inducing points $\mathcal{I}_t = \{\phi_s\}_{s=1}^{m_t}$

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

$\widetilde{\mathcal{H}}_t = \text{Span}(\mathcal{I}_t)$ defined using m_t inducing points $\mathcal{I}_t = \{\phi_s\}_{s=1}^{m_t}$

Use RLS (KORS) to select inducing points

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

$\widetilde{\mathcal{H}}_t = \mathrm{Span}(\mathcal{I}_t)$ defined using m_t inducing points $\mathcal{I}_t = \{\phi_s\}_{s=1}^{m_t}$

Use RLS (KORS) to select inducing points

↳ SQUEAK without removal ($\mathcal{I}_t \subseteq \mathcal{I}_{t+1}$, $\widetilde{\mathcal{H}}_t \subseteq \widetilde{\mathcal{H}}_{t+1}$)

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

$\widetilde{\mathcal{H}}_t = \mathrm{Span}(\mathcal{I}_t)$ defined using m_t inducing points $\mathcal{I}_t = \{\phi_s\}_{s=1}^{m_t}$

Use RLS (KORS) to select inducing points

↳ SQUEAK without removal ($\mathcal{I}_t \subseteq \mathcal{I}_{t+1}$, $\widetilde{\mathcal{H}}_t \subseteq \widetilde{\mathcal{H}}_{t+1}$)

w.h.p. accurate and maximum size $|\widetilde{\mathcal{H}}_t| \leq \mathcal{O}(d_{\text{eff}}^T(\gamma) \log^2(T))$

Subspace approximation error

$\widetilde{\mathcal{H}}$ cannot be fixed

↳ the adversary will find orthogonal points and exploit this

same for fixed budget (e.g., k-rank approx [Luo et al., 2016])

Use Nyström approximation instead and adapt it online

↳ if the adversary plays a "sufficiently orthogonal" ϕ_t, add it to $\mathcal{I}_{t+1}$

$\widetilde{\mathcal{H}}_t = \mathrm{Span}(\mathcal{I}_t)$ defined using m_t inducing points $\mathcal{I}_t = \{\phi_s\}_{s=1}^{m_t}$

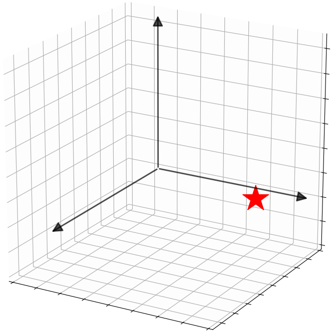
Use RLS (KORS) to select inducing points

↳ SQUEAK without removal ($\mathcal{I}_t \subseteq \mathcal{I}_{t+1}$, $\widetilde{\mathcal{H}}_t \subseteq \widetilde{\mathcal{H}}_{t+1}$)

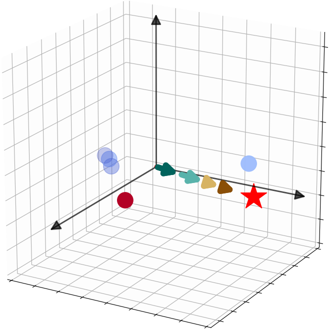
w.h.p. accurate and maximum size $|\widetilde{\mathcal{H}}_t| \leq \mathcal{O}(d_{\mathrm{eff}}^T(\gamma)\log^2(T))$

$\widetilde{\mathcal{O}}(d_{\mathrm{eff}}^T(\gamma)^2)$ time/space cost to run exact KONS in $\widetilde{\mathcal{H}}_t$

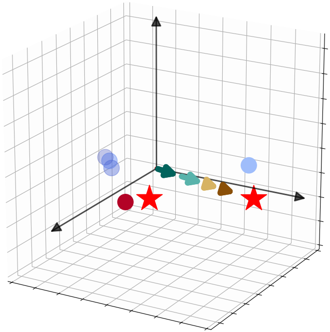
PROS-N-KONS



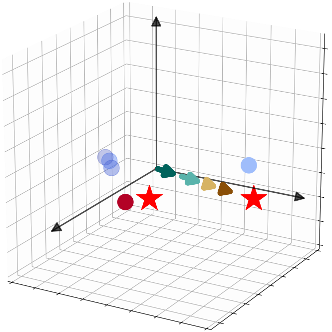
PROS-N-KONS



PROS-N-KONS



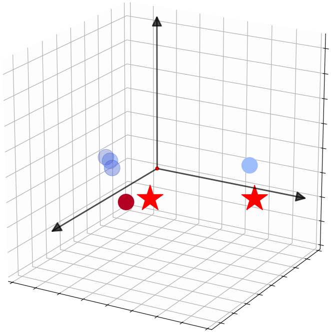
PROS-N-KONS



Every time we change $\widetilde{\mathcal{H}}$ we pay $\alpha\|\overline{\mathbf{w}}_j - \mathbf{w}_{t_j}\|_2^2$ (initial error in GD)

