September 4, 2024

A low-rank high-precision solver for semidefinite programming

Joint work with Daniel Brosch and Angelika Wiegele

FWF Der Wissenschaftsfonds.

UNIVERSITÄT KLAGENFURT

Modeling - Analysis - Optimization DOC.FUNDS DOCTORAL SCHOOL

Semidefinite programming

SDP in standard form

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{SDP}$$

- $C, A_1, \ldots, A_m \in \mathcal{S}_n,\ b \in \mathbb{R}^m$, linear operator $\mathcal{A}\colon \mathcal{S}_n \to \mathbb{R}^m$

$$\mathcal{A}(X) = \begin{pmatrix} \langle A_1, X \rangle \\ \vdots \\ \langle A_m, X \rangle \end{pmatrix}$$

Semidefinite programming

SDP in standard form

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{SDP}$$

- $C, A_1, \ldots, A_m \in \mathcal{S}_n,\ b \in \mathbb{R}^m$, linear operator $\mathcal{A}\colon \mathcal{S}_n \to \mathbb{R}^m$

$$\mathcal{A}(X) = \begin{pmatrix} \langle A_1, X \rangle \\ \vdots \\ \langle A_m, X \rangle \end{pmatrix}$$

$$\begin{aligned} \max \quad & b^\top y \\ \text{s.t.} \quad & C - \mathcal{A}^\top(y) = Z \\ & y \in \mathbb{R}^m,\ Z \in \mathcal{S}_n^+ \end{aligned} \tag{DSDP}$$

- adjoint operator $\mathcal{A}^\top\colon \mathbb{R}^m \to \mathcal{S}_n$ with $\mathcal{A}^\top(y) = \sum_{i=1}^m y_i A_i$

Semidefinite programming

SDP in standard form

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{SDP}$$

- $C, A_1, \ldots, A_m \in \mathcal{S}_n,\ b \in \mathbb{R}^m$, linear operator $\mathcal{A} \colon \mathcal{S}_n \to \mathbb{R}^m$

$$\mathcal{A}(X) = \begin{pmatrix} \langle A_1, X \rangle \\ \vdots \\ \langle A_m, X \rangle \end{pmatrix}$$

$$\begin{aligned} \max \quad & b^\top y \\ \text{s. t.} \quad & C - \mathcal{A}^\top(y) = Z \\ & y \in \mathbb{R}^m,\ Z \in \mathcal{S}_n^+ \end{aligned} \tag{DSDP}$$

- adjoint operator $\mathcal{A}^\top \colon \mathbb{R}^m \to \mathcal{S}_n$ with $\mathcal{A}^\top(y) = \sum_{i=1}^m y_i A_i$
- **assumption:** strong duality holds

Motivation

Fixed-precision solvers:

- interior-point methods, ADMMs, eigenvalue optimization, . . .
- double-precision floating-point arithmetic

Motivation

Fixed-precision solvers:

- interior-point methods, ADMMs, eigenvalue optimization, . . .
- double-precision floating-point arithmetic

Issue:

- There are numerically unstable SDPs with a large range of eigenvalues in their solutions.

Motivation

Fixed-precision solvers:

- interior-point methods, ADMMs, eigenvalue optimization, ...
- double-precision floating-point arithmetic

Issue:

- There are numerically unstable SDPs with a large range of eigenvalues in their solutions.

Arbitrary-precision solvers:

- `SDPA-GMP`
- `Clarabel`
- `COSMO`
- `ClusteredLowRankSolver`
- `Hypatia`

- much slower than fixed-precision solvers

Motivation

Fixed-precision solvers:

- interior-point methods, ADMMs, eigenvalue optimization, ...
- double-precision floating-point arithmetic

Issue:

- There are numerically unstable SDPs with a large range of eigenvalues in their solutions.

Arbitrary-precision solvers:

- `SDPA-GMP`
- `Clarabel`
- `COSMO`
- `ClusteredLowRankSolver`
- `Hypatia`

- much slower than fixed-precision solvers

Our contribution:

- new arbitrary-precision solver implemented in Julia

Motivation

Fixed-precision solvers:

- interior-point methods, ADMMs, eigenvalue optimization, ...
- double-precision floating-point arithmetic

Issue:

- There are numerically unstable SDPs with a large range of eigenvalues in their solutions.

Arbitrary-precision solvers:

- `SDPA-GMP`
- `Clarabel`
- `COSMO`
- `ClusteredLowRankSolver`
- `Hypatia`

- much slower than fixed-precision solvers

Our contribution:

- new arbitrary-precision solver implemented in Julia
- project name: `The Augmented Mixing Method`

Main idea

Burer & Monteiro (2003): `SDPLR` (SDPs in standard form)

Main idea

Burer & Monteiro (2003): `SDPLR` (SDPs in standard form)

- reformulate (SDP) using a matrix factorization
- augmented Lagrangian approach
- subproblems with at most $\approx n\sqrt{2m}$ variables

Main idea

Burer & Monteiro (2003): `SDPLR` (SDPs in standard form)

- reformulate (SDP) using a matrix factorization
- augmented Lagrangian approach
- subproblems with at most $\approx n\sqrt{2m}$ variables

