# Sparsifying sums of positive semidefinite matrices

**Cristiane Sato**
Joint work with **Nick Harvey** [1] and **Marcel Silva** [2]

Federal University of the ABC Region, Brazil
Center of Mathematics, Computing and Cognition

[1]University of British Columbia
[2]University of São Paulo

# Cut Sparsifiers

## Theorem (Karger '94)

- *weighted graph* $G = (V, E, w)$ *where* $w : E \to \mathbb{R}_+$,
- $\varepsilon > 0$ *small*

*There exists a subgraph* $H = (V, F, y)$ *of* $G$ *and* $y : F \to \mathbb{R}_+$ *s.t.*

$$|F| = O(n \ln n/\varepsilon^2)$$

- *The weight of every cut is approximately preserved*

# Cut Sparsifiers

## Theorem (Karger '94)

- *weighted graph* $G = (V, E, w)$ *where* $w : E \to \mathbb{R}_+$,
- $\varepsilon > 0$ *small*

*There exists a subgraph* $H = (V, F, y)$ *of* $G$ *and* $y : F \to \mathbb{R}_+$ *s.t.*

$$|F| = O(n \ln n/\varepsilon^2)$$

- *The weight of every cut is approximately preserved*
- *That is,*

$$w(\delta_G(S)) = (1 \pm \varepsilon) y(\delta_H(S)), \qquad \forall S \subseteq V$$

# Cut Sparsifiers

## Theorem (Karger '94)

- *weighted graph* $G = (V, E, w)$ *where* $w : E \to \mathbb{R}_+$,
- $\varepsilon > 0$ *small*

*There exists a subgraph* $H = (V, F, y)$ *of* $G$ *and* $y : F \to \mathbb{R}_+$ *s.t.*

$$|F| = O(n \ln n/\varepsilon^2)$$

- *The weight of every cut is approximately preserved*
- *That is,*

$$w(\delta_G(S)) = (1 \pm \varepsilon) y(\delta_H(S)), \qquad \forall S \subseteq V$$

- Application: Faster algorithms by preprocessing the graph

# Cut Sparsifiers

## Theorem (Karger '94)

- *weighted graph* $G = (V, E, w)$ *where* $w : E \to \mathbb{R}_+$,
- $\varepsilon > 0$ *small*

*There exists a subgraph* $H = (V, F, y)$ *of* $G$ *and* $y : F \to \mathbb{R}_+$ *s.t.*

$$|F| = O(n \ln n/\varepsilon^2)$$

- *The weight of every cut is approximately preserved*
- *That is,*
$$w(\delta_G(S)) = (1 \pm \varepsilon) y(\delta_H(S)), \qquad \forall S \subseteq V$$

- Application: Faster algorithms by preprocessing the graph
- How sparse can $H$ be?
- Can we build $H$ efficiently?

# Weighted Laplacians

- $G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$
- Laplacian of $G$ is the $V \times V$ matrix $\mathrm{Lapl}_G$ s.t.
  $\mathrm{Lapl}_G(i, i) =$ degree of $i$
  $\mathrm{Lapl}_G(i, j) = -w_{i,j}$ if $ij \in E$

# Weighted Laplacians

- $G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$
- Laplacian of $G$ is the $V \times V$ matrix $\mathrm{Lapl}_G$ s.t.
  $\mathrm{Lapl}_G(i, i) =$ degree of $i$
  $\mathrm{Lapl}_G(i, j) = -w_{i,j}$ if $ij \in E$
- $\mathrm{Lapl}_G$ is positive semidefinite
  All eigenvalues are $\geq 0$
  Notation: $\mathrm{Lapl}_G \succeq 0$

# Spectral sparsifiers

## Theorem (Spielman, Teng '04)

- $G = (V, E, w)$ *a weighted graph, where* $w : E \to \mathbb{R}_+$
- $\varepsilon > 0$ *small*

*There are new weights* $y : E \to \mathbb{R}_+$ *s.t.*

- *has* $n \operatorname{polylog}(n)/\varepsilon^2$ *nonzero entries*
- $H := (V, E, y)$ *satisfies*

$$\mathrm{Lapl}_G \preceq \mathrm{Lapl}_H \preceq (1+\varepsilon)\,\mathrm{Lapl}_G$$

# Spectral sparsifiers

## Theorem (Spielman, Teng '04)

- *$G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$*
- *$\varepsilon > 0$ small*