↳ the adversary can influence $\mathbf{w}_{t_j}$ and make it large

PROS-N-KONS



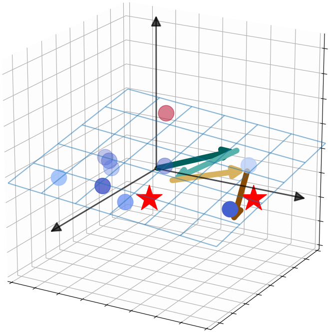
Reset $\widetilde{\mathbf{w}}_t$ and $\widetilde{\mathbf{A}}_t$ when $\widetilde{\mathcal{H}}_t$ changes

↳ wasteful, but not too often. At most $J \leq d_{\text{eff}}^T(\gamma)$ times.

learning is preserved through $\widetilde{\mathcal{H}}_t$ that always improves

adaptive doubling trick

PROS-N-KONS



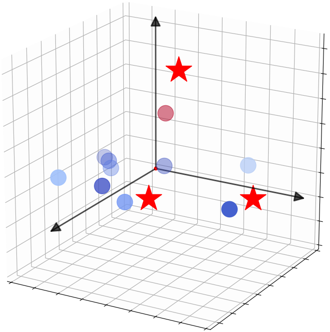
Reset $\widetilde{\mathbf{w}}_t$ and $\widetilde{\mathbf{A}}_t$ when $\widetilde{\mathcal{H}}_t$ changes

↳ wasteful, but not too often. At most $J \leq d_{\text{eff}}^T(\gamma)$ times.

learning is preserved through $\widetilde{\mathcal{H}}_t$ that always improves

adaptive doubling trick

PROS-N-KONS



Reset $\widetilde{\mathbf{w}}_t$ and $\widetilde{\mathbf{A}}_t$ when $\widetilde{\mathcal{H}}_t$ changes

↳ wasteful, but not too often. At most $J \leq d_{\text{eff}}^T(\gamma)$ times.

learning is preserved through $\widetilde{\mathcal{H}}_t$ that always improves

adaptive doubling trick

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma)\log^2(T)}_{\text{restarts}}(\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha)\log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}}\text{ gap}} /\alpha\Big),$$

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma)\log^2(T)}_{\text{restarts}}(\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha)\log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}}\text{ gap}}/\alpha\Big),$$

Setting $\gamma = \alpha/T$ removes second term

↳ regret/computational cost is $\widetilde{\mathcal{O}}(d_{\text{eff}}^T(1/T)^2)$

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma)\log^2(T)}_{\text{restarts}}(\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha)\log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}}\text{ gap}}/\alpha\Big),$$

Setting $\gamma = \alpha/T$ removes second term

↳ regret/computational cost is $\widetilde{\mathcal{O}}(d_{\text{eff}}^T(1/T)^2)$

still small in many cases, scale with eigenvalue decay

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma) \log^2(T)}_{\text{restarts}} (\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha) \log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}} \text{ gap}} /\alpha \Big),$$

Setting $\gamma = \alpha/T$ removes second term

↳ regret/computational cost is $\widetilde{\mathcal{O}}(d_{\text{eff}}^T(1/T)^2)$

still small in many cases, scale with eigenvalue decay

- If $\lambda_t = t^{-q}$, regret is $o(d_{\text{eff}}(1/T)) \leq o(T^{1/q})$

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma)\log^2(T)}_{\text{restarts}}(\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha)\log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}}\text{ gap}}/\alpha\Big),$$