Wang et al. (2018): `Mixing Method` (Max-Cut relaxation)

- Burer-Monteiro factorization
- coordinate descent approach
- small subproblems with analytic solution

Main idea

Burer & Monteiro (2003): `SDPLR` (SDPs in standard form)

- reformulate (SDP) using a matrix factorization
- augmented Lagrangian approach
- subproblems with at most $\approx n\sqrt{2m}$ variables

Wang et al. (2018): `Mixing Method` (Max-Cut relaxation)

- Burer-Monteiro factorization
- coordinate descent approach
- small subproblems with analytic solution

Idea: combine both approaches (Augmented Mixing Method)

- tackle general SDPs
- solve small subproblems with high precision
- ADMM-style method

Burer-Monteiro factorization (Burer & Monteiro, 2003)

Theorem (Barvinok, 1995; Pataki, 1998)

If $\bar{X}$ is an extreme point of (SDP), then $\operatorname{rank}(\bar{X}) \leq k_m$, *where* $k_m := \max\{k \in \mathbb{N} \colon k(k+1)/2 \leq m\}$.

Burer-Monteiro factorization (Burer & Monteiro, 2003)

Theorem (Barvinok, 1995; Pataki, 1998)

If $\bar{X}$ is an extreme point of (SDP), then $\text{rank}(\bar{X}) \leq k_m$, *where* $k_m := \max\{k \in \mathbb{N} \colon k(k+1)/2 \leq m\}$.

(SDP) is equivalent to

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+, \; \text{rank}(X) \leq k_m \end{aligned} \qquad \text{(LR-SDP)}$$

Burer-Monteiro factorization (Burer & Monteiro, 2003)

Theorem (Barvinok, 1995; Pataki, 1998)

If $\bar{X}$ is an extreme point of (SDP), then $\operatorname{rank}(\bar{X}) \leq k_m$*, where* $k_m := \max\{k \in \mathbb{N} \colon k(k+1)/2 \leq m\}$.

(SDP) is equivalent to

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s. t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+, \ \operatorname{rank}(X) \leq k_m \end{aligned} \tag{LR-SDP}$$

Change of variables: $X = V^\top V$

$$\{X \colon X \in \mathcal{S}_n^+, \ \operatorname{rank}(X) \leq k\} = \{V^\top V \colon V \in \mathbb{R}^{k \times n}\}$$

Burer-Monteiro factorization (Burer & Monteiro, 2003)

Theorem (Barvinok, 1995; Pataki, 1998)

If $\bar{X}$ is an extreme point of (SDP), then $\operatorname{rank}(\bar{X}) \leq k_m$, *where* $k_m := \max\{k \in \mathbb{N} \colon k(k+1)/2 \leq m\}$.

(SDP) is equivalent to

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b \\ & X \in \mathcal{S}_n^+, \ \operatorname{rank}(X) \leq k_m \end{aligned} \qquad \text{(LR-SDP)}$$

Change of variables: $X = V^\top V$

$$\{X \colon X \in \mathcal{S}_n^+, \ \operatorname{rank}(X) \leq k\} = \{V^\top V \colon V \in \mathbb{R}^{k \times n}\}$$

- Barvinok-Pataki bound: $k \geq \sqrt{2m}$

Non-convex reformulation

Reformulation

$$\begin{aligned} \min \quad & \langle C, V^\top V \rangle \\ \text{s.t.} \quad & \mathcal{A}(V^\top V) = b \\ & V \in \mathbb{R}^{k \times n} \end{aligned} \tag{$*$}$$

- $(*)$ is a quadratic, nonconvex problem

Non-convex reformulation

Reformulation

$$\begin{aligned} \min \quad & \langle C, V^\top V \rangle \\ \text{s.t.} \quad & \mathcal{A}(V^\top V) = b \\ & V \in \mathbb{R}^{k \times n} \end{aligned} \qquad (*)$$

- $(*)$ is a quadratic, nonconvex problem
- X local minimum $\Rightarrow$ any V with $X = V^\top V$ local minimum

Non-convex reformulation

Reformulation

$$\begin{aligned} \min \quad & \langle C, V^\top V \rangle \\ \text{s.t.} \quad & \mathcal{A}(V^\top V) = b \\ & V \in \mathbb{R}^{k \times n} \end{aligned} \tag{$*$}$$

- $(*)$ is a quadratic, nonconvex problem
- X local minimum $\Rightarrow$ any V with $X = V^\top V$ local minimum

Proposition (Burer & Monteiro, 2005)

Suppose $X = V^\top V$ is feasible for (LR-SDP). Then X is a local minimum of (LR-SDP) if and only if V is a local minimum of $(*)$.

