High-Performance Computing in Global Optimization and Optimization-Based Visualization

Julius Žilinskas

Institute of Mathematics and Informatics, Lithuania
Outline of talk

• Global optimization.
• Lipschitz optimization with improved bounds over simplices.
• Copositive optimization by simplicial partitioning.
• Multilevel optimization in multidimensional scaling – a technique for exploratory analysis of multidimensional data.
Global optimization

Find \( f^* = \min_{x \in D} f(x) \) and \( x^* \in D, \ f(x^*) = f^* \), where \( D \subseteq \mathbb{R}^n \).

Example:

- \( n = 1 \);
- \( D = [0, 10] \);
- Objective function
  \[
  f(x) = \sum_{j=1}^{5} j \sin((j+1)x+j);
  \]
- \( f^* = -12.0312 \);
- \( x^* = 5.7918 \).
Local optimization

- A point $x^*$ is a local minimum point if $f(x^*) \leq f(x)$ for $x \in N$, where $N$ is a neighborhood of $x^*$.
- A local minimum point can be found stepping in the direction of steepest descent.
- Without additional information one cannot say if the local minimum is global.
- How do we know if we are in the deepest hole?
Parallel global optimization

• Global optimization problems are classified difficult in the sense of the algorithmic complexity theory. Global optimization algorithms are computationally intensive.

• When computing power of usual computers is not sufficient to solve a practical global optimization problem, high performance parallel computers and computational grids may be helpful.

• An algorithm is more applicable in case its parallel implementation is available, because larger practical problems may be solved by means of parallel computations.
Simplicial partitions

• An $n$-simplex is the convex hull of a set of $(n + 1)$ affinely independent points in $n$-dimensional Euclidean space.

• A simplex is a polyhedron in $n$-dimensional space, which has the minimal number of vertices.

• Simplicial partitions are preferable when values of the objective function at vertices of partitions are used to evaluate sub-regions.

• A hyper-rectangular feasible region is face-to-face vertex triangulated: it is partitioned into $n$-simplices, where the vertices of simplices are also the vertices of the feasible region.

• A feasible region defined by linear inequality constraints may be vertex triangulated. In this way constraints are managed by initial covering.
Lipschitz optimization

• Lipschitz optimization is one of the most deeply investigated subjects of global optimization. It is based on the assumption that the slope of an objective function is bounded.

• The multivariate function $f : D \rightarrow \mathbb{R}, D \subset \mathbb{R}^n$ is said to be Lipschitz if it satisfies the condition

$$|f(x) - f(y)| \leq L\|x - y\|, \quad \forall x, y \in D,$$

where $L > 0$ is a constant called Lipschitz constant, the domain $D$ is compact and $\| \cdot \|$ denotes a norm.

• Branch and bound algorithm with Lipschitz bounds may be built: if the evaluated bound is worse than the known function value, the sub-region cannot contain optimal solutions and the branch describing it can be pruned.
Lipschitz bounds

- The upper bound for the maximum (or the lower bound for the minimum) is evaluated exploiting Lipschitz condition.

- The sharpest upper bound for the maximum given the knowledge of the function values $f(v)$ and of the Lipschitz constant $L$, is provided by

$$UB_F(I) = \max_{x \in I} \left( \min_{v \in V(I)} \{ f(v) + L \| x - v \| \} \right).$$

- Similarly the lower bound for the minimum is provided by

$$LB_F(I) = \min_{x \in I} \left( \max_{v \in V(I)} \{ f(v) + L \| x - v \| \} \right).$$

- Such bounds are not easy to find in multidimensional case: intersection of cones in the Euclidean norm and intersection of pyramids in the first norm.
The upper bounding function with the first norm

- The graph is the intersection of \( n \)-dimensional pyramids.
- The maximum point can be found solving a system of \( n \) linear equations.
Parallel branch and bound with Lipschitz bounds

- The progress of search and the number of evaluated sub-regions may depend on the number of processors. Pseudo efficiency

\[ pe_p = \frac{t_1/T_1}{p \times t_p/T_p}, \]

where \( T_p \) is the measure of amount of work done (NFE).
Copositivity detection by simplicial partitioning

