A Cone Decomposition Method for Optimal Contribution Selection in Forest Tree Management

Sena Safarina[1], Tim J. Mullin[2], and Makoto Yamashita[1]

[1] *Department of Mathematical and Computing Science, Tokyo Institute of Technology, 2-12-1-W8-29 Ookayama, Meguro-ku, Tokyo 152-8552, Japan.*

[2] *The Swedish Forestry Research Institute (Skogforsk), Box 3, Sävar 918 21, Sweden; and 224 rue du Grand-Royal Est, QC, J2M 1R5, Canada.*

1 Optimal Contribution Selection

In a forest tree management, one of essential phases for tree improvement is on recurrent cycles of selection. In that phase, genetic diversity is a main consideration for genetic gain performance in the future. Therefore, an objective of optimal contribution selection (OCS) [1, 8, 11, 16] is to maximize the genetic benefit under a genetic diversity constraint by determining the gene contribution from each candidate.

This paper is concerned on OCS with an equal deployment problem (EDP) that designates a specified number of selected individuals to have equally contribution to the gene pool; and it can be formulated as:

$$\begin{array}{lll} \text{maximize} & : & \boldsymbol{g}^T\boldsymbol{x} \\ \text{subject to} & : & \boldsymbol{e}^T\boldsymbol{x} = 1,\ x_i \in \left\{0, \frac{1}{N}\right\}\ (i = 1, \ldots, m),\ \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 2\theta. \end{array} \tag{1}$$

Here, the objective is to maximize the total benefit $\boldsymbol{g}^T\boldsymbol{x}$ where $\boldsymbol{g} = (g_1, g_2, \ldots, g_m)^T$ is the estimated breeding values (EBVs) [7] representing the genetic value of candidates in the gene contribution $\boldsymbol{x} \in \mathbb{R}^m$, and m is the total number of candidates. In this objective function, our decision variable is $\boldsymbol{x}$ and we assume that $\boldsymbol{g}$ is given. The first constraint $\boldsymbol{e}^T\boldsymbol{x} = 1$, with a vector of ones $\boldsymbol{e} \in \mathbb{R}^m$, demands unity of the total contribution from all candidates. The second constraint $x_i \in \left\{0, \frac{1}{N}\right\}$ interprets an equal contribution from each candidate, with N being the parameter to indicate the number of chosen candidates. Shortly speaking, N individuals has to be exactly chosen from a list of m available candidates in the EDP.

The last constraint in (1), $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 2\theta$, is our substantial constraint that requires a group coancestry $\frac{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}{2}$ be under an appropriate level $\theta > 0$. If the group coancestry $\frac{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}{2}$ is too high, the close relatedness among individuals in the population will decrease genetic diversity and also impact on the reduction of long-term genetic performance. The group coancestry [3] is computed with the Wright numerator relationship matrix [15] $\boldsymbol{A} \in \mathbb{R}^{m \times m}$; and [12] emphasized that $\boldsymbol{A}$ is always symmetric and semi-definite positive.

In recent years, the OCS with EDP has been solved through a software package `GENCONT` [8] which is an implementation based on Lagrange multipliers, but it forcibly fixes variables that exceed lower or upper bound $\left(0 \leq x_i \leq \frac{1}{N}\right)$ at the corresponding lower and upper bound. Thus, even though GENCONT generates a solution quickly, the solution is often suboptimal. To resolve this difficulty in GENCONT, dsOpt, integrated in the software package OPSEL [9], was proposed by Mullin and Belotti[11]. Since dsOpt implements the branch-and-bound method combined with an outer approximation method [4], dsOpt generates a very large number of subproblems in the framework of branch-and-bound. This implementation is designed to acquire exact optimal solutions, but computing the solution takes a long time.