Setting $\gamma = \alpha/T$ removes second term

↳ regret/computational cost is $\widetilde{\mathcal{O}}(d_{\text{eff}}^T(1/T)^2)$

still small in many cases, scale with eigenvalue decay

- If $\lambda_t = t^{-q}$, regret is $o(d_{\text{eff}}(1/T)) \leq o(T^{1/q})$
- If $\lambda_t = e^{-t}$ (Gaussian $\mathcal{H}$), regret is $o(\text{polylog}(T))$

Final regret guarantees

For any curved loss

$$R_T(\mathbf{w}) \leq \mathcal{O}\Big(\underbrace{d_{\text{eff}}^T(\gamma)\log^2(T)}_{\text{restarts}}(\alpha\|\mathbf{w}\|^2 + \underbrace{d_{\text{eff}}^T(\alpha)\log(T/\alpha)}_{\text{online-offline gap}}) + \underbrace{\gamma T}_{\mathcal{H}\text{-}\widetilde{\mathcal{H}}\text{ gap}} /\alpha\Big),$$

Setting $\gamma = \alpha/T$ removes second term

↳ regret/computational cost is $\widetilde{\mathcal{O}}(d_{\text{eff}}^T(1/T)^2)$

still small in many cases, scale with eigenvalue decay

- If $\lambda_t = t^{-q}$, regret is $o(d_{\text{eff}}(1/T)) \leq o(T^{1/q})$
- If $\lambda_t = e^{-t}$ (Gaussian $\mathcal{H}$), regret is $o(\text{polylog}(T))$
- If $\mathcal{H} = \mathbb{R}^d$ regret is $\mathcal{O}(r\log(T))$ [Luo et al., 2016]

Final regret guarantees

For squared loss only and $\gamma = \alpha$

$$R(\mathbf{w}^*) \leq \widetilde{\mathcal{O}}\left(J\left(\alpha\|\mathbf{w}^*\|_2^2 + d_{\text{eff}}^T(\alpha)\log(T/\alpha)\right) + J\mathcal{L}^*\right)$$

Final regret guarantees

For squared loss only and $\gamma = \alpha$

$$R(\mathbf{w}^*) \leq \widetilde{\mathcal{O}}\left(J\left(\alpha\|\mathbf{w}^*\|_2^2 + d_{\text{eff}}^T(\alpha)\log(T/\alpha)\right) + J\mathcal{L}^*\right)$$

Last term $\mathcal{L}^* = \sum_{t=1}^{T} \ell_t(\phi_t \mathbf{w}^*) + \alpha\|\mathbf{w}^*\|_2^2$ replaces $\frac{\gamma}{\alpha}T$

↳ regularized cumulative loss of $\mathbf{w}^*$, very small if $\mathcal{H}$ is good

Final regret guarantees

For squared loss only and $\gamma = \alpha$

$$R(\mathbf{w}^*) \leq \widetilde{\mathcal{O}}\left(J\left(\alpha\|\mathbf{w}^*\|_2^2 + d_{\text{eff}}^T(\alpha)\log(T/\alpha)\right) + J\mathcal{L}^*\right)$$

Last term $\mathcal{L}^* = \sum_{t=1}^T \ell_t(\phi_t \mathbf{w}^*) + \alpha\|\mathbf{w}^*\|_2^2$ replaces $\frac{\gamma}{\alpha}T$

↳ regularized cumulative loss of $\mathbf{w}^*$, very small if $\mathcal{H}$ is good

First-order regret bound, $\mathcal{L}^*$ constant if model is correct

↳ constant $\mathcal{H}$-$\widetilde{\mathcal{H}}$ gap is enough if instantaneous loss goes to 0.

Experiments - regression

$\alpha = 1, \gamma = 1$						
Algorithm	cadata $n = 20k$, $d = 8$			casp $n = 45k$, $d = 9$		
	Avg. Squared Loss	#SV	Time	Avg. Squared Loss	#SV	Time
FOGD	0.04097 $\pm$ 0.00015	30	—	0.08021 $\pm$ 0.00031	30	—
NOGD	0.03983 $\pm$ 0.00018	30	—	0.07844 $\pm$ 0.00008	30	—
PROS-N-KONS	0.03095 $\pm$ 0.00110	20	18.59	**0.06773** $\pm$ 0.00105	21	40.73
CON-KONS	**0.02850** $\pm$ 0.00174	19	18.45	**0.06832** $\pm$ 0.00315	20	40.91
B-KONS	0.03095 $\pm$ 0.00118	19	18.65	**0.06775** $\pm$ 0.00067	21	41.13
BATCH	0.02202 $\pm$ 0.00002	—	—	0.06100 $\pm$ 0.00003	—	—