- A matrix $A = A^T$ is called copositive if $x^T A x \geq 0$ for all $x \in \mathbb{R}_+^n$.
- Recently algorithmic copositivity detection by simplicial partition has been propose.
- The algorithm starts with the standard simplex, whose vertices are the unit vectors $e_1, \ldots, e_n$.
- Simplices are subdivided until:
  - $v^T A v < 0$ for one vertex $v$ of one of the simplices what means that the matrix $A$ is not copositive or
  - $v_i^T A v_j \geq 0$ for all pairs of vertices $v_i$ and $v_j$ of all simplices what means that the matrix $A$ is copositive.
- Depth first selection without storing the set of simplices and corresponding matrices may be applied.
Copositivity detection for solution of copositive programs

• A quadratic programming problem with a single quadratic constraint

\[
\min \langle Q, X \rangle \quad \text{s.t.} \quad \langle D, X \rangle = b, \quad X = xx^T, \quad x \geq 0.
\]

• Copositive formulation with a variable \( y \in \mathbb{R} \) and the cone of copositive matrices \( C \)

\[
\max \{ y : Q - yD \in C \}.
\]

• The maximum clique problem may be formulated as

\[
\omega(G) = \min \{ t : tQ - J \in C \},
\]

where \( t \in \mathbb{N} \) is a variable, \( \omega(G) \) is the clique number, and
\( Q = J - A_G \), where \( A_G \) is the adjacency matrix of the graph \( G \).
Reformulation of conditions in copositivity detection

• Observe that for the problem \( \max \{ y : Q - yD \in C \} \), \( A = Q - yD \) with copositive \( D \), the condition \( v_i^T A v_j \geq 0 \) can be rewritten as

\[
y \leq \frac{v_i^T Q v_j}{v_i^T D v_j} : v_i^T (Q - yD) v_j = v_i^T Q v_j - y v_i^T D v_j.
\]

• Therefore, the matrix \( A \) is not copositive, if

\[
y > \frac{v^T Q v}{v^T D v}
\]

for one vertex \( v \) of one of the simplices in the partition.

• Moreover, the matrix \( Q - (y - \varepsilon)D \) is copositive if

\[
y - \varepsilon \leq \frac{v_i^T Q v_j}{v_i^T D v_j}
\]

for all vertices \( v_i, v_j \) of all simplices in the partition \( \mathcal{P} \).
Sketch of the algorithm for $\max\{y : Q - yD \in C\}$, $
abla$
\begin{align*}
\max\{y : Q - yJ \in C\}, \min\{t : tQ - J \in C\}
\end{align*}
Start with the standard simplex

while not stopped do

\begin{align*}
y & \leftarrow \min \left\{ y, \frac{v_i^T Q v_i}{v_i^T D v_i} \right\}, \min \{ y, v_i^T Q v_i \}, t \leftarrow \max \left\{ t, \frac{1}{v_i^T Q v_i} \right\}, \\
i = 1, \ldots, n
\end{align*}

if $y - \varepsilon \leq \frac{v_i^T Q v_i}{v_i^T D v_j}$, $y - \varepsilon \leq v_i^T Q v_j$, $t + \varepsilon \geq \frac{1}{v_i^T Q v_j}$, $i, j = 1, \ldots, n$
then

restore vertices, change vertex or stop the cycle
remember the changed vertex

else

subdivide the edge with the smallest $\frac{v_i^T Q v_i}{v_i^T D v_j}$, $v_i^T Q v_j$, $v_i^T Q v_j$
remember the changed vertex

