ARBETSRAPPORT 954-2017

OPSEL 2.0: a computer program for optimal selection in tree breeding

Dataprogram för optimalt urval i skogsträdsförädlingen

PHOTO: CURT ALMQVIST/SKOGFORSK

Tim J. Mullin

skogforsk

Sammanfattning

Skogsträdförädlare måste ofta beakta bevarandet av genetisk diversitet och samtidigt maximera selektionsvinsten. Vid urval av kloner till en förädlingspopulation måste förädlaren välja metoder som både stödjer snabba förädlingsframsteg och lämnar utrymme för framtida genetiska förbättringar i kommande generationer. För en köpare av plantagefrö gäller det att maximera värdeproduktionen på odlingslokalen samtidigt som hänsyn måste tas till genetisk diversitet i den etablerade föryngringen.

Vid ett optimalt urval undviks inte släktskap mellan utvalda träd helt och hållet, utan den genetiska vinsten kommer snarare att maximeras givet en fastställd begränsning vad gäller tillåtet genomsnittligt släktskap mellan utvalda träd. I denna rapport presenteras uppdaterad dokumentation för hjälpverktyget ”OPSEL”, som kan användas i förädlingsarbetet både vid urval av träd till förädlingspopulationer och till fröplantager. Även om urvalen till förädlingspopulationer och fröplantager verkar likvärdiga, kräver de i realiteten olika angreppssätt vid formulering av funktionen som skall optimeras och hur lösningen skall uppnås. I denna nya version av OPSEL kvarstår användandet av Mixed-Integer Linear Programming vid optimering av urval med likvärdigt genotypbidrag. För urvalssituationer med obalanserat genotypbidrag har vi istället introducerat Second-Order Cone Programming, vars algoritm genomför optimeringsberäkningen betydligt snabbare jämfört med tidigare samt genererar en pseudo-exakt heltalslösning, en nödvändighet vid planering av korsningsarbete med obalanserat urval.

OPSEL och dess stödjande program är fritt tillgängliga från Skogforsk för icke-kommersiella ändamål (på villkor fastställda under ”GNU General Public License”). Rapporten är tänkt som en vägledning vid installation av programvaran, preparering av indata samt programstyrning och resultattolkning

skogforsk

Uppsala Science Park, 751 83 Uppsala
skogforsk@skogforsk.se
skogforsk.se

 ISSN 1404-305X

Preface

This research has received funding from (1) Föreningen Skogsträdsförädling (The Tree Breeding Association, Sweden), (2) the European Union Horizon 2020 Research and Innovation Program under grant agreement no. 676876 (Project GenTree), and (3) Skogforsk (The Swedish Forestry Research Institute).

2017-08-18

Tim J. Mullin

Contents

Summary

Tree breeders must often consider conservation of genetic diversity, while at the same time maximizing response to selection. Breeders need to select breeding populations that make rapid progress in terms of genetic gain, while maintaining genetic diversity for future genetic improvement. Similarly, buyers of seed-orchard seed want maximum performance, while satisfying a restriction, sometimes legislated, on the diversity deployed to the forest.

Optimal selection will not completely avoid kinship, but rather maximize gain while imposing a constraint on average relatedness. This report provides updated documentation for the use of an integrated tool called “OPSEL”, for optimum selection in tree breeding, both for selection of breeding populations and seed orchards. While the two problems appear similar, they require rather different approaches to formulating the object function to be optimized, and the solvers required to accomplish the task. In this new version of OPSEL, we retain the use of Mixed-Integer Linear Programming to optimize selection where genotypes contribute equally. For the unequal contribution situation, we introduce a new approach with Second-Order Cone Programming, to perform the optimization much more quickly, and with the option to provide a pseudo-exact integer solution that can be used to plan crossing where parents are used unequally.

OPSEL 2.0 and its supporting software tools are freely available from Skogforsk for non-commercial purposes under general licenses. This report is intended as a guide to the installation of the software, preparation of input data, running the optimizations, and interpreting the output.

Introduction

It is common in tree breeding that selection of populations must consider conservation of genetic diversity, while at the same time attempting to maximize response to selection. It is generally recognized that one cannot to simply select the "best" trees, without also taking into account the degree of relatedness among them. Managing relatedness among selections becomes complicated as early as the first cycle of breeding, when parents, siblings and other close relatives have similar ranks. The optimal solution is not to completely avoid kinship, but rather to find the set of selections that maximizes gain under a relatedness constraint.