Algorithm	slice $n = 53k$, $d = 385$			year $n = 463k$, $d = 90$		
	Avg. Squared Loss	#SV	Time	Avg. Squared Loss	#SV	Time
FOGD	**0.00726** $\pm$ 0.00019	30	—	0.01427 $\pm$ 0.00004	30	—
NOGD	0.02636 $\pm$ 0.00460	30	—	0.01427 $\pm$ 0.00004	30	—
DUAL-SGD	—	—	—	0.01440 $\pm$ 0.00000	100	—
PROS-N-KONS	did not complete	—	—	0.01450 $\pm$ 0.00014	149	884.82
CON-KONS	did not complete	—	—	0.01444 $\pm$ 0.00017	147	889.42
B-KONS	**0.00913** $\pm$ 0.00045	100	60	**0.01302** $\pm$ 0.00006	100	505.36
BATCH	0.00212 $\pm$ 0.00001	—	—	0.01147 $\pm$ 0.00001	—	—

Experiments - binary classification

$\alpha = 1,\ \gamma = 1$						
Algorithm	ijcnn1 $n = 141,691,\ d = 22$			cod-rna $n = 271,617,\ d = 8$		
	accuracy	#SV	time	accuracy	#SV	time
FOGD	9.06 ± 0.05	400	—	10.30 ± 0.10	400	—
NOGD	9.55 ± 0.01	100	—	13.80 ± 2.10	100	—
DUAL-SGD	**8.35** ± 0.20	100	—	**4.83** ± 0.21	100	—
PROS-N-KONS	9.70 ± 0.01	100	211.91	13.95 ± 1.19	38	270.81
CON-KONS	9.64 ± 0.01	101	215.71	18.99 ± 9.47	38	271.85
B-KONS	9.70 ± 0.01	98	206.53	13.99 ± 1.16	38	274.94
BATCH	8.33 ± 0.03	—	—	3.781 ± 0.01	—	—

$\alpha = 0.01,\ \gamma = 0.01$						
Algorithm	ijcnn1 $n = 141,691,\ d = 22$			cod-rna $n = 271,617,\ d = 8$		
	accuracy	#SV	time	accuracy	#SV	time
FOGD	9.06 ± 0.05	400	—	10.30 ± 0.10	400	—
NOGD	9.55 ± 0.01	100	—	13.80 ± 2.10	100	—
DUAL-SGD	8.35 ± 0.20	100	—	4.83 ± 0.21	100	—
PROS-N-KONS	10.73 ± 0.12	436	1003.82	4.91 ± 0.04	111	459.28
CON-KONS	6.23 ± 0.18	432	987.33	5.81 ± 1.96	111	458.90
B-KONS	**4.85** ± 0.08	100	147.22	**4.57** ± 0.05	100	333.57
BATCH	5.61 ± 0.01	—	—	3.61 ± 0.01	—	—

PROS-N-KONS - recap

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS: avoid curse of kernelization, constant per-step cost

PROS-N-KONS - recap

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS: avoid curse of kernelization, constant per-step cost

First approximate method with logarithmic regret

PROS-N-KONS - recap

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS: avoid curse of kernelization, constant per-step cost
First approximate method with logarithmic regret

Future work

PROS-N-KONS - recap

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS: avoid curse of kernelization, constant per-step cost

First approximate method with logarithmic regret

Future work

Restarts really necessary?

PROS-N-KONS - recap

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS: avoid curse of kernelization, constant per-step cost

First approximate method with logarithmic regret

Future work

Restarts really necessary?

Adaptive α and γ?