To deliver the problem in [8, 11], we consider to employ a second-order cone form into the quadratic constraint in 1. Utilizing Cholesky factorization of $\boldsymbol{A}$ so that $\boldsymbol{A} = \boldsymbol{U}^T\boldsymbol{U}$, $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 2\theta$ can be reformulated:

$$\begin{aligned} \text{maximize} \quad &: \quad \boldsymbol{g}^T\boldsymbol{x} \\ \text{subject to} \quad &: \quad \boldsymbol{e}^T\boldsymbol{x} = 1,\ x_i \in \left\{0, \tfrac{1}{N}\right\}\ (i = 1, \ldots, m),\ \left(\sqrt{2\theta}N, \boldsymbol{U}\boldsymbol{x}\right) \in \mathcal{K}^m. \end{aligned} \tag{2}$$

with $\mathcal{K}^m = \{(v_0, \boldsymbol{v}) \in \mathbb{R}_+ \times \mathbb{R}^m : ||\boldsymbol{v}||_2 \leq v_0\}$ is an $(m+1)$-dimensional second-order cone. However, non-linearity arising from this second-order cone also leads to a heavy computation cost.

In this paper, we examine and propose a cone decomposition method (CDM) that is based on a geometric cut in a combination with a Lagrangian multiplier method and also draws on another form of second-order cones. A cutting plane is a geometric cut if the plane is computed with an orthogonal projection [2]. Cone decomposition itself has already been used in CPLEX which depends on an outer approximation, therefore, the proposed CDM generates a different linear approximation.

In addition, we prove that the Lagrangian multiplier method in the framework of CDM gives the analytical form for the geometric cut, therefore, the proposed CDM generates the linear cuts without relying on iterative methods.

The remainder of this paper is organized as follows. In Section 2, we propose the framework of CDM and demonstrates that the geometric cut in CDM has an analytical form. The numerical results will be presented in Section 3. Finally, in Section 4, we give some conclusions and discuss future studies.

2 Cone Decomposition Method

In this section, we focus on the proposed cone decomposition method (CDM) for EDP (2) that impose another form of second-order cones as in the following corollary [13].

Corollary 2.1. *A second-order cone $\mathcal{K}^m$ can be also written as*

$$\mathcal{K}^m = \left\{ (v_0, \boldsymbol{v}) \in \mathbb{R}^{m+1} : \exists \boldsymbol{w} \in \mathbb{R}^m \text{ such that } v_j^2 \leq w_j v_0\ (j = 1, \ldots, m),\ \sum_{j=1}^{m} w_j \leq v_0,\ v_0 \geq 0 \right\}.$$

Proof. Let $\hat{\mathcal{K}}^m$ be $\left\{(v_0, \boldsymbol{v}) \in \mathbb{R}^{m+1} : \exists \boldsymbol{w} \in \mathbb{R}^m \text{ s.t. } v_j^2 \leq w_j v_0\ (j = 1, \ldots, m),\ \sum_{j=1}^m w_j \leq v_0,\ v_0 \geq 0\right\}$.

For $(\hat{v}_0, \hat{\boldsymbol{v}}) \in \hat{\mathcal{K}}^m$, if $\hat{v}_0 = 0$, then $\hat{\boldsymbol{v}} = \boldsymbol{0}$ due to the constraint $\hat{v}_j^2 \leq w_j \hat{v}_0$, therefore, we know $(\hat{v}_0, \hat{\boldsymbol{v}}) \in \mathcal{K}^m$. In the case $\hat{v}_0 > 0$, it holds that $\hat{v}_0 \geq \sum_{j=1}^m w_j \geq \sum_{j=1}^m \hat{v}_j^2/\hat{v}_0$, and this leads to $\hat{v}_0 \geq \sqrt{\sum_{j=1}^m \hat{v}_j^2}$.