end if

end while
## Results for the DIMACS benchmark problems

<table>
<thead>
<tr>
<th>Graph</th>
<th>Nodes</th>
<th>Edges</th>
<th>Clique number</th>
<th>t</th>
<th>$n_{simp}$</th>
<th>max l</th>
<th>time, s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brock200_1</td>
<td>200</td>
<td>14834</td>
<td>21</td>
<td>16</td>
<td>708261</td>
<td>1989</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock200_2</td>
<td>200</td>
<td>9876</td>
<td>12</td>
<td>10</td>
<td>710547</td>
<td>980</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock200_3</td>
<td>200</td>
<td>12048</td>
<td>15</td>
<td>11</td>
<td>699520</td>
<td>1330</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock200_4</td>
<td>200</td>
<td>13089</td>
<td>17</td>
<td>14</td>
<td>708598</td>
<td>1504</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock400_1</td>
<td>400</td>
<td>59723</td>
<td>27</td>
<td>20</td>
<td>18115</td>
<td>4984</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock400_2</td>
<td>400</td>
<td>59786</td>
<td>29</td>
<td>20</td>
<td>17955</td>
<td>4946</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock400_3</td>
<td>400</td>
<td>59681</td>
<td>31</td>
<td>20</td>
<td>18027</td>
<td>4920</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock400_4</td>
<td>400</td>
<td>59765</td>
<td>33</td>
<td>20</td>
<td>18162</td>
<td>4978</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Brock800_1</td>
<td>800</td>
<td>207505</td>
<td>23</td>
<td>17</td>
<td>19290</td>
<td>8376</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Brock800_2</td>
<td>800</td>
<td>208166</td>
<td>24</td>
<td>17</td>
<td>19317</td>
<td>8482</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Brock800_3</td>
<td>800</td>
<td>207333</td>
<td>25</td>
<td>17</td>
<td>19275</td>
<td>8385</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Brock800_4</td>
<td>800</td>
<td>207643</td>
<td>26</td>
<td>16</td>
<td>19379</td>
<td>8389</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Hamming6-2</td>
<td>64</td>
<td>1824</td>
<td>32</td>
<td>32</td>
<td>15805366</td>
<td>883</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Hamming6-4</td>
<td>64</td>
<td>704</td>
<td>4</td>
<td>4</td>
<td>16612788</td>
<td>112</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Hamming8-2</td>
<td>256</td>
<td>31616</td>
<td>128</td>
<td>128</td>
<td>128307</td>
<td>14441</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Hamming8-4</td>
<td>256</td>
<td>20864</td>
<td>16</td>
<td>16</td>
<td>128632</td>
<td>1628</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Hamming10-2</td>
<td>1024</td>
<td>518656</td>
<td>512</td>
<td>512</td>
<td>3899</td>
<td>3899</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Hamming10-4</td>
<td>1024</td>
<td>434176</td>
<td>40</td>
<td>33</td>
<td>3857</td>
<td>3857</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Johnson8-2-4</td>
<td>28</td>
<td>210</td>
<td>4</td>
<td>4</td>
<td>230691654</td>
<td>56</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Johnson8-4-4</td>
<td>70</td>
<td>1855</td>
<td>14</td>
<td>14</td>
<td>14850322</td>
<td>405</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Johnson16-2-4</td>
<td>120</td>
<td>5460</td>
<td>8</td>
<td>8</td>
<td>3202090</td>
<td>488</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Johnson32-2-4</td>
<td>496</td>
<td>107880</td>
<td>16</td>
<td>16</td>
<td>7818</td>
<td>4308</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Keller4</td>
<td>171</td>
<td>9435</td>
<td>11</td>
<td>8</td>
<td>696414</td>
<td>794</td>
<td>$10^4$</td>
</tr>
<tr>
<td>Keller5</td>
<td>776</td>
<td>225990</td>
<td>27</td>
<td>20</td>
<td>19377</td>
<td>8608</td>
<td>$10^5$</td>
</tr>
<tr>
<td>Keller6</td>
<td>3361</td>
<td>4619898</td>
<td>$\geq59$</td>
<td>37</td>
<td>1493</td>
<td>1493</td>
<td>$10^6$</td>
</tr>
<tr>
<td>MANN_a9</td>
<td>45</td>
<td>918</td>
<td>16</td>
<td>16</td>
<td>57018692</td>
<td>320</td>
<td>$10^4$</td>
</tr>
<tr>
<td>MANN_a27</td>
<td>378</td>
<td>70551</td>
<td>126</td>
<td>121</td>
<td>242995</td>
<td>23055</td>
<td>$10^5$</td>
</tr>
<tr>
<td>MANN_a45</td>
<td>1035</td>
<td>533115</td>
<td>345</td>
<td>336</td>
<td>74763</td>
<td>74763</td>
<td>$10^6$</td>
</tr>
<tr>
<td>MANN_a81</td>
<td>3321</td>
<td>5506380</td>
<td>$\geq1100$</td>
<td>302</td>
<td>1550</td>
<td>1550</td>
<td>$10^6$</td>
</tr>
</tbody>
</table>
Multidimensional scaling (MDS) – a technique for exploratory analysis of multidimensional data

- Pairwise dissimilarities between \( n \) objects are given by a matrix \((\delta_{ij}), i, j = 1, \ldots, n\), it is supposed that \(\delta_{ij} = \delta_{ji}\).