Conclusions

Goal 1: find a small, provably accurate dictionary in near-linear time

SQUEAK and DISQUEAK
↳match space/accuracy of oracle RLS sampling
linear or sublinear runtime, single-pass

Goal 2: use dictionary to solve down-stream problems efficiently

PROS-N-KONS
↳preserve logarithmic rate with constant per-step cost

Leverage existing analysis to get provably accurate linear-time algorithms

Open questions

Short-term: more applications, more experiments

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Anytime KORS, adaptive tree SQUEAK

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Anytime KORS, adaptive tree SQUEAK

From full (gradient descent) to partial feedback (linear/GP bandits)

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Anytime KORS, adaptive tree SQUEAK

From full (gradient descent) to partial feedback (linear/GP bandits)

From RLS to volume sampling/DPP

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Anytime KORS, adaptive tree SQUEAK

From full (gradient descent) to partial feedback (linear/GP bandits)

From RLS to volume sampling/DPP

Long-term: new problems

Open questions

Short-term: more applications, more experiments

Kernel Ridge Regression - Gaussian Process - Laplacian Smoothing

Kernel PCA - Graph Spectral Embedding

Empirically: which kernel/γ for which dataset/α

Middle-term: non-trivial extensions

Anytime KORS, adaptive tree SQUEAK

From full (gradient descent) to partial feedback (linear/GP bandits)

From RLS to volume sampling/DPP

Long-term: new problems

Deterministic algorithms [Ghashami et al., 2015]

Bibliography I

- Alaoui, Ahmed El and Michael W. Mahoney (2015). "Fast randomized kernel methods with statistical guarantees". In: Neural Information Processing Systems (cited on pages 10–12, 23, 54–57, 87–93, 116, 117).
- Bach, Francis (2013). "Sharp analysis of low-rank kernel matrix approximations". In: Conference on Learning Theory (cited on pages 10–12, 23, 87–93, 116, 117).
- Calandriello, Daniele, Alessandro Lazaric, and Michal Valko (2015). "Large-scale semi-supervised learning with online spectral graph sparsification". In: Resource-Efficient Machine Learning workshop at International Conference on (cited on pages 23, 117).
- – (2016). "Analysis of Kelner and Levin graph sparsification algorithm for a streaming setting". In: arXiv preprint arXiv:1609.03769 (cited on pages 23, 117).
- – (2017a). "Distributed Sequential Sampling for Kernel Matrix Approximation". In: AISTATS (cited on pages 13–17, 60–64, 81–93).

Bibliography II

Calandriello, Daniele, Alessandro Lazaric, and Michal Valko (2017b). "Efficient Second-Order Online Kernel Learning with Adaptive Embedding". In: Advances in Neural Information Processing Systems (cited on pages 19–22, 119).

– (2017c). "Second-Order Kernel Online Convex Optimization with Adaptive Sketching". In: International Conference on Machine Learning (cited on pages 13–17, 19–22, 87–93, 119, 132–136, 140–146).

Ghashami, Mina, Edo Liberty, Jeff M. Phillips, and David P. Woodruff (Jan. 7, 2015). "Frequent Directions : Simple and Deterministic Matrix Sketching". In: arXiv:1501.01711 [cs]. arXiv: 1501.01711. URL: http://arxiv.org/abs/1501.01711 (visited on 10/23/2015) (cited on pages 180–187).

Hazan, Elad, Adam Kalai, Satyen Kale, and Amit Agarwal (2006). "Logarithmic regret algorithms for online convex optimization". In: Conference on Learning Theory. Springer, pages 499–513 (cited on pages 127–130, 132–136).

Bibliography III

Hazan, Elad, Alexander Rakhlin, and Peter L Bartlett (2008). "Adaptive online gradient descent". In: Advances in Neural Information Processing Systems, pages 65–72 (cited on pages 125, 126).

Kivinen, J., A.J. Smola, and R.C. Williamson (Aug. 2004). "Online Learning with Kernels". en. In: IEEE Transactions on Signal Processing 52.8. (Visited on 02/18/2017) (cited on pages 124–126).

Luo, Haipeng, Alekh Agarwal, Nicolo Cesa-Bianchi, and John Langford (2016). "Efficient second-order online learning via sketching". In: Neural Information Processing Systems (cited on pages 132–136, 147–155, 163–168).

Musco, Cameron and Christopher Musco (2017). "Recursive Sampling for the Nyström Method". In: Advances in Neural Information Processing Systems (cited on pages 23, 87–93, 116, 117).