Conversely, we take $(v_0, \boldsymbol{v}) \in \mathcal{K}^m$. If $v_0 = 0$, we again have $\boldsymbol{v} = 0$; thus $(v_0, \boldsymbol{v}) \in \hat{\mathcal{K}}^m$. For positive v_0, we can use $w_j = v_j^2/v_0$ to show $(v_0, \boldsymbol{v}) \in \hat{\mathcal{K}}^m$. □

Utilizing Corollary 2.1 and introducing new variable $\boldsymbol{y} = N\boldsymbol{x}$ to EDP (2) derives mixed-integer quadratic constraint problem (MI-QCP):

$$\begin{array}{lll} \text{maximize} & : & \frac{\boldsymbol{g}^T\boldsymbol{y}}{N} \\ \text{subject to} & : & \boldsymbol{e}^T\boldsymbol{y} = N, \boldsymbol{z} = \boldsymbol{U}\boldsymbol{y}, \\ & & z_i^2 \leq w_i c_0 \ (i = 1, \ldots, m), \ \sum_{i=1}^m w_i \leq c_0, \ y_i \in \{0,1\} \ (i = 1, \ldots, m) \end{array} \tag{3}$$

where z_i is the ith element of $\boldsymbol{z}$, $c_0 = \sqrt{2\theta N^2}$, and the decision variables of our new formulation are $\boldsymbol{y}, \boldsymbol{z}$, and $\boldsymbol{w}$.

The nonlinear constraint in (3) is only the quadratic constraint $z_i^2 \leq w_i c_0$ with two variables z_i and w_i. In the proposed CDM, we generate the geometric cuts as cutting planes to these quadratic constraint by employing orthogonal projections [2]. Therefore, the framework of the proposed CDM is given as Algorithm 2.2.

Algorithm 2.2. [Cone decomposition method (CDM)]

Step 1 Let P^0 be an MI-LP problem that is generated from an optimization problem (3) by omitting the quadratic constraints $z_i^2 \leq w_i c_0$ $(i = 1, \ldots, m)$. Apply an MI-LP solver to P^0, and let its optimal solution be $\left(\hat{\boldsymbol{y}}^0, \hat{\boldsymbol{z}}^0, \hat{\boldsymbol{w}}^0\right)$. Let $k = 0$.

Step 2 Let a set of generated cuts $\mathcal{C}^k = \emptyset$.

Step 3 For each $i = 1, \ldots, m$, if $(\hat{z}_i^k)^2 \leq \hat{w}_i^k c_0$ is violated, apply the following steps.

Step 3-1 Compute the orthogonal projection of $(\hat{z}_i^k, \hat{w}_i^k)$ onto $z_i^2 \leq w_i c_0$ by solving the following sub-problem with the Lagrangian multiplier method;

$$\begin{array}{ll} \text{minimize} & : \frac{1}{2}\left(\bar{z} - \hat{z}_i^k\right)^2 + \frac{1}{2}\left(\bar{w} - \hat{w}_i^k\right)^2 \\ \text{subject to} & : \bar{z}^2 \leq \bar{w} c_0. \end{array} \tag{4}$$

Let $(\bar{z}_i^k, \bar{w}_i^k)$ be the solution of this subproblem.

Step 3-2 Add to $\mathcal{C}^k$ the following linear constraint

$$\begin{pmatrix} \hat{z}_i^k - \bar{z}_i^k \\ \hat{w}_i^k - \bar{w}_i^k \end{pmatrix}^T \begin{pmatrix} z_i - \bar{z}_i^k \\ w_i - \bar{w}_i^k \end{pmatrix} \leq 0.$$

Step 4 If $\mathcal{C}^k$ is empty, output $\hat{\boldsymbol{y}}^k$ as the solution and terminate.

Step 5 Build a new MI-LP P^{k+1} by adding $\mathcal{C}^k$ to P^k. Let the optimal solution of P^{k+1} be $\left(\hat{\boldsymbol{y}}^{k+1}, \hat{\boldsymbol{z}}^{k+1}, \hat{\boldsymbol{w}}^{k+1}\right)$. Return to Step 2 with $k \leftarrow k + 1$.

Step 3-1 of Algorithm 2.2 computes the orthogonal projection. It would be desirable to compute the orthogonal projection on the original quadratic constraint $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 2\theta$, but such orthogonal projection does not have an analytic form. A numerical method has been proposed in [6], but it is an iterative method. Another iterative method is also proposed by [5] to solve different case of second-order cones. In contrast, the orthogonal projection in Step 3-1 is onto a specific cone $\bar{z}^2 \leq \bar{w} c_0$. The decomposition in Corollary 2.1 enables us to derive its analytical form, as proven in the following theorem.