- The points representing objects in an \( m \)-dimensional embedding space \( x_i \in \mathbb{R}^m, i = 1, \ldots, n \) should be found whose inter-point distances fit the given dissimilarities.

- The problem is reduced to minimization of a fitness criterion, e.g. so called \( \textit{STRESS} \) function

\[
S(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (d(x_i, x_j) - \delta_{ij})^2 ,
\]

where \( x = \{x_1, \ldots, x_n\} \); \( d(x_i, x_j) \) denotes the distance between the points \( x_i \) and \( x_j \); weights \( w_{ij} > 0, i, j = 1, \ldots, n \).
MDS is a difficult global optimization problem

- Although \( STRESS \) function is defined by an analytical formula which seems rather simple, it normally has many local minima.

- The problem is high dimensional: \( x \in \mathbb{R}^N \) and the number of variables is equal to \( N = n \times m \).

- \( STRESS \) function is invariant with respect to translation, rotation and mirroring.

- Smoothness of \( STRESS \) function depends on distances \( d(x_i, x_j) \), however, non-differentiability normally cannot be ignored.

Minkowski distances

\[
d_r(x_i, x_j) = \left( \sum_{k=1}^{m} |x_{ik} - x_{jk}|^r \right)^{1/r}.
\]
Example of multidimensional data: Experimental testing of soft drinks

<table>
<thead>
<tr>
<th>Soft drinks</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Pepsi</td>
<td>0</td>
<td>127</td>
<td>169</td>
<td>204</td>
<td>309</td>
<td>320</td>
<td>286</td>
<td>317</td>
<td>321</td>
<td>238</td>
</tr>
<tr>
<td>2. Coke</td>
<td>127</td>
<td>0</td>
<td>143</td>
<td>235</td>
<td>318</td>
<td>322</td>
<td>256</td>
<td>318</td>
<td>318</td>
<td>231</td>
</tr>
<tr>
<td>3. Classic Coke</td>
<td>169</td>
<td>143</td>
<td>0</td>
<td>243</td>
<td>326</td>
<td>327</td>
<td>258</td>
<td>318</td>
<td>318</td>
<td>242</td>
</tr>
<tr>
<td>4. Diet Pepsi</td>
<td>204</td>
<td>235</td>
<td>243</td>
<td>0</td>
<td>285</td>
<td>288</td>
<td>259</td>
<td>312</td>
<td>317</td>
<td>194</td>
</tr>
<tr>
<td>5. Diet Slice</td>
<td>309</td>
<td>318</td>
<td>326</td>
<td>285</td>
<td>0</td>
<td>155</td>
<td>312</td>
<td>131</td>
<td>170</td>
<td>285</td>
</tr>
<tr>
<td>6. Diet 7-Up</td>
<td>320</td>
<td>322</td>
<td>327</td>
<td>288</td>
<td>155</td>
<td>0</td>
<td>306</td>
<td>164</td>
<td>136</td>
<td>281</td>
</tr>
<tr>
<td>7. Dr Pepper</td>
<td>286</td>
<td>256</td>
<td>258</td>
<td>259</td>
<td>312</td>
<td>306</td>
<td>0</td>
<td>300</td>
<td>295</td>
<td>256</td>
</tr>
<tr>
<td>8. Slice</td>
<td>317</td>
<td>318</td>
<td>318</td>
<td>312</td>
<td>131</td>
<td>164</td>
<td>300</td>
<td>0</td>
<td>132</td>
<td>291</td>
</tr>
<tr>
<td>9. 7-Up</td>
<td>321</td>
<td>318</td>
<td>318</td>
<td>317</td>
<td>170</td>
<td>136</td>
<td>295</td>
<td>132</td>
<td>0</td>
<td>297</td>
</tr>
<tr>
<td>10. Tab</td>
<td>238</td>
<td>231</td>
<td>242</td>
<td>194</td>
<td>285</td>
<td>281</td>
<td>256</td>
<td>291</td>
<td>297</td>
<td>0</td>
</tr>
</tbody>
</table>
MDS with city-block distances