*There are new weights $y : E \to \mathbb{R}_+$ s.t.*

- *has $n \operatorname{polylog}(n)/\varepsilon^2$ nonzero entries*
- *$H := (V, E, y)$ satisfies*

$$\mathrm{Lapl}_G \preceq \mathrm{Lapl}_H \preceq (1+\varepsilon)\,\mathrm{Lapl}_G$$

Notation: $A \preceq B \iff B - A$ is positive semidefinite

- nearly-linear time solvers for symmetric, diagonally-dominant linear systems (Spielman and Teng + Koutis, Miller, and Peng, 2004)

# Spectral sparsifiers

## Theorem (Spielman, Teng '04)

- *$G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$*
- *$\varepsilon > 0$ small*

*There are new weights $y : E \to \mathbb{R}_+$ s.t.*

- *has $n \operatorname{polylog}(n)/\varepsilon^2$ nonzero entries*
- *$H := (V, E, y)$ satisfies*

$$\mathrm{Lapl}_G \preceq \mathrm{Lapl}_H \preceq (1+\varepsilon)\,\mathrm{Lapl}_G$$

*$y$ may be found in $\tilde{O}(m)$ time*

Notation: $A \preceq B \iff B - A$ is positive semidefinite

- nearly-linear time solvers for symmetric, diagonally-dominant linear systems (Spielman and Teng + Koutis, Miller, and Peng, 2004)

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

- $h^{\mathsf{T}} \operatorname{Lapl}_G h = \sum_{ij \in E} x_{ij} (h_i - h_j)^2$

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

- $h^\mathsf{T} \operatorname{Lapl}_G h = \sum_{ij \in E} x_{ij}(h_i - h_j)^2$
- $h^\mathsf{T} \operatorname{Lapl}_G h$ is the $w$-weight of the cut $\delta(S)$, i.e., $w(\delta(S))$

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

- $h^{\mathsf{T}} \operatorname{Lapl}_G h = \sum_{ij \in E} x_{ij}(h_i - h_j)^2$
- $h^{\mathsf{T}} \operatorname{Lapl}_G h$ is the $w$-weight of the cut $\delta(S)$, i.e., $w(\delta(S))$
- $h^{\mathsf{T}} \operatorname{Lapl}_H h$ is the $y$-weight of the cut $\delta(S)$, i.e., $y(\delta(S))$

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

- $h^{\mathsf{T}} \operatorname{Lapl}_G h = \sum_{ij \in E} x_{ij}(h_i - h_j)^2$
- $h^{\mathsf{T}} \operatorname{Lapl}_G h$ is the $w$-weight of the cut $\delta(S)$, i.e., $w(\delta(S))$
- $h^{\mathsf{T}} \operatorname{Lapl}_H h$ is the $y$-weight of the cut $\delta(S)$, i.e., $y(\delta(S))$
- $\operatorname{Lapl}_H \succeq \operatorname{Lapl}_G$ implies

$$h^{\mathsf{T}}(\operatorname{Lapl}_H - \operatorname{Lapl}_G)h \geq 0$$

That is,

$$h^{\mathsf{T}} \operatorname{Lapl}_H h \geq h^{\mathsf{T}} \operatorname{Lapl}_G h$$

# Spectral sparsifiers are cut sparsifiers

- $h$ the incidence vector of $S \subseteq V$,

$$h_v = 1 \text{ if } v \in S$$
$$0, \text{ otherwise}$$

- $h^{\mathsf{T}} \operatorname{Lapl}_G h = \sum_{ij \in E} x_{ij}(h_i - h_j)^2$
- $h^{\mathsf{T}} \operatorname{Lapl}_G h$ is the $w$-weight of the cut $\delta(S)$, i.e., $w(\delta(S))$
- $h^{\mathsf{T}} \operatorname{Lapl}_H h$ is the $y$-weight of the cut $\delta(S)$, i.e., $y(\delta(S))$
- $\operatorname{Lapl}_H \succeq \operatorname{Lapl}_G$ implies

$$h^{\mathsf{T}}(\operatorname{Lapl}_H - \operatorname{Lapl}_G)h \geq 0$$

  That is,

$$h^{\mathsf{T}} \operatorname{Lapl}_H h \geq h^{\mathsf{T}} \operatorname{Lapl}_G h$$

- $\operatorname{Lapl}_H \preceq (1+\varepsilon) \operatorname{Lapl}_G$ implies $h^{\mathsf{T}} \operatorname{Lapl}_H h \leq (1+\varepsilon) h^{\mathsf{T}} \operatorname{Lapl}_G h$