Theorem 2.3. *Assume that* $(\hat{z}, \hat{w}) \in \mathbb{R}^2$ *violates* $\hat{z}^2 \leq \hat{w}c_0$*. Let* $(\bar{z}, \bar{w}) \in \mathbb{R}^2$ *be the orthogonal projection of* $(\hat{z}, \hat{w})$ *onto* $z^2 \leq wc_0$*. Then,* $(\bar{z}, \bar{w})$ *can be given by an analytical form.*

Proof. As in Step 3-1 of Algorithm 2.2, the orthogonal projection $(\bar{z}, \bar{w}) \in \mathbb{R}^2$ is the optimal solution of the subproblem (4) that has a convex closed feasible region. Since $(\hat{z}, \hat{w})$ is outside of the region $z^2 \leq wc_0$, the optimal solution of (4) can be obtained with the following problem:

$$\begin{array}{ll} \text{minimize} & : \frac{1}{2}(z-\hat{z})^2 + \frac{1}{2}(w-\hat{w})^2 \\ \text{subject to} & : z^2 = wc_0. \end{array} \tag{5}$$

Next, we define a Lagrangian function of (5) with a Lagrangian multiplier $\lambda \in \mathbb{R}$ as:

$$\mathcal{L}(z, w, \lambda) = \frac{1}{2}(z-\hat{z})^2 + \frac{1}{2}(w-\hat{w})^2 - \lambda(wc_0 - z^2).$$

We consider $\nabla\mathcal{L} = 0$ in the Lagrangian multiplier method, therefore we obtain the conditions below

$$z - \hat{z} + 2\lambda c_0 = 0; \;\; w - \hat{w} - \lambda c_0 = 0; \;\; -c_0 w + z^2 = 0$$

that results in a following cubic function with respect to λ:

$$4c_0^2\lambda^3 + (4c_0^2 + 4c_0\hat{w})\lambda^2 + (c_0^2 + 4c_0\hat{w})\lambda + (c_0\hat{w} - (\hat{z})^2) = 0.$$

When we apply Cardano's Formula [14] to this cubic function, we obtain only one real root $\bar{\lambda}$. This leads to $\bar{z} = \hat{z} - 2\bar{\lambda}c_0$ and $\bar{w} = \hat{w} + \bar{\lambda}c_0$. Therefore, the optimal solution $(\bar{z}, \bar{w})$ of (4) has an analytical form. □

The termination of the proposed method is guaranteed by the following theorem.

Theorem 2.4. *Algorithm 2.2 terminates in a finite number of iterations.*

Proof. Due to the binary constraints $y_i \in \{0, 1\}$ for $i = 1, \ldots, m$, the number of solution candidates is at most 2^m. In the proposed method, we remove at least one candidate, therefore, the number of iterations is bounded above by 2^m. □

3 Numerical Experiment

Numerical experiments were conducted for performance comparison of the proposed method CDM, with the existing software `dsOpt` (as integrated in `OPSEL`) and `GENCONT`, and a general MI-SOCP solver `CPLEX`. We implemented CDM in `MATLAB 9.3.0.713579 (R2017b)` by setting `CPLEX 12.71` as the MI-LP solver. All numerical experiments were performed on Intel(R) Xeon(R) CPU E3-1231 (3.40 GHz) and 8 GB memory space under 64-bit Windows 10 operating system. The generated data by the simulation POPSIM [10] were taken from `https://doi.org/10.5061/dryad.9pn5m`. The sizes of the test instances are $m = 200, 1050, 2045, 5050, 10100,$ and 15222. We set parameter $N = 50, 100$, and as a stopping criterion for `CPLEX`, we used gap $= 1\%, 5\%$. The computation time was limited to 3 hours for each execution.