Augmented Lagrangian approach

Augmented Lagrangian

$$\mathcal{L}(V, y; \mu) := \langle C, V^\top V\rangle + \frac{\mu}{2}\|b - \mathcal{A}(V^\top V)\|^2 + \langle y, b - \mathcal{A}(V^\top V)\rangle$$

- penalty parameter $\mu > 0$
- vector of Lagrange multipliers $y \in \mathbb{R}^m$

Augmented Lagrangian approach

Augmented Lagrangian

$$\mathcal{L}(V, y; \mu) := \langle C, V^\top V\rangle + \frac{\mu}{2}\|b - \mathcal{A}(V^\top V)\|^2 + \langle y, b - \mathcal{A}(V^\top V)\rangle$$

- penalty parameter $\mu > 0$
- vector of Lagrange multipliers $y \in \mathbb{R}^m$

Derivative w.r.t V

$$\nabla_V \mathcal{L}(V, y; \mu) = 2V\tilde{S}$$

where

$$\tilde{S} = C - \sum_{i=1}^{m} \tilde{y}_i A_i$$

$$\tilde{y} = y - \mu(\mathcal{A}(V^\top V) - b)$$

Implementation of augmented Lagrangian approach

Algorithm 1 SDPLR (Burer & Monteiro, 2003)

1. Choose starting values V^0, y^0, μ^0, and set $p := 0$.
2. While $\|b - \mathcal{A}(V^{p\top} V^p)\|$ too large:
   - Compute $V^{p+1} := \arg\min_{V \in \mathbb{R}^{k \times n}} \mathcal{L}(V^p, y^p; \mu^p)$.
   - If $\|b - \mathcal{A}(V^{p+1\top} V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}(V^{p+1\top} V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

Implementation of augmented Lagrangian approach

Algorithm 1 SDPLR (Burer & Monteiro, 2003)

1. Choose starting values V^0, y^0, μ^0, and set $p := 0$.
2. While $\|b - \mathcal{A}({V^p}^\top V^p)\|$ too large:
   - Compute $V^{p+1} := \arg\min_{V \in \mathbb{R}^{k \times n}} \mathcal{L}(V^p, y^p; \mu^p)$.
   - If $\|b - \mathcal{A}({V^{p+1}}^\top V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}({V^{p+1}}^\top V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

- primal method in kn variables
- quasi-Newton method used
- no eigenvalue computations, exploits sparsity

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \; i = 1, \ldots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \; i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \qquad \text{(MC-SDP)}$$

The Burer-Monteiro method for (MC-SDP)...

- has no global convergence guarantee **if** $k < \sqrt{2n}$.

  (Waldspurger & Waters, 2020)

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1,\ i = 1, \ldots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

The Burer-Monteiro method for (MC-SDP)...

- has no global convergence guarantee **if** $k < \sqrt{2n}$.
  (Waldspurger & Waters, 2020)
- can fail for (MC-SDP), even **if** $k = n/2$. (O'Carroll et al., 2022)

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \ i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

The Burer-Monteiro method for (MC-SDP). . .

- has no global convergence guarantee **if** $k < \sqrt{2n}$.
  (Waldspurger & Waters, 2020)
- can fail for (MC-SDP), even **if** $k = n/2$. (O'Carroll et al., 2022)
- can get stuck in a spurious local maxima **if** $k \in [\sqrt{2n}, n/2]$.
  (O'Carroll et al., 2022)

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \; i = 1, \ldots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

The Burer-Monteiro method for (MC-SDP)...

- has no global convergence guarantee **if** $k < \sqrt{2n}$.
  (Waldspurger & Waters, 2020)
- can fail for (MC-SDP), even **if** $k = n/2$. (O'Carroll et al., 2022)
- can get stuck in a spurious local maxima **if** $k \in [\sqrt{2n}, n/2]$.
  (O'Carroll et al., 2022)
- is globally convergent **if** $k > n/2$. (Boumal et al., 2018)

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \ i = 1, \ldots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

The Burer-Monteiro method for (MC-SDP)...

- has no global convergence guarantee **if** $k < \sqrt{2n}$.
  (Waldspurger & Waters, 2020)
- can fail for (MC-SDP), even **if** $k = n/2$. (O'Carroll et al., 2022)
- can get stuck in a spurious local maxima **if** $k \in [\sqrt{2n}, n/2]$.
  (O'Carroll et al., 2022)
- is globally convergent **if** $k > n/2$. (Boumal et al., 2018)

However: strong practical performance with smaller values of k

Theoretical results

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \; i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \qquad \text{(MC-SDP)}$$

The Burer-Monteiro method for (MC-SDP)...

- has no global convergence guarantee **if** $k < \sqrt{2n}$. (Waldspurger & Waters, 2020)
- can fail for (MC-SDP), even **if** $k = n/2$. (O'Carroll et al., 2022)
- can get stuck in a spurious local maxima **if** $k \in [\sqrt{2n}, n/2]$. (O'Carroll et al., 2022)
- is globally convergent **if** $k > n/2$. (Boumal et al., 2018)

However: strong practical performance with smaller values of k
Issue: cannot achieve very high accuracy in many cases

Burer-Monteiro factorization for Max-Cut

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s.t.} \quad & X_{ii} = 1, \ \ i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \qquad \text{(MC-SDP)}$$

Burer-Monteiro factorization for Max-Cut

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \; i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \qquad \text{(MC-SDP)}$$

Column-wise storage

$X = V^\top V \succeq 0, \; V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$:

Burer-Monteiro factorization for Max-Cut

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \;\; i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \tag{MC-SDP}$$

Column-wise storage

$X = V^\top V \succeq 0, \;\; V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$:

$$\begin{aligned} \max \quad & \langle C, V^\top V \rangle \\ \text{s. t.} \quad & \|v_i\| = 1, \;\; i = 1, \dots, n \end{aligned} \tag{MC-vec}$$

Burer-Monteiro factorization for Max-Cut

Max-Cut relaxation

$$\begin{aligned} \max \quad & \langle C, X \rangle \\ \text{s. t.} \quad & X_{ii} = 1, \ i = 1, \dots, n \\ & X \in \mathcal{S}_n^+ \end{aligned} \qquad \text{(MC-SDP)}$$

Column-wise storage

$X = V^\top V \succeq 0, \ V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$:

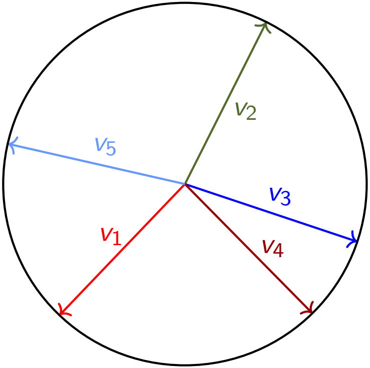
$$\begin{aligned} \max \quad & \langle C, V^\top V \rangle \\ \text{s. t.} \quad & \|v_i\| = 1, \ i = 1, \dots, n \end{aligned} \qquad \text{(MC-vec)}$$

- Barvinok-Pataki bound: (MC-SDP) $\Leftrightarrow$ (MC-vec) for $k \geq \sqrt{2n}$

Geometric interpretation

Optimization problem (MC-vec)

$$\max \quad \langle C, V^\top V \rangle = \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \qquad \text{(MC-vec)}$$
$$\text{s. t.} \quad \|v_i\| = 1, \; i = 1, \ldots, n$$



$$v_i^\top v_j = \|v_i\| \cdot \|v_j\| \cdot \cos \measuredangle(v_i, v_j)$$
$$= \cos \measuredangle(v_i, v_j)$$

Coordinate ascent approach

Optimization problem (MC-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \ldots, n \end{aligned} \tag{MC-vec}$$

Coordinate ascent approach

Optimization problem (MC-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \ i = 1, \ldots, n \end{aligned} \tag{MC-vec}$$

Coordinate ascent

We fix all columns except v_i.

Coordinate ascent approach

Optimization problem (MC-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \ i = 1, \dots, n \end{aligned} \tag{MC-vec}$$

Coordinate ascent

We fix all columns except v_i. (MC-vec) reduces to

$$\begin{aligned} \max \quad & g^\top v_i \\ \text{s. t.} \quad & \|v_i\| = 1, \ v_i \in \mathbb{R}^k \end{aligned}$$

where $g = \sum_{j=1, j \neq i}^{n} C_{ij} v_j = V \cdot C_{(i)} - C_{ii} v_i$.

Coordinate ascent approach

Optimization problem (MC-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \tag{MC-vec}$$

Coordinate ascent

We fix all columns except v_i. (MC-vec) reduces to

$$\begin{aligned} \max \quad & g^\top v_i = \|g\| \cdot \|v_i\| \cdot \cos \measuredangle(g, v_i) \\ \text{s. t.} \quad & \|v_i\| = 1, \; v_i \in \mathbb{R}^k \end{aligned}$$

where $g = \sum_{j=1, j \neq i}^{n} C_{ij} v_j = V \cdot C_{(i)} - C_{ii} v_i$.

Coordinate ascent approach

Optimization problem (MC-vec)

$$\begin{aligned} \max \quad & \sum_{i,j=1}^{n} C_{ij} v_i^\top v_j \\ \text{s. t.} \quad & \|v_i\| = 1, \; i = 1, \dots, n \end{aligned} \tag{MC-vec}$$

Coordinate ascent

We fix all columns except v_i. (MC-vec) reduces to

$$\begin{aligned} \max \quad & g^\top v_i = \|g\| \cdot \|v_i\| \cdot \cos \measuredangle(g, v_i) \\ \text{s. t.} \quad & \|v_i\| = 1, \; v_i \in \mathbb{R}^k \end{aligned}$$

where $g = \sum_{j=1, j \neq i}^{n} C_{ij} v_j = V \cdot C_{(i)} - C_{ii} v_i$.