# Laplacian matrix as a sum of matrices

- $G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$
- The Laplacian of $G$ is the $V \times V$ matrix

$$\mathrm{Lapl}_G := \sum_{ij \in E} w_{ij} \begin{array}{c} \\ i \\ j \\ \\ \end{array} \begin{pmatrix} & i & j & \\ & 1 & -1 & \\ & -1 & 1 & \\ & & & \end{pmatrix}$$

# Laplacian matrix as a sum of matrices

- $G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$
- The Laplacian of $G$ is the $V \times V$ matrix

$$\mathrm{Lapl}_G := \sum_{ij \in E} w_{ij} \begin{matrix} i \\ \\ j \\ \\ \end{matrix} \begin{pmatrix} \\ 1 \\ \\ -1 \\ \\ \end{pmatrix} \begin{matrix} & i & j & \\ ( & 1 & -1 & \quad ) \end{matrix}$$

# Laplacian matrix as a sum of matrices

- $G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$
- The Laplacian of $G$ is the $V \times V$ matrix

$$\mathrm{Lapl}_G := \sum_{ij \in E} w_{ij} \begin{matrix} \\ i \\ \\ j \\ \\ \end{matrix} \begin{pmatrix} \\ 1 \\ \\ -1 \\ \\ \end{pmatrix} \begin{matrix} & i & j & \\ ( & 1 & -1 & \quad ) \end{matrix}$$

- $\mathrm{Lapl}_G$ is a sum of rank-one positive semidefinite matrices

# Sparsifiers of Sums of Rank-One PSD Matrices

## Theorem (Batson, Spielman, Srivastava '09)

- *$B_1, \ldots, B_m$ p.s.d. $n \times n$ matrices of rank one*
- *$B := \sum_i B_i$*
- *$\varepsilon > 0$ small*

*There are new weights $y \in \mathbb{R}_+^m$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- *$B \preceq \sum_i y_i B_i \preceq (1+\varepsilon)B$*

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Sparsifiers of Sums of Rank-One PSD Matrices

## Theorem (Batson, Spielman, Srivastava '09)

- $B_1, \ldots, B_m$ *p.s.d.* $n \times n$ *matrices of rank one*
- $B := \sum_i B_i$
- $\varepsilon > 0$ *small*

*There are new weights* $y \in \mathbb{R}_+^m$ *s.t.*

- $y$ *has* $O(n/\varepsilon^2)$ *nonzero entries*
- $B \preceq \sum_i y_i B_i \preceq (1+\varepsilon)B$

$y$ *may be found in* $O(mn^3/\varepsilon^2)$ *time*

- (Lee-Sun'15) Almost-linear time method

# Sparsifiers of Sums of Rank-One PSD Matrices

## Theorem (Batson, Spielman, Srivastava '09)

- *$B_1, \ldots, B_m$ p.s.d. $n \times n$ matrices of rank one*
- $B := \sum_i B_i$
- $\varepsilon > 0$ *small*

*There are new weights $y \in \mathbb{R}_+^m$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- $B \preceq \sum_i y_i B_i \preceq (1+\varepsilon)B$

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Sparsifiers of Sums of PSD Matrices

Theorem (de Carli Silva, Harvey, S., '11)

- *$B_1, \ldots, B_m$ p.s.d. $n \times n$ matrices of any rank*
- *$B := \sum_i B_i$*
- *$\varepsilon > 0$ small*

*There are new weights $y \in \mathbb{R}_+^m$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- *$B \preceq \sum_i y_i B_i \preceq (1+\varepsilon)B$*

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Applications

- spectral sparsifiers of graphs with extra properties
- cut sparsifiers of uniform hypergraphs (specially 3-uniform)
- sparse solutions to semidefinite programs

# Sparsifiers with Costs

## Theorem

- *$G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$*
- *$\varepsilon > 0$ small*

*There are new weights $y : E \to \mathbb{R}_+$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- *the reweighted graph $H := (V, E, y)$ satisfies*

$$\mathrm{Lapl}_G \preceq \mathrm{Lapl}_H \preceq (1+\varepsilon)\,\mathrm{Lapl}_G$$

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Sparsifiers with Costs

## Theorem

- *$G = (V, E, w)$ a weighted graph, where $w : E \to \mathbb{R}_+$*
- *$\varepsilon > 0$ small*
- *"costs" $c : E \to \mathbb{R}_+$*