Table 1 shows the results from the OCS solver `GENCONT`. In this table, the first, second, and third columns are the given parameter N, number of candidates m, and 2θ. The columns "$\boldsymbol{g}^T\boldsymbol{x}$" and "$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$" are the obtained objective values and group coancestry, respectively. Table 1 shows

Table 1: The result from `GENCONT`

N	m	2θ	$\boldsymbol{g}^T\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$	time (sec)	# chosen N
50	200	0.0334	11.472	0.03340	3.54	64
	1050	0.0627	25.91	0.06270	7.20	81
	2045	0.0711	438.36	0.07109	111.52	71
	5050	0.1081	43.44	0.10810	1561.43	78
100	200	0.0258	8.89	0.02580	0.48	93
	1050	0.0539	24.07	0.0539	4.77	94
	2045	0.0628	432.75	0.06279	106.48	74
	5050	0.0994	42.08	0.09940	1533.31	81

the computation time in the sixth column; and the number of chosen candidates (# chosen N) by `GENCONT` in the last column. For a feasible solution $\boldsymbol{x}$, it should hold $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq 2\theta$ and the number of chosen candidate should be exactly N. However, the numerical result shows that # chosen N did not match the given N so that `GENCONT` failed to obtain feasible solutions. In addition, insufficient memory involved failure to output solution for $m > 5050$.

Tables 2 and 3 shows feasible solutions of the other methods for the given parameter $N = 50$ and $N = 100$, respectively. Contrary to `GENCONT`, the number of chosen candidates of the rest methods match the given N. the first column displays different methods for our numerical experiment. In both tables, `CPLEX-default` means that we used the default setting in `CPLEX`, and `CPLEX-LPrelax` means that we explicitly set a parameter so that `CPLEX` used LP relaxation forcibly. We set the time limit of 3 hours, and we indicate the violations of this time limit by '> 3 hours' and we used the best objective values in the 3 hours. In the case of out of memory, we used "OOM."

`CPLEX-default` shows its computation efficiency when the gap (the stopping criterion) is 5%. On the other hand, for larger problems or smaller gaps, `CPLEX-default` is more time-consuming than other methods. For instance, we can see a large time difference for the smallest size $m = 200$. `CPLEX-default` for gap 5% is the most efficient method among the seven methods, but it turns to be the slowest method when we set the gap as 1%. For such a tight gap, `CPLEX-LPrelax` and CDM can reduce computation time to less than 5 seconds. In the case $m = 15222$, `CPLEX-default` could not finish its computation within the time limit (three hours), and the best objective value in the three hours was much worse than `CPLEX-LPrelax` and CDM; `CPLEX-default` only reached $\boldsymbol{g}^T\boldsymbol{x} = 107.56$.

From the difference between the results of `CPLEX-default` and those of `CPLEX-LPrelax`, we can infer that the default setting of `CPLEX` cannot solve EDPs efficiently, and we have to explicitly let `CPLEX` know that LP relaxation is effective for EDPs.

Table 3 shows the results for the case $N = 100$. Similar with the result in the previous table that `CPLEX-LPrelax` and CDM obtain feasible solutions without having a memory problem. However, when our proposed method CDM is compared with `CPLEX-LPrelax`, `CPLEX-LPrelax` gives better computation time performance than CDM for $m \leq 10100$. This is not only shown by Table 3, but also it is on Table 2. For example, the computation time of CDM is 5 times slower than `CPLEX-LPrelax` to generate the solution of OCS with $m = 10100$. Only for the largest size problem $m = 15222$, CDM can show its efficiency among all methods.