The issue of how to truly optimize the balance between gain and relatedness can be approached in several ways. In the context of tree breeding, the problem was formulated by Lindgren and Mullin (1997) who expressed "group merit" of a selected population as a function of average genetic value, penalized by a weight on relatedness among individuals, as expressed by their "group coancestry" (*sensu* Cockerham, 1967). While they provided a way to maximize group merit for a fixed number of selections with equal representation ("Group-Merit Selection": GMS), the constraint on relatedness is applied indirectly, so that achieving a specified level when selecting breeding populations requires a trial-and-error approach, typically with many iterations.

The special case of optimizing deployment of genotypes to a wind-pollinated seed orchard was explored by Lindgren and co-workers (Lindgren et al. 1989; Lindgren and Matheson 1986) who suggested that the optimum relationship between candidate breeding value and their contribution to the population would be linear. The application of "linear deployment" is only possible when the candidates are unrelated, as might be the case when selecting "backward" on progeny tested plus-tree candidates. If the candidate pool includes relatives, Bondesson and Lindgren (1993) suggested that a more complex formulation would be required, perhaps using a Lagrange function. In more recent work, Lindgren's group used the Microsoft Excel add-in tool, "Solver", to maximize gain by Linear Programming under a Lagrange equality constraint on relatedness (Danusevičius and Lindgren 2008; Lindgren et al. 2009).

Other methods to optimize the gain-diversity balance have been proposed. These include the Optimum Contribution (OC) algorithm based on Lagrangian multipliers (LM), originally proposed for selection in dairy cattle (Hinrichs et al. 2006; Meuwissen 1997) and recommended for application to forest trees by Kerr et al. (1998). The application of OC has been demonstrated in the management of breeding (Hallander and Waldmann 2009a) and in the optimal selection of Scots pine parents for a seed orchard with unequal numbers of ramets (Hallander and Waldmann 2009b).

There are some serious drawbacks with the LM method. First, by removing candidates or fixing their contributions to zero and re-optimizing with a new subset of candidates, it is possible that the true optimum solution is bypassed in the iterative procedure. Second, there is no restriction on the maximum allowed contribution of any particular candidate. This means that *ad-hoc* manipulations of the final solution may be required to satisfy other operational constraints on its implementation. For example, in forest tree breeding, one major constraint for the establishment of grafted seed orchards is the number of scions that can be collected from a given genotype. Finally, the objective of selection can also be to form a breeding population of fixed size, whose members will contribute equally to a recruitment population; optimal solutions that satisfy such constraints are not possible with the LM approach.

This report documents the use of version 2 of an integrated tool known as "OPSEL" for optimum selection in tree breeding, both for selection of seed orchard and breeding populations. OPSEL was first introduced in 2014 as version 1.0 (Mullin 2014), but several improvements have been made since then in both the optimization techniques and usability of the software. While this documentation is intended to be complete, we have assumed that the user is familiar with the terminology and procedures commonly used in genetic evaluation, selection, and measures of relatedness. Some users, particularly those without formal training in quantitative genetics, may find it useful to review concepts in an introductory textbook on forest tree breeding or crop improvement (e.g., White et al. 2007). Similarly, the guide assumes a general familiarity with basic procedures and configuration of PC-compatible personal computers. Details on these procedures can be found in the original hardware and operating system documentation and in a host of other references available in any computer store or library.

Background theory

In OPSEL, the two variations on the optimal selection problem are solved by means of mathematical programming techniques, but the approaches are quite different. What follows is a brief description of each approach; for detailed theoretical development, the reader is referred to Yamashita et al. (2017) and to Mullin and Belotti (2016), respectively.

OPTIMIZING UNEQUAL CONTRIBUTIONS TO A SELECTED POPULATION

OPSEL uses an approach based on second-order cone programming (SOCP) to optimize objective functions where genotypes are allowed to contribute unequally. SOCP is a convex optimization that maximizes a linear objective function over second-order cone constraints. It can be considered a special case of Semi-Definite Programming (SDP), an approach first proposed for optimal selection by Pong-Wong and Woolliams (2007) and implemented by Skogforsk (Ahlinder et al. 2014; Mullin 2014). SOCP can be efficiently solved with interior-point methods in a similar way to SDP and a freeware solver package known as ECOS is available (Domahidi et al. 2013).

Starting with the original SDP formulation, Yamashita et al. (2017) developed an SOCP adaptation could be solved more quickly. They made several improvements, including exploiting the sparsity embedded in the numerator relationship matrix and integrating the Quaas-Henderson algorithm (Henderson 1976; Quaas 1976) that made processing the matrix and solution of very large candidate lists many times faster.