*There are new weights $y : E \to \mathbb{R}_+$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- *the reweighted graph $H := (V, E, y)$ satisfies*

$$\mathrm{Lapl}_G \preceq \mathrm{Lapl}_H \preceq (1+\varepsilon)\,\mathrm{Lapl}_G$$

- *$c^\mathsf{T} w \leq c^\mathsf{T} y \leq (1+\varepsilon) c^\mathsf{T} w$*

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Add extra info to Laplacian

$$\sum_{ij \in E} w_{ij} \begin{array}{c} \\ i \\ j \end{array} \begin{array}{c} \begin{array}{cc} i & j \end{array} \\ \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \end{array}$$

# Add extra info to Laplacian

$$\sum_{ij \in E} \begin{array}{c} \\ i \\ j \end{array} \begin{array}{c} \begin{array}{cc} i & j \end{array} \\ \begin{pmatrix} w_{ij} & -w_{ij} \\ -w_{ij} & w_{ij} \end{pmatrix} \end{array}$$

# Add extra info to Laplacian

$$\sum_{ij \in E} \begin{array}{c} \\ i \\ j \\ 0 \end{array} \begin{array}{c} \begin{array}{ccc} i & j & 0 \end{array} \\ \begin{pmatrix} w_{ij} & -w_{ij} & \\ -w_{ij} & w_{ij} & \\ & & c_{ij} \end{pmatrix} \end{array}$$

# Cut Sparsifiers of 3-Uniform Hypergraphs

## Theorem

- *$\mathcal{G} = (V, \mathcal{E}, w)$ a weighted 3-uniform hypergraph, where $w : \mathcal{E} \to \mathbb{R}_+$*
- *i.e., $\mathcal{E} \subseteq \binom{V}{3}$*
- *$\varepsilon > 0$ small*

*There are new weights $y : \mathcal{E} \to \mathbb{R}_+$ s.t.*

- *$y$ has $O(n/\varepsilon^2)$ nonzero entries*
- *the reweighted hypergraph $\mathcal{H} := (V, \mathcal{E}, y)$ satisfies*

$$w(\delta_{\mathcal{G}}(S)) \leq y(\delta_{\mathcal{H}}(S)) \leq (1+\varepsilon)w(\delta_{\mathcal{G}}(S)) \qquad \forall S \subseteq V$$

*$y$ may be found in $O(mn^3/\varepsilon^2)$ time*

# Hypergraph Laplacians

$$\sum_{ijk \in \mathcal{E}} w_{ijk} \begin{array}{c} \\ i \\ j \\ k \end{array} \begin{array}{c} \begin{array}{ccc} i & j & k \end{array} \\ \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix} \end{array}$$

# Semidefinite Programs

## Theorem

- *$B_1, \ldots, B_m$ p.s.d. $n \times n$ matrices*
  *$B$ sym $n \times n$ matrix*
- $c \in \mathbb{R}_+^m$
- *Semidefinite program (SDP)*

$$
\begin{aligned}
\min \quad & c^\mathsf{T} z \\
& \sum_i z_i B_i \succeq B \\
& z \in \mathbb{R}_+^m
\end{aligned}
$$

- *feasible solution $z^*$*
- $\varepsilon \in (0,1)$
  *There exists a feasible solution $\tilde{z}$ with at most $O(n/\varepsilon^2)$ nonzero entries and $c^\mathsf{T}\tilde{z} \leq (1+\varepsilon)c^\mathsf{T}z^*$.*

# Future directions

- Find more applications of the arbitrary-rank sparsification result
- Improve running times

# Future directions

- Find more applications of the arbitrary-rank sparsification result
- Improve running times

- Positive Semidefiniteness Assumption

  For each $n > 0$,

  there exist $B_1, \ldots, B_m$ with $m = \Omega(n^2)$

  and $B := \sum_i B_i$ p.d. such that

  for every $\varepsilon \in (0, 1)$ and $y \in \mathbb{R}^m_+$ with $(1 - \varepsilon)B \preceq \sum_i y_i B_i$,