Table 2: Numerical comparison for EDPs ($N = 50$)

Method	m	2θ	gap = 5%			gap = 1%		
			$\boldsymbol{g}^T\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$	time (sec)	$\boldsymbol{g}^T\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$	time (sec)
`CPLEX-default`	200	0.0334	24.99	0.03340	1.06	25.19	0.03340	8735.24
`CPLEX-LPrelax`			25.16	0.03340	1.96	25.19	0.03340	3.96
`dsOpt`			25.12	0.03340	5.32	25.18	0.03340	606.94
CDM			25.02	0.03340	1.69	25.15	0.03340	1.72
`CPLEX-default`	1050	0.0627	24.97	0.06267	3.56	24.97	0.06267	6.64
`CPLEX-LPrelax`			24.94	0.06265	4.27	24.94	0.06265	4.64
`dsOpt`			24.97	0.06169	5.19	24.85	0.06268	> 3 hours
CDM			24.65	0.06118	9.41	24.96	0.06238	12.11
`CPLEX-default`	2045	0.0711	437.21	0.07100	3.95	437.21	0.07100	3.83
`CPLEX-LPrelax`			438.07	0.07060	2.97	438.08	0.07060	3.52
`dsOpt`			432.94	0.06700	7.09	435.87	0.07020	14.42
CDM			434.26	0.06760	1.76	437.38	0.06960	2.52
`CPLEX-default`	5050	0.1081	41.90	0.10776	73.16	42.57	0.10781	> 3 hours
`CPLEX-LPrelax`			42.46	0.10658	11.42	42.46	0.10658	15.19
`dsOpt`			41.57	0.10471	236.70	42.67	0.10807	> 3 hours
CDM			42.56	0.10742	187.24	42.56	0.10742	182.10
`CPLEX-default`	10100	0.0701	44.89	0.06931	> 3 hours	44.89	0.06931	> 3 hours
`CPLEX-LPrelax`			45.91	0.06789	104.44	46.48	0.07008	200.55
`dsOpt`			46.00	0.07005	4509.83	46.21	0.06975	8787.37
CDM			45.27	0.06896	1003.67	46.43	0.07005	1204.47
`CPLEX-default`	15222	0.0388	118.33	0.03840	> 3 hours	107.56	0.03280	> 3 hours
`CPLEX-LPrelax`			454.07	0.03860	350.14	458.85	0.03880	1080.17
`dsOpt`					OOM			OOM
CDM			452.57	0.03880	450.84	461.83	0.03880	547.02

4 Conclusion and Future Work

In this study, we proposed the implementation of cone decomposition method to optimal contribution selection in forest tree management. The computation time problem difficulty from `OPSEL` makes us consider to propose the efficiency methods for solving OCS. We compared the efficiency of our proposed implementation with the existing breeding selection software (`GENCONT` and `OPSEL`) and also with the optimization solver `CPLEX`.

Based on the numerical result, we observed that our proposed relaxations, CDM still needs further improvement. It is seen by comparing CDM with `CPLEX-LPrelax` that CDM can only give better performance than `CPLEX-LPrelax` on the largest size problem. Therefore, in future study, we want to implement a sparsity structure on CDM so that it can reduce the computation time problem.

Acknowledgment

M. Y. research was partially supported by JSPS Grant-in-Aid for Scientific Research (C) No:18K11176.

Table 3: Numerical comparison for EDPs ($N = 100$)

Method	m	2θ	gap = 5%			gap = 1%		
			$\boldsymbol{g}^T\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$	time (sec)	$\boldsymbol{g}^T\boldsymbol{x}$	$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$	time (sec)
`CPLEX-default`	200	0.0258	23.19	0.02580	4.31	23.49	0.02580	13.14
`CPLEX-LPrelax`			23.52	0.02580	3.08	23.52	0.02580	3.54
`dsOpt`			23.14	0.02575	1.30	23.54	0.02580	566.89
CDM			23.53	0.02580	1.78	23.55	0.02580	2.03
`CPLEX-default`	1050	0.0539	22.53	0.05389	6.68	22.53	0.05389	3.64
`CPLEX-LPrelax`			22.55	0.05371	8.10	22.55	0.05371	5.61
`dsOpt`			21.79	0.05358	6.07	22.25	0.05382	193.08
CDM			22.49	0.05339	17.02	22.49	0.05339	15.23
`CPLEX-default`	2045	0.0628	420.04	0.06100	3.21	420.04	0.06100	3.08
`CPLEX-LPrelax`			420.79	0.06190	4.28	420.79	0.06190	3.08
`dsOpt`			419.53	0.06155	7.93	419.53	0.06155	7.96
CDM			418.67	0.06010	2.56	418.67	0.06010	2.43
`CPLEX-default`	5050	0.0994	40.63	0.09932	58.37	40.63	0.09932	54.43
`CPLEX-LPrelax`			40.56	0.09868	27.22	40.56	0.09868	19.23
`dsOpt`			40.13	0.09860	134.55	40.47	0.09936	367.29
CDM			40.28	0.09821	183.56	40.35	0.09742	197.38
`CPLEX-default`	10100	0.0610	43.79	0.06059	2720.18	44.34	0.06070	> 3 hours
`CPLEX-LPrelax`			44.43	0.06061	197.51	44.42	0.06061	216.66
`dsOpt`			43.36	0.06018	584.77	44.44	0.06100	7538.99
CDM			43.86	0.06095	948.07	44.53	0.06092	1282.72
`CPLEX-default`	15222	0.0300	436.92	0.02990	5084.69	436.92	0.02990	> 3 hours
`CPLEX-LPrelax`			423.75	0.02985	603.78	438.96	0.03000	710.21
`dsOpt`					OOM			OOM
CDM			432.13	0.02865	632.34	439.88	0.02960	448.82