- closed-form solution: $v_i = \frac{g}{\|g\|}$ if $g \neq 0$

Mixing Method (Wang et al., 2018)

Algorithm 2 Mixing Method (Wang et al., 2018)

Input: $C \in \mathbb{R}^{n \times n}$ with $\operatorname{diag}(C) = 0,\ k \in \mathbb{N}_{\geq 1}$
Output: approximate solution $V = (v_1 | \dots | v_n) \in \mathbb{R}^{k \times n}$ of (SDP-vec)
for $i \leftarrow 1$ **to** n **do**
    $v_i \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$
end
while *not yet converged* **do**
    for $i \leftarrow 1$ **to** n **do**
        $v_i \leftarrow \frac{V \cdot C_{(i)}}{\|V \cdot C_{(i)}\|}$
    end
end

Mixing Method (Wang et al., 2018)

Algorithm 2 Mixing Method (Wang et al., 2018)

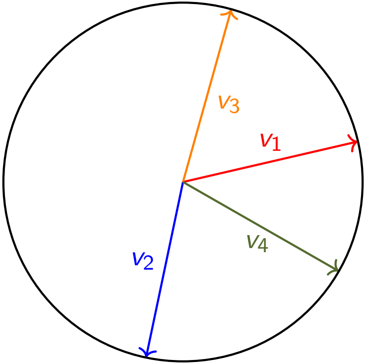
Input: $C \in \mathbb{R}^{n\times n}$ with $\operatorname{diag}(C) = 0,\ k \in \mathbb{N}_{\geq 1}$
Output: approximate solution $V = (v_1|\dots|v_n) \in \mathbb{R}^{k\times n}$ of (SDP-vec)
for $i \leftarrow 1$ **to** n **do**
  $v_i \leftarrow$ random vector on the unit sphere $\mathcal{S}^{k-1}$
end
while *not yet converged* **do**
  for $i \leftarrow 1$ **to** n **do**
    $v_i \leftarrow \frac{V\cdot C_{(i)}}{\|V\cdot C_{(i)}\|}$
  end
end

Theorem: local linear convergence (Wang et al., 2018)

Let $k > \sqrt{2n}$. If the iterates do not degenerate, then the Mixing Method converges locally to the global optimum of (SDP-vec) at a linear rate.

Example with $n = 4$ and $k = 2$

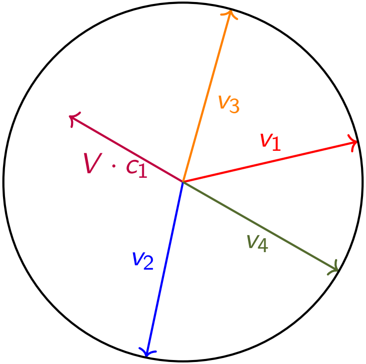
$$C = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = -2.469151715641014$$

Example with $n = 4$ and $k = 2$

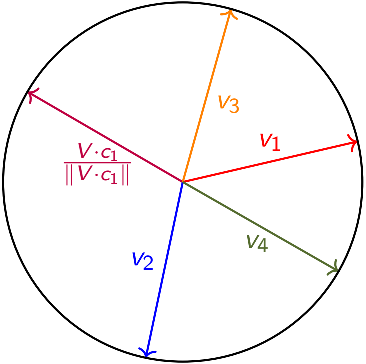
$$C = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = -2.469151715641014$$

Example with $n = 4$ and $k = 2$

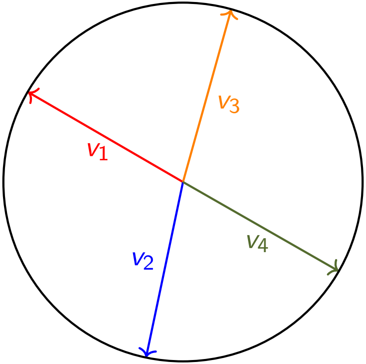
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = -2.469151715641014$$

Example with $n = 4$ and $k = 2$

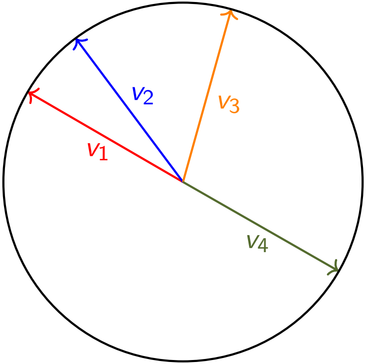
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V\rangle = 0.0701836938398076$$

Example with $n = 4$ and $k = 2$

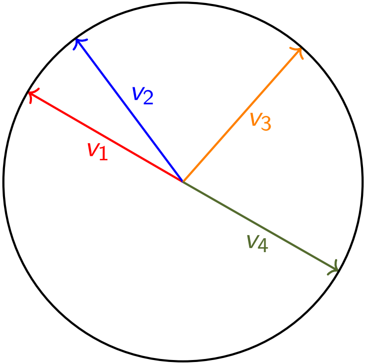
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.1042821481042009$$

Example with $n = 4$ and $k = 2$

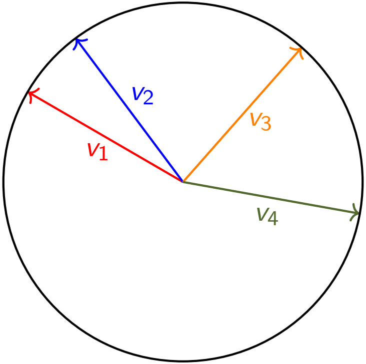
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V\rangle = 2.1248497956082537$$