  every entry of $y$ is nonzero

# Pseudoinverse

We may assume that

$$\sum_{i=1}^{m} B_i = I$$

by applying Moore-Penrose pseudoinverse

# $O(n \log n/\varepsilon^2)$ versions

- Ahlswede–Winter Theorem
- Can be de-randomized using pessimistic estimators

# The approach

1. Start with $A = 0$
2. In each iteration choose a matrix $B_i$ and compute a weight $\alpha$.

Set $A = A + \alpha B_i$

# The approach

1. Start with $A = 0$
2. In each iteration choose a matrix $B_i$ and compute a weight $\alpha$.
Set $A = A + \alpha B_i$

**Batson, Spielman, Srivastava**: $B_i = vv^T$ is a rank-one matrix.
Use Sherman-Morrison formula

$$\mathrm{Tr}(M - \alpha vv^T)^{-1} = \mathrm{Tr}(M^{-1}) + \frac{\alpha v^{\mathsf{T}} M^{-2} v}{1 - v^{\mathsf{T}} M^{-1} v}$$

# The approach

1. Start with $A = 0$
2. In each iteration choose a matrix $B_i$ and compute a weight $\alpha$.
Set $A = A + \alpha B_i$

**Batson, Spielman, Srivastava**: $B_i = vv^T$ is a rank-one matrix.
Use Sherman-Morrison formula

$$\mathrm{Tr}(M - \alpha vv^T)^{-1} = \mathrm{Tr}(M^{-1}) + \frac{\alpha v^\mathsf{T} M^{-2} v}{1 - v^\mathsf{T} M^{-1} v}$$

**Ours:** $B_i = VV^T$ is an arbitrary-rank matrix.
Sherman-Morrison-Woodbury formula:

$$\mathrm{Tr}(M - \alpha VV^T)^{-1} = \mathrm{Tr}\left(M^{-1}\right) + \mathrm{Tr}\left(\alpha M^{-1} V (I - \alpha V^\mathsf{T} M^{-1} V)^{-1} V^T M^{-1}\right)$$

# Upper barrier

- $u$ (upper bound for eigenvalues)
- Barrier function

$$\Phi^u(A) = \sum_{i=1}^{n} \frac{1}{u - \lambda_i(A)} = \operatorname{Tr}(uI - A)^{-1}$$

# Upper barrier

- $u$ (upper bound for eigenvalues)
- Barrier function

$$\Phi^u(A) = \sum_{i=1}^{n} \frac{1}{u - \lambda_i(A)} = \mathrm{Tr}\,(uI - A)^{-1}$$

- matrices $A$ and $B \succeq 0$, we want to control the $\lambda_{\max}(A + \alpha B)$
- Given $\delta_u > 0$,
  suppose we want $\lambda_{\max}(A + \alpha B) \leq u + \delta_U := u'$.
  What conditions on $\alpha$ guarantee that?

# Upper barrier

- $u$ (upper bound for eigenvalues)
- Barrier function

$$\Phi^u(A) = \sum_{i=1}^{n} \frac{1}{u - \lambda_i(A)} = \mathrm{Tr}\,(uI - A)^{-1}$$

- matrices $A$ and $B \succeq 0$, we want to control the $\lambda_{\max}(A + \alpha B)$
- Given $\delta_u > 0$,
  suppose we want $\lambda_{\max}(A + \alpha B) \leq u + \delta_U := u'$.
  What conditions on $\alpha$ guarantee that?
- We have

$$1/\alpha \geq U_A(B) \qquad \text{implies} \qquad \Phi^{u'}(A + \alpha B) \leq \Phi^u(A).$$

and $\lambda_{\max}(A + \alpha B) < u'$
where $M := u'I - A$

$$U_A(B) := \frac{\langle M^{-2}, B \rangle}{\Phi^u(A) - \Phi^{u'}(A)} + \langle M^{-1}, B \rangle$$

# Lower barrier

- $\ell$ (lower bound for eigenvalues)
- Barrier function

$$\Phi_\ell(A) = \sum_{i=1}^{n} \frac{1}{\lambda_i(A) - \ell} = \mathrm{Tr}\,(A - \ell I)^{-1}$$

- Given $\delta_\ell > 0$,
  suppose we want $\lambda_{\min}(A + \alpha B) \geq \ell + \delta_\ell := \ell'$.
  What conditions on $\alpha$ guarantee that?
- We have

$$1/\alpha \leq L_A(B) \qquad \text{implies} \qquad \Phi_{\ell'}(A + \alpha B) \leq \Phi_\ell(A).$$

and $\lambda_{\min}(A + \alpha B) > \ell'$
where $N := A - \ell' I$

$$L_A(B) := \frac{\langle N^{-2}, B\rangle}{\Phi_{\ell'}(A) - \Phi_\ell(A)} + \langle N^{-1}, B\rangle$$

# Overview

- $A(0) := 0$ and $y(0) := 0$.
  Parameters $u_0, \ell_0, \delta_L, \delta_U$ to be chosen
  $T := 4n/\varepsilon^2$.
  Define the barrier functions $\Phi^{u_0}(A)$ and $\Phi_{\ell_0}(A)$.