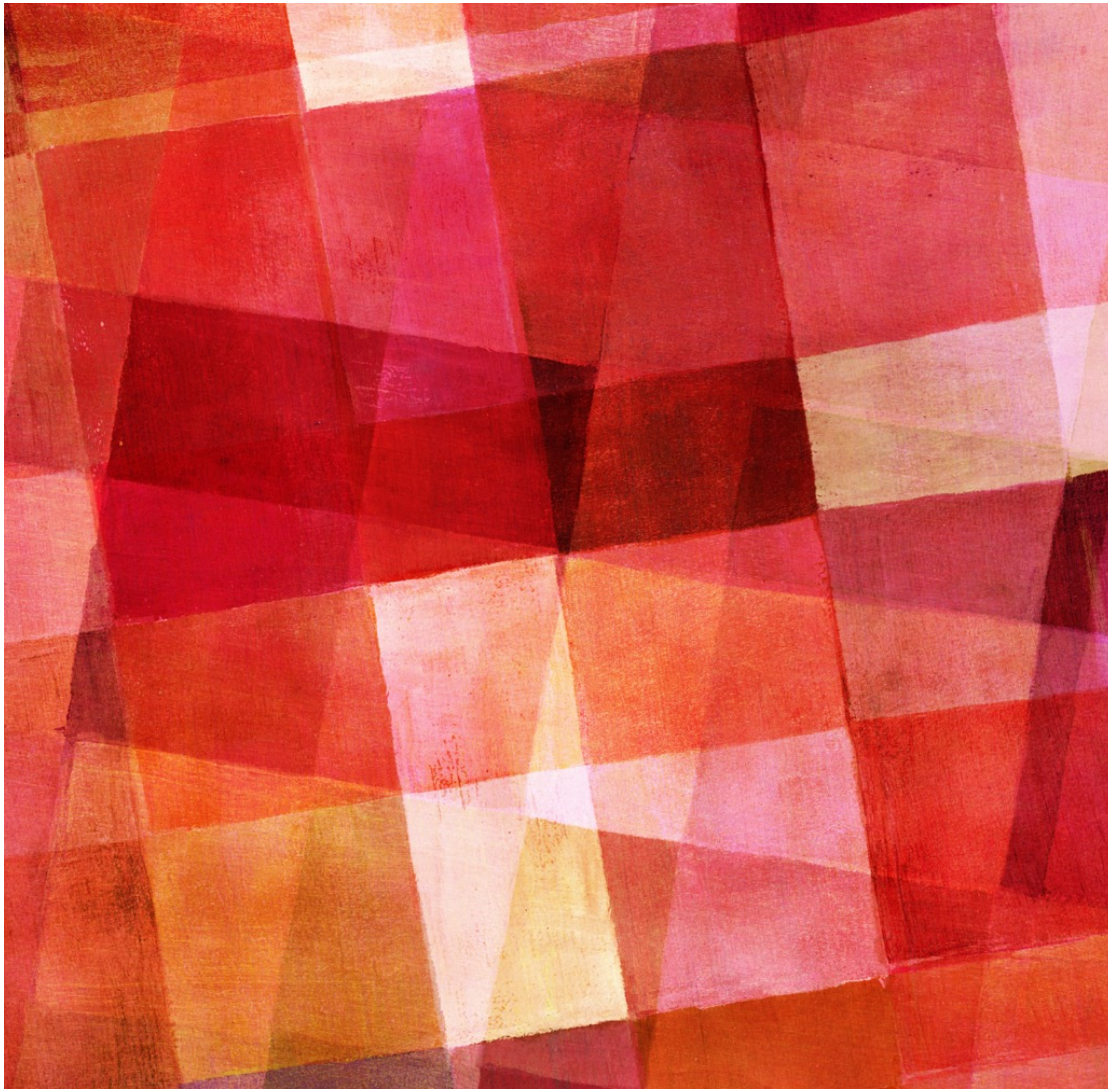
Bibliography IV

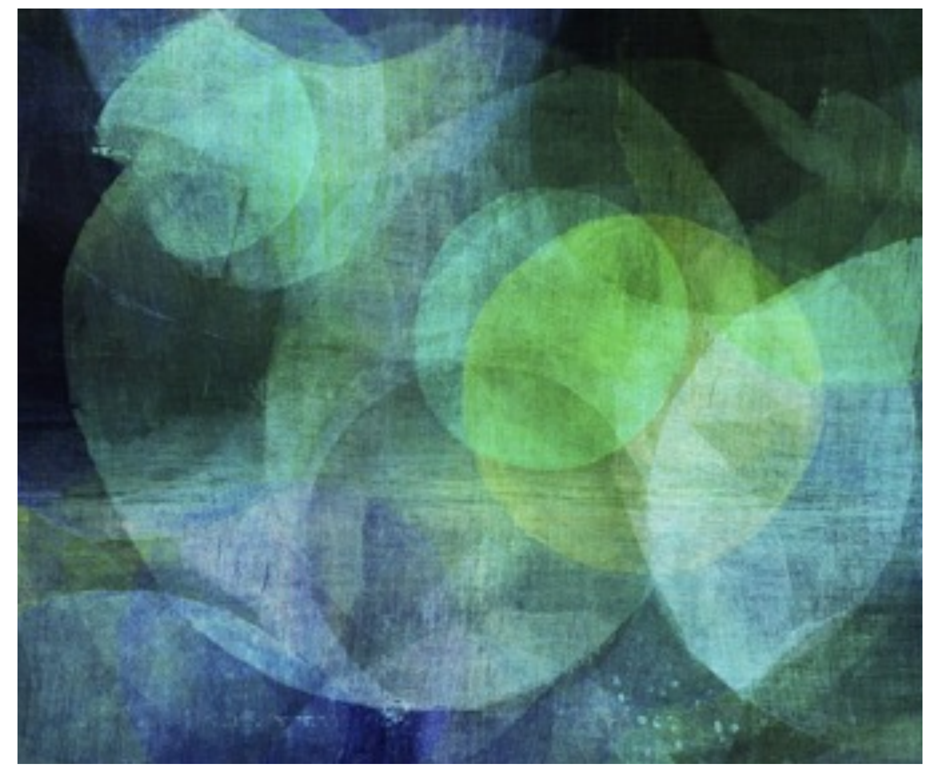
Rudi, Alessandro, Raffaello Camoriano, and Lorenzo Rosasco (2015). "Less is more: Nystrom computational regularization". In: Neural Information Processing Systems (cited on pages 23, 116, 117).

Rudi, Alessandro, Luigi Carratino, and Lorenzo Rosasco (2017). "FALKON: An Optimal Large Scale Kernel Method". In: Advances in Neural Information Processing Systems (cited on page 119).

Zhdanov, Fedor and Yuri Kalnishkan (Oct. 2010). "An Identity for Kernel Ridge Regression". en. In: Algorithmic Learning Theory. Edited by Springer. Lecture Notes in Computer Science, pages 405–419 (cited on pages 127–130).

Zinkevich, Martin (2003). "Online Convex Programming and Generalized Infinitesimal Gradient Ascent". In: International Conference on Machine Learning, pages 928–936 (cited on pages 124–126).





Michal Valko, SequeL, Inria Lille - Nord Europe, michal.valko@inria.fr
http://researchers.lille.inria.fr/~valko/hp/

Reconstruction guarantees

Consider the regularized projection $\mathbf{\Gamma}_n$

