Momentum-inspired Low-Rank Coordinate Descent for Diagonally Constrained SDPs

Antonio Lara, Zhiwei Zhang, Anastasios Kyrillidis
Department of Computer Science, Rice University
Motivation

SDP with Unit Diagonal Constraints

\[
\min_{X \succeq 0} \langle C, X \rangle \quad \text{subject to } X_{ii} = 1, \quad i \in [n].
\]

Applications: a reliable convex relaxation of
MaxCut
MaxSAT
Word embedding
Graphical model inference

Interior point methods **scale poorly**

Factorization (non-convex):

\[
\min_{V \in \mathbb{R}^{n \times k}} \langle C, V^T V \rangle \quad \text{for } \|v_i\|_2 = 1, \quad i \in [n]
\]

Search space is reduced

**low rank** \(k\) is enough for preserving the global minimum

However, the convergence is still slow
Methodology for acceleration

SDP with Unit Diagonal Constraints (convex):

\[
\min_{X \succeq 0} \langle C, X \rangle \quad \text{subject to } X_{ii} = 1, \quad i \in [n].
\]

factorization (non-convex):

\[
\min_{V \in \mathbb{R}^{n \times k}} \langle C, V^T V \rangle \quad \text{for } \|v_i\|_2 = 1, \quad i \in [n]
\]

Mixing method

In [7], the authors showed an algorithm with asymptotically convergent guarantees for (−CVX).

**Algorithm 1** Mixing Method [7]

```
Input: C, ε > 0.
while not yet less than ε do
    for i = 1 to n do
        v_i ← normalize \(-\sum_{j=1}^{n} c_{ij}v_j\)
    end for
end while
```

Mixing Method++

**Method details.** Our goal is to introduce acceleration's ideas (Polyak, 1987; Nesterov, 2013) to improve the overall performance of Mixing Method.

**Algorithm 2** Mixing Method++

```
Input: C, ε > 0, β.
while not yet less than ε do
    for i = 1 to n do
        u_i ← normalize \(-\sum_{j=1}^{n} c_{ij}v_j\)
        v_i ← normalize(u_i - βv_i)
    end for
end while
```

Theoretical Guarantee: Our method always converges to a local optimum with **linear** convergence rate.
Experimental Results on MaxCut problems

Our method, Mixing++ provides **reliable solution quality** and **significant acceleration** compared with state-of-the-art convex and non-convex SDP solvers.

We test the algorithm with 203 MaxCut instances. For all 203 instances, within 24 hours, Mixing Method++ solved most instances (111) among all solvers, and is 5x to 1316x faster than other solvers.

<table>
<thead>
<tr>
<th>Solver</th>
<th>Solved instances</th>
<th>Acceleration (x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEDUMI</td>
<td>60</td>
<td>360.20</td>
</tr>
<tr>
<td>MoSeK</td>
<td>74</td>
<td>348.37</td>
</tr>
<tr>
<td>SDPNAL+</td>
<td>52</td>
<td>1316.35</td>
</tr>
<tr>
<td>CGAL</td>
<td>92</td>
<td>49.19</td>
</tr>
<tr>
<td>SDPLR</td>
<td>95</td>
<td>9.14</td>
</tr>
<tr>
<td>Mixing Method</td>
<td>106</td>
<td>5.26</td>
</tr>
<tr>
<td>Mixing Method++</td>
<td>111</td>
<td>1</td>
</tr>
</tbody>
</table>