References

[1] J. Ahlinder, T. Mullin, and M. Yamashita. Using semidefinite programming to optimize unequal deployment of genotypes to a clonal seed orchard. *Tree genetics & genomes*, 10(1):27–34, 2014.

[2] P. Białoń. Some variants of projection methods for large nonlinear optimization problems. *Journal of Telecommunications and Information Technology*, pages 43–49, 2003.

[3] C. C. Cockerham. Group inbreeding and coancestry. *Genetics*, 56(1):89, 1967.

[4] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339, 1986.

[5] O. Ferreira and S. Németh. How to project onto extended second order cones. *Journal of Global Optimization*, 70(4):707–718, 2018.

[6] Y. N. Kiseliov. Algorithms of projection of a point onto an ellipsoid. *Lithuanian Mathematical Journal*, 34(2):141–159, 1994.

[7] M. Lynch and B. Walsh. *Genetics and analysis of quantitative traits*. Sinauer Sunderland, MA, 1998.

[8] T. H. Meuwissen. GENCONT: an operational tool for controlling inbreeding in selection and conservation schemes. In *Proceedings of the 7th Congress on Genetics Applied to Livestock Production*, pages 19–23, 2002.

[9] T. J. Mullin. OPSEL 2.0: A computer program for optimal selection in tree breeding. *Arbetsrapport från Skogforsk Nr 954-2017, Skogforsk, Uppsala, SE*, 2017.

[10] T. J. Mullin. Popsim: a computer program for simulation of tree breeding programs over multiple generations. *Arbetsrapport från Skogforsk Nr. 984-2018, Skogforsk, Uppsala, SE*, 2018.

[11] T. J. Mullin and P. Belotti. Using branch-and-bound algorithms to optimize selection of a fixed-size breeding population under a relatedness constraint. *Tree genetics & genomes*, 12(1):4, 2016.

[12] R. Pong-Wong and J. A. Woolliams. Optimisation of contribution of candidate parents to maximise genetic gain and restricting inbreeding using semidefinite programming (open access publication). *Genetics Selection Evolution*, 39(1):3, 2007.

[13] J. P. Vielma, I. Dunning, J. Huchette, and M. Lubin. Extended formulations in mixed integer conic quadratic programming. *Mathematical Programming Computation*, 9(3):369–418, 2017.

[14] R. Wituła and D. Słota. Cardano's formula, square roots, chebyshev polynomials and radicals. *Journal of Mathematical Analysis and Applications*, 363(2):639–647, 2010.

[15] S. Wright. Coefficients of inbreeding and relationship. *The American Naturalist*, 56(645):330–338, 1922.

[16] M. Yamashita, T. J. Mullin, and S. Safarina. An efficient second-order cone programming approach for optimal selection in tree breeding. *Optimization Letters*, 12(7):1683–1697, 2018.