$$\mathbf{\Gamma}_n = \Phi_n\Phi_n^{\mathsf{T}}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1} = (\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\Phi_n\Phi_n^{\mathsf{T}}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1}$$
$$= \sum_{i=1}^{n}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\phi_i\phi_i^{\mathsf{T}}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1} = \sum_{i=1}^{n}\psi_i\psi_i^{\mathsf{T}}$$
$$\widetilde{\mathbf{\Gamma}}_n = (\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1}\Phi_n\mathbf{S}_n\mathbf{S}_n^{\mathsf{T}}\Phi_n^{\mathsf{T}}(\Phi_n\Phi_n^{\mathsf{T}} + \gamma\mathbf{I})^{-1} = \sum_{j=1}^{m} w_j\psi_j\psi_j^{\mathsf{T}}$$

An accurate dictionary satisfies

$$\|\mathbf{\Gamma}_n - \widetilde{\mathbf{\Gamma}}_n\|_2^2 \leq \varepsilon$$

equivalent to mixed additive/multiplicative error in quadratic form

$$(1-\varepsilon)\Phi_n\Phi_n^{\mathsf{T}} - \varepsilon\gamma\mathbf{I} \preceq \Phi_n\mathbf{S}_n\mathbf{S}_n^{\mathsf{T}}\Phi_n^{\mathsf{T}} \preceq (1+\varepsilon)\Phi_n\Phi_n^{\mathsf{T}} + \varepsilon\gamma\mathbf{I}$$