Example with $n = 4$ and $k = 2$

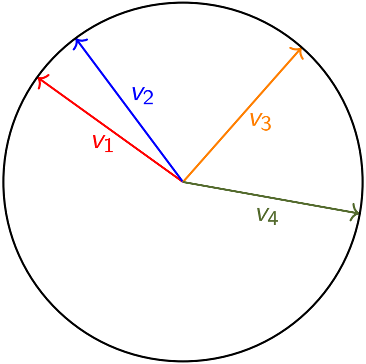
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2584781813631301$$

Example with $n = 4$ and $k = 2$

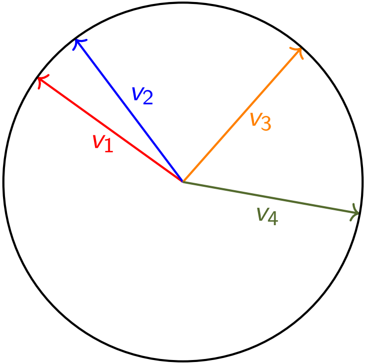
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2669613535505473$$

Example with $n = 4$ and $k = 2$

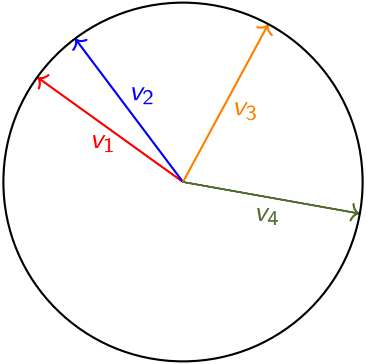
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2669669930002718$$

Example with $n = 4$ and $k = 2$

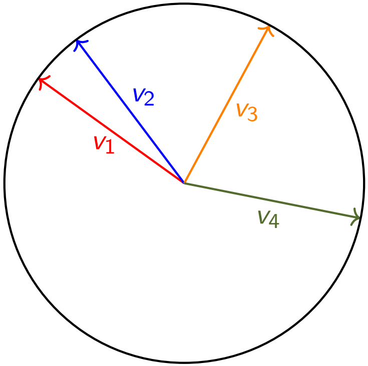
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2820426702215686$$

Example with $n = 4$ and $k = 2$

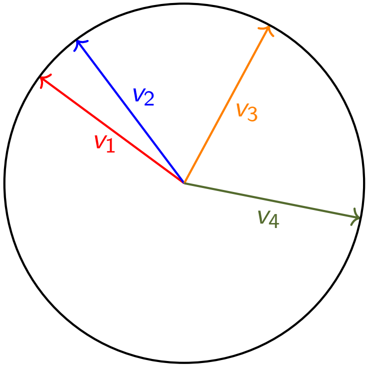
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2824146853764495$$

Example with $n = 4$ and $k = 2$

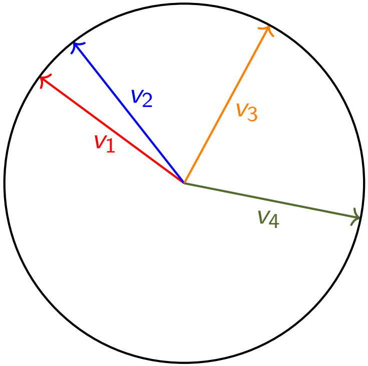
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2825485984904232$$

Example with $n = 4$ and $k = 2$

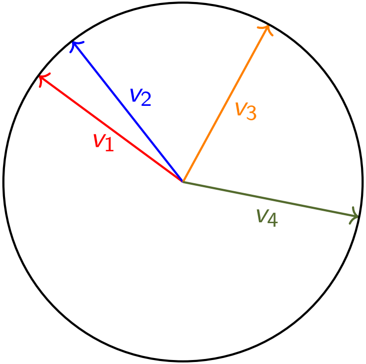
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2827921992397187$$

Example with $n = 4$ and $k = 2$

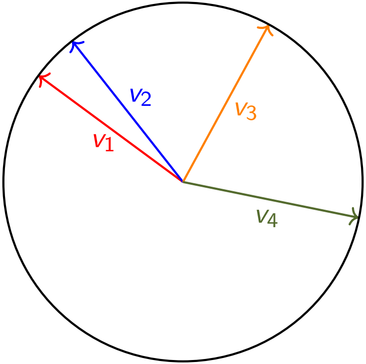
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2827965824488148$$

Example with $n = 4$ and $k = 2$

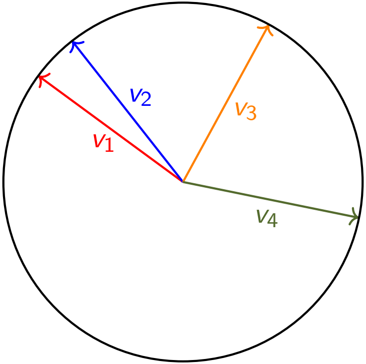
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2828175664597827$$

Example with $n = 4$ and $k = 2$

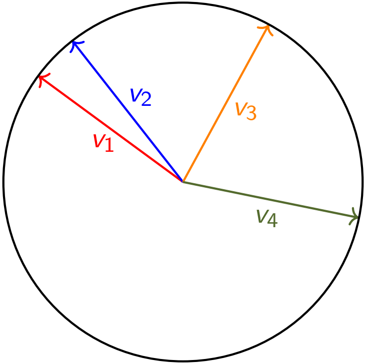
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2828214514872149$$