- If city-block distances $d_1(x_i, x_j)$ are used, $STRESS$ can be redefined as

$$S(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left( \sum_{k=1}^{m} |x_{ik} - x_{jk}| - \delta_{ij}\right)^2.$$

- In the case of city-block distances and $m \geq 2$ $STRESS$ can be non-differentiable even at a minimum point. With this respect the case of city-block distances is different from the other cases of Minkowski distances when positiveness of distances $d(x^*_i, x^*_j)$, $i, j = 1, \ldots, n$ at a local minimum point $x^*$ implies differentiability of $STRESS$.

- However $STRESS$ with city-block distances is piecewise quadratic, and such a structure can be exploited for tailoring of ad hoc global optimization algorithms.
Two level optimization for MDS (JOGO, 2007)

• Taking into account the structure of the minimization problem a two level minimization method can be applied: to solve a combinatorial problem at the upper level, and to solve a quadratic programming problem at the lower level:

\[
\min_{\mathbf{P}} S(\mathbf{P}), \text{ s.t. } S(\mathbf{P}) = \min_{\mathbf{x} \in A(\mathbf{P})} S(\mathbf{x}) \sim \\
\sim \min \left( -\mathbf{c}_\mathbf{P}^T \mathbf{x} + \frac{1}{2} \mathbf{x}^T \mathbf{Q}_\mathbf{P} \mathbf{x} \right) \text{ s.t. } \mathbf{E} \mathbf{x} = 0, \; \mathbf{A}_\mathbf{P} \mathbf{x} \geq 0,
\]

\[
\mathbf{c}_\mathbf{P} \in \mathbb{R}^{nm}, \; \mathbf{Q}_\mathbf{P} \in \mathbb{R}^{nm \times nm}, \; \mathbf{E} \in \mathbb{R}^{m \times nm}, \; \mathbf{A}_\mathbf{P} \in \mathbb{R}^{(n-1) \times nm}.
\]

• The upper level problem is defined over the set of \( m \)-tuple of permutations of \( 1, \ldots, n \). It can be solved exactly using explicit enumeration of all feasible solutions or branch and bound. Evolutionary algorithm is applied for larger problems.
Solution of the lower level problem

- The lower level problem is convex quadratic programming problem with positive definite matrix and linear constraints.
- A standard quadratic programming method can be applied.
- quadprog_1.4.7 relies on
  - dpofa factors a double precision symmetric positive definite matrix.
  - dposl solves the double precision symmetric positive definite system \( a \times x = b \) using the factors computed by dpofa.
  - dpori computes the inverse of the factor of a double precision symmetric positive definite matrix using the factors computed by dpofa. Uses dscal and daxpy.
Explicit enumeration, b&b (JOGO, 2009), parallel b&b
Efficiency of parallelization of parallel b&b

\[ e_p = \frac{s_p}{p} = \frac{t_1}{p \times t_p}, \quad pe_p = \frac{t_1/T_1}{p \times t_p/T_p}, \]

\( T_p \) is the measure of amount of work done (the number of lower level problems).
Results of parallel evolutionary (memetic) algorithm (CAMWA, 2006)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>$f^*$</td>
<td>perc</td>
</tr>
<tr>
<td>6-dimensional cube</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>0.3513</td>
<td>100</td>
</tr>
<tr>
<td>63-dimensional simplex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>0.3998</td>
<td>100</td>
</tr>
<tr>
<td>Iris data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>0.04710</td>
<td>100</td>
</tr>
<tr>
<td>Morse code confusion data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>0.2944</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>153.001</td>
<td></td>
</tr>
</tbody>
</table>
Possible directions of future research and collaboration

- Parallelization of optimization algorithms at the level of linear algebra.
- Multilevel parallelization at different levels.
- New parallel optimization algorithms for multidimensional scaling: exact and heuristic.