To formulate the selection with unequal contributions, we first consider selecting a cohort from a complex pedigree, containing totally Z genotypes that are to contribute their genes in optimal proportions. Our goal is to maximize the expected genetic merit of contributions from the selected cohort, given by $\boldsymbol{g}^T\boldsymbol{x}$ where: the estimated breeding values (EBV) for all pedigree members are found in vector $\boldsymbol{g}$, of size $Z\times1$; and the contribution of genes as a proportion is denoted $\boldsymbol{x}$, also of size $Z\times1$. In our problem, the decision variable is $\boldsymbol{x}$, where $1\geq x_i \geq 0$, and the sum of all contributions from the selected cohort equals unity ($\Sigma_1^Z x_i=1$).

Meuwissen (1997) recommended using a quadratic constraint based on the numerator relationship matrix $\boldsymbol{A}$, to impose a restriction on the relatedness or group coancestry, θ, of the selected cohort, specified as $\theta \geq \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}/2$, where the additive or numerator relationship

matrix of the pedigree is denoted $\boldsymbol{A}$, of size $Z \times Z$. The maximum and minimum contributions that a particular individual can make are denoted $\boldsymbol{u}$ (upper limit) and $\boldsymbol{l}$ (lower limit), respectively, with both vectors of size $Z \times 1$. Here, if pedigree member i is not itself a candidate for selection (e.g., not physically available for use), the corresponding maximum contribution is set to zero (i.e., $u_i = 0$). Similarly, while the minimum number of contributions for a genotype might normally be zero, there may be times when prior investments or operational requirements might motivate setting l_i to a specific value greater than zero, provided that $u_i \geq l_i$.

Our general framework for optimizing genetic contributions can then be formulated:

$$\text{Maximize:} \quad \boldsymbol{g}^T\boldsymbol{x} \qquad (1a)$$

$$\text{Subject to:} \quad \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}/2 \leq \theta \qquad (1b)$$

$$\boldsymbol{e}^T\boldsymbol{x} = 1 \qquad (1c)$$

$$\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \qquad (1d)$$

where $\boldsymbol{e}$ is a vector of size $Z \times 1$ containing 1's.

Yamashita et al. (2017) recognized the vast difference in density between the numerator relationship matrix $\boldsymbol{A}$ and its inverse $\boldsymbol{A}^{-1}$. Figure 2 illustrates the non-zero elements of $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$, for a pedigree of 10 100 records. Clearly, $\boldsymbol{A}^{-1}$ is much sparser, having 56 092 non-zero elements, compared with 56 754 980 in $\boldsymbol{A}$; their relative density of the full 10 000 x 10 000 matrix is 0.0549% and 55.6%, respectively.

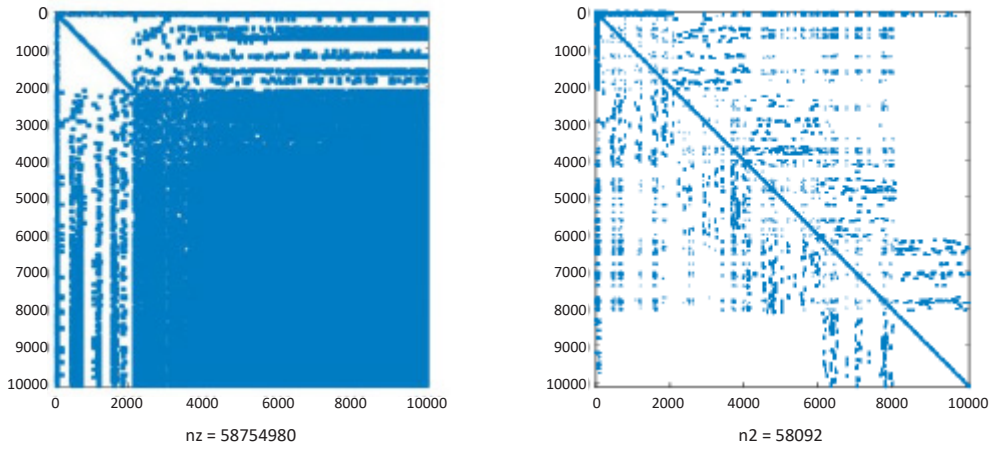

Figure 2. Non-zero elements indicated in blue, for $\boldsymbol{A}$ (left) and $\boldsymbol{A}^{-1}$ (right) with example pedigree of size 10 100.