Example with $n = 4$ and $k = 2$

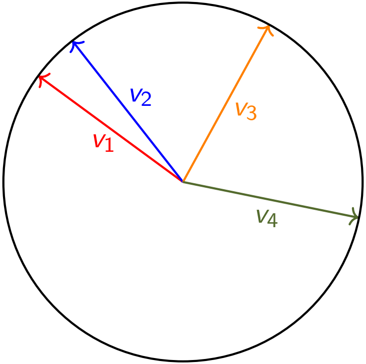
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2828225671023645$$

Example with $n = 4$ and $k = 2$

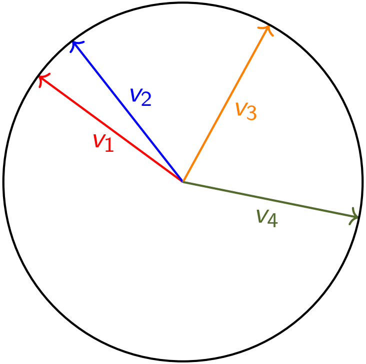
$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V\rangle = 2.2828245614424776$$

Example with $n = 4$ and $k = 2$

$$C = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & -3 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -4 & 2 \\ -3 & -1 & 2 & 2 \end{pmatrix}$$



$$\langle C, V^\top V \rangle = 2.2828250454404815$$

The Augmented Mixing Method

Algorithm 3 `SDPLR`

---

1. Choose starting values V^0, y^0, μ^0, and set $p := 0$.
2. While $\|b - \mathcal{A}({V^p}^\top V^p)\|$ too large:
   - Compute $V^{p+1} := \arg\min_{V \in \mathbb{R}^{k \times n}} \mathcal{L}(V^p, y^p; \mu^p)$.
   - If $\|b - \mathcal{A}({V^{p+1}}^\top V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}({V^{p+1}}^\top V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

The Augmented Mixing Method

Algorithm 3 `Augmented Mixing Method`

1. Choose starting values V^0, y^0, μ^0, and set $p := 0$.
2. While $\|b - \mathcal{A}({V^p}^\top V^p)\|$ too large:
   - **For** $i = 1, \ldots, n$ **do**
     Update $v_i^p := \arg\min_{v_i \in \mathbb{R}^k} \mathcal{L}(V^p, y^p; \mu^p)$.
     end
     Set $V^{p+1} := V^p$.
   - If $\|b - \mathcal{A}({V^{p+1}}^\top V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}({V^{p+1}}^\top V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

The Augmented Mixing Method

Algorithm 3 `Augmented Mixing Method`

1. Choose starting values V^0, y^0, μ^0, and set $p := 0$.
2. While $\|b - \mathcal{A}(V^{p\top} V^p)\|$ too large:
   - **For** $i = 1, \ldots, n$ **do**
     Update $v_i^p := \arg\min_{v_i \in \mathbb{R}^k} \mathcal{L}(V^p, y^p; \mu^p)$.
     end
     Set $V^{p+1} := V^p$.
   - If $\|b - \mathcal{A}(V^{p+1\top} V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}(V^{p+1\top} V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

- subproblems only have k variables
- can easily be solved with arbitrary precision

The Augmented Mixing Method

Algorithm 3 `Augmented Mixing Method`

1. Get V^0, y^0, μ^0 by warm-starting from `SDPLR`, and set $p := 0$.
2. While $\|b - \mathcal{A}(V^{p\top} V^p)\|$ too large:
   - **For** $i = 1, \dots, n$ **do**

     Update $v_i^p := \arg\min_{v_i \in \mathbb{R}^k} \mathcal{L}(V^p, y^p; \mu^p)$.

     end

     Set $V^{p+1} := V^p$.
   - If $\|b - \mathcal{A}(V^{p+1\top} V^{p+1})\|$ has sufficiently decreased:
     - Update $y^{p+1} := y^p - \mu^p(\mathcal{A}(V^{p+1\top} V^{p+1}) - b)$.
     - Set $\mu^{p+1} := \mu^p$.

     Otherwise:
     - Set $y^{p+1} := y^p$.
     - Choose larger penalty parameter $\mu^{p+1} > \mu^p$.
   - Set $p := p + 1$.

- subproblems only have k variables
- can easily be solved with arbitrary precision

Small subproblems

Full augmented Lagrangian

$$\mathcal{L}(V, y; \mu) := \langle C, V^\top V \rangle + \frac{\mu}{2} \| b - \mathcal{A}(V^\top V) \|^2 + \langle y, b - \mathcal{A}(V^\top V) \rangle$$

Small subproblems

Full augmented Lagrangian

$$\mathcal{L}(V, y; \mu) := \langle C, V^\top V \rangle + \frac{\mu}{2}\|b - \mathcal{A}(V^\top V)\|^2 + \langle y, b - \mathcal{A}(V^\top V) \rangle$$