# Overview

- $A(0) := 0$ and $y(0) := 0$.
  Parameters $u_0, \ell_0, \delta_L, \delta_U$ to be chosen
  $T := 4n/\varepsilon^2$.
  Define the barrier functions $\Phi^{u_0}(A)$ and $\Phi_{\ell_0}(A)$.
- For $t = 1, \ldots, T$
  - $u_t := u_{t-1} + \delta_U$ and $\ell_t := \ell_{t-1} + \delta_L$.
  - Find a matrix $B_j$ and a value $\alpha > 0$ such that

    $$\Phi^{u_t}(A(t-1) + \alpha B_j) \leq \Phi^{u_{t-1}}(A(t-1))$$

    $$\Phi_{\ell_t}(A(t-1) + \alpha B_j) \leq \Phi_{\ell_{t-1}}(A(t-1)).$$

    and so $A(t-1) + \alpha B_j \in [\ell_t, u_t]$
  - $A(t) := A(t-1) + \alpha B_j$
    $y(t) := y(t-1) + \alpha e_j$.

# End of the algorithm

$$\frac{\lambda_{\max}(A(T))}{\lambda_{\min}(A(T))} \le \frac{u_0 + \delta_U T}{\ell_0 + \delta_L T} \le \frac{1+\varepsilon}{1-\varepsilon}$$

with $T = 4n/\varepsilon^2$

$$\delta_L := 1 \qquad \varepsilon_L := \frac{\varepsilon}{2} \qquad \ell_0 := -\frac{n}{\varepsilon_L}$$

$$\delta_U := \frac{2+\varepsilon}{2-\varepsilon} \qquad \varepsilon_U := \frac{\varepsilon}{2\delta_U} \qquad u_0 := \frac{n}{\varepsilon_U}.$$

# Overview

- $A(0) := 0$ and $y(0) := 0$.
  Parameters $u_0, \ell_0, \delta_L, \delta_U$ to be chosen
  $T := 4n/\varepsilon^2$.
  Define the barrier functions $\Phi^{u_0}(A)$ and $\Phi_{\ell_0}(A)$.
- For $t = 1, \ldots, T$
  - $u_t := u_{t-1} + \delta_U$ and $\ell_t := \ell_{t-1} + \delta_L$.
  - Find a matrix $B_j$ and a value $\alpha > 0$ such that

$$\Phi^{u_t}(A(t-1) + \alpha B_j) \leq \Phi^{u_{t-1}}(A(t-1))$$

$$\Phi_{\ell_t}(A(t-1) + \alpha B_j) \leq \Phi_{\ell_{t-1}}(A(t-1)).$$

    and so $A(t-1) + \alpha B_j \in [\ell_t, u_t]$
  - $A(t) := A(t-1) + \alpha B_j$
    $y(t) := y(t-1) + \alpha e_j$.

# Satisfying both barriers at the same time

- Averaging argument

$$\sum_i U_A(B_i) \leq 1/\delta_U + \varepsilon_U < 1/\delta_L + \varepsilon_L \leq \sum_i L_A(B_i)$$

# Satisfying both barriers at the same time

- Averaging argument

$$\sum_i U_A(B_i) \leq 1/\delta_U + \varepsilon_U < 1/\delta_L + \varepsilon_L \leq \sum_i L_A(B_i)$$

- $\exists\ i$ s.t. $U_A(B_i) \leq L_A(B_i)$
- We can choose $\alpha$ such that

$$U_A(B_i) \leq 1/\alpha \leq L_A(B_i)$$

as needed for the algorithm

# Satisfying both barriers at the same time

- Averaging argument

$$\sum_i U_A(B_i) \leq 1/\delta_U + \varepsilon_U < 1/\delta_L + \varepsilon_L \leq \sum_i L_A(B_i)$$

- $\exists\ i$ s.t. $U_A(B_i) \leq L_A(B_i)$
- We can choose $\alpha$ such that

$$U_A(B_i) \leq 1/\alpha \leq L_A(B_i)$$

  as needed for the algorithm
- Compute $U_A(B_i)$ and $L_A(B_i)$