Subproblem

$$\min_{v_i \in \mathbb{R}^k} \mathcal{L}(V^p, y^p; \mu^p)$$

Small subproblems

Full augmented Lagrangian

$$\mathcal{L}(V, y; \mu) := \langle C, V^\top V\rangle + \frac{\mu}{2}\|b - \mathcal{A}(V^\top V)\|^2 + \langle y, b - \mathcal{A}(V^\top V)\rangle$$

Subproblem

$$\min_{v_i \in \mathbb{R}^k} \mathcal{L}(V^p, y^p; \mu^p)$$

Derivative w.r.t v_i

$$\frac{\partial}{\partial v_i}\mathcal{L}(V, y; \mu) = 2V\tilde{S}_{(i)}$$

where

$$\tilde{S} = C - \sum_{i=1}^{m} \tilde{y}_i A_i$$

$$\tilde{y} = y - \mu(\mathcal{A}(V^\top V) - b)$$

Implementation

Standard stopping criteria for SDPs

For $X, Z \in \mathcal{S}_n^+$ and $y \in \mathbb{R}^m$:

$$\texttt{pinf} := \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} < \texttt{tol}$$

$$\texttt{gap} := \frac{|\langle C, X\rangle - b^\top y|}{1 + |\langle C, X\rangle| + |b^\top y|} < \texttt{tol}$$

$$\texttt{dinf} := \frac{\|C - \mathcal{A}^\top(y) - Z\|_F}{1 + \|C\|_F} < \texttt{tol}$$

Implementation

Standard stopping criteria for SDPs

For $X, Z \in \mathcal{S}_n^+$ and $y \in \mathbb{R}^m$:

$$\texttt{pinf} := \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} < \texttt{tol}$$

$$\texttt{gap} := \frac{|\langle C, X\rangle - b^\top y|}{1 + |\langle C, X\rangle| + |b^\top y|} < \texttt{tol}$$

$$\texttt{dinf} := \frac{\|C - \mathcal{A}^\top(y) - Z\|_F}{1 + \|C\|_F} < \texttt{tol}$$

- $Z \in \mathcal{S}_n^+$ needs to be computed (expensive!)
- goal: $\texttt{tol} < 10^{-50}$

Implementation

Standard stopping criteria for SDPs

For $X, Z \in \mathcal{S}_n^+$ and $y \in \mathbb{R}^m$:

$$\texttt{pinf} := \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} < \texttt{tol}$$

$$\texttt{gap} := \frac{|\langle C, X\rangle - b^\top y|}{1 + |\langle C, X\rangle| + |b^\top y|} < \texttt{tol}$$

$$\texttt{dinf} := \frac{\|C - \mathcal{A}^\top(y) - Z\|_F}{1 + \|C\|_F} < \texttt{tol}$$

- $Z \in \mathcal{S}_n^+$ needs to be computed (expensive!)
- goal: $\texttt{tol} < 10^{-50}$
- current implementation is slow...

Implementation

Standard stopping criteria for SDPs

For $X, Z \in \mathcal{S}_n^+$ and $y \in \mathbb{R}^m$:

$$\texttt{pinf} := \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} < \texttt{tol}$$

$$\texttt{gap} := \frac{|\langle C, X\rangle - b^\top y|}{1 + |\langle C, X\rangle| + |b^\top y|} < \texttt{tol}$$

$$\texttt{dinf} := \frac{\|C - \mathcal{A}^\top(y) - Z\|_F}{1 + \|C\|_F} < \texttt{tol}$$

- $Z \in \mathcal{S}_n^+$ needs to be computed (expensive!)
- goal: $\texttt{tol} < 10^{-50}$
- current implementation is slow...
- promising: competitive on **dense** SDPs

Implementation

Standard stopping criteria for SDPs

For $X, Z \in \mathcal{S}_n^+$ and $y \in \mathbb{R}^m$:

$$\texttt{pinf} := \frac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2} < \texttt{tol}$$

$$\texttt{gap} := \frac{|\langle C, X\rangle - b^\top y|}{1 + |\langle C, X\rangle| + |b^\top y|} < \texttt{tol}$$

$$\texttt{dinf} := \frac{\|C - \mathcal{A}^\top(y) - Z\|_F}{1 + \|C\|_F} < \texttt{tol}$$

- $Z \in \mathcal{S}_n^+$ needs to be computed (expensive!)
- goal: $\texttt{tol} < 10^{-50}$
- current implementation is slow...
- promising: competitive on **dense** SDPs
- `SDPA-GMP` needs roughly one minute for $(n, m) = (20, 40)$

Conclusion and future work

Conclusion:

- arbitrary-precision solvers for SDPs are an important tool
- `Augmented Mixing Method` avoids some bottlenecks other SDP solvers are facing

Conclusion and future work

Conclusion:

- arbitrary-precision solvers for SDPs are an important tool
- `Augmented Mixing Method` avoids some bottlenecks other SDP solvers are facing

Future work:

- finalize the implementation
- extend to multiple SDP blocks
- publish a Julia package

Conclusion and future work

Conclusion:

- arbitrary-precision solvers for SDPs are an important tool
- `Augmented Mixing Method` avoids some bottlenecks other SDP solvers are facing

Future work:

- finalize the implementation
- extend to multiple SDP blocks
- publish a Julia package

Thank you!