A key step in the approach is the introduction of a new variable, $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$. We can then replace our decision variable $\boldsymbol{x}$ by $\boldsymbol{A}^{-1}\boldsymbol{y}$ to give an equivalent optimization problem:

$$\text{Maximize:} \quad (\boldsymbol{A}^{-1}\boldsymbol{g})^T\boldsymbol{y} \qquad (2a)$$

$$\text{Subject to:} \quad (\boldsymbol{A}^{-1}\boldsymbol{e})^T\boldsymbol{y} = 1 \qquad (2b)$$

$$\boldsymbol{y}^T\boldsymbol{A}^{-1}\boldsymbol{y}/2 \leq \theta \qquad (2c)$$

$$\boldsymbol{l} \leq \boldsymbol{A}^{-1}\boldsymbol{y} \leq \boldsymbol{u} \qquad (2d)$$

OPSEL uses an open-source solver to optimize the SOCP, known as ECOS (Domahidi et al. 2013). While SOCP is very flexible, with flexibility comes complexity and room for error in setting up the program properly and interpreting the output. OPSEL simplifies the task, receiving input regarding the total number of contributions required, the constraint on relatedness (as group coancestry or Status Number), whether a minimum is to be imposed on individual genotype contributions (i.e., $\boldsymbol{u} \neq \mathbf{0}$), and the name of a file containing the EBVs of all candidate genotypes, their maximum (and optionally minimum) frequency in the selected group, as well as the complete pedigree including ancestors. These data are used to prepare the SOCP for solving by ECOS. Once ECOS has completed its work, OPSEL then reads the ECOS output and generates a file with the original data, as submitted in the pedigree file, with additional columns specifying the optimum contribution as a proportion and as an integer number of ramets for each genotype.

BRANCH-AND-BOUND OPTIMIZATION OF FIXED-SIZE BREEDING POPULATIONS

When selecting a fixed-size breeding population, we are searching for a cohort of exactly N individual genotypes from a complex pedigree containing totally Z members that are to contribute their genes as breeding parents in equal proportions. The object is to maximize the expected genetic merit of contributions from the selected cohort, given by $\boldsymbol{g}^T\boldsymbol{x}$ where the estimated breeding values (EBV) for all pedigree members are found in vector $\boldsymbol{g}$, of size $Z\times1$; the contribution of genes as a proportion is denoted $\boldsymbol{x}$, of size ($Z\times1$), where $c_i \in \{0,1/N\}$ so that each individual in the selected cohort contributes exactly $1/N$ to the future gene pool, and if not selected contributes zero; and the sum of all contributions must equal unity ($\Sigma_1^Z x_i=1$).

To control relatedness in the breeding population, we wish to impose a constraint on the group coancestry, θ, of the selected cohort of trees, such that $\theta \geq \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}/2$, where the additive or numerator relationship matrix of the pedigree is denoted $\boldsymbol{A}$, of size $Z\times Z$.

For the variable $\boldsymbol{x} \in \mathbb{R}^Z$, the problem can be expressed in a form that uses binary variables, i.e., variables that can take only values 0 or 1. This formulation is suited to a particular class of solution algorithms called *branch-and-bound*, which are very well known in the optimization literature (Land and Doig 1960). A new variable $\boldsymbol{y}$ is defined such that $\boldsymbol{y}=N\boldsymbol{x}$ and the problem respecified to minimize the objective function:

$$\text{Maximize:} \quad -\boldsymbol{y}^T\boldsymbol{g} \tag{3a}$$

$$\text{Subject to:} \quad \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y} \leq 2\,\theta N^2 \tag{3b}$$

$$\boldsymbol{y}^T\mathbf{1} = N \tag{3c}$$

$$y_i \in \{0,1\} \quad (i = 1, \ldots, Z) \tag{3d}$$

The advantage in using binary variables lies in the availability of mixed-integer linear programming (MILP) software packages, such as CBC[1] (COIN-OR branch-and-cut) that accept models with optimization variables that are discrete in nature and take on integer values. The solver used by OPSEL to optimize selection of fixed-size breeding populations, dsOpt[2], was custom written by Dr. Pietro Belotti using the open-source MILP routines available in CBC.

[1] The CBC libraries are written in C++ and are available as open-source code at http://coin-or.org/

[2] The current version of dsOpt.exe is available from the author

OPSEL accepts input regarding the total number of genotypes to select for the breeding population, the constraint on relatedness (as group coancestry or Status Number), and the name of a text file containing the EBVs of all candidate genotypes, as well as the complete pedigree including ancestors. These data are used to prepare the input file for solving by dsOpt. If the solver completes successfully, OPSEL will read the solution output and generate a file with the original data, as submitted in the pedigree file, and with an additional column specifying the *N* selected genotypes with "1" and the remainder as "0".

Hardware/software requirements for OPSEL

OPSEL is written as a 64-bit program for PC-compatible computers, but will run on any computer under the Microsoft Windows Operating System from version "XP" to present, provided a 64-bit version is installed. OPSEL will also require access to dsOpt.exe, filesocp.exe, and/or ecos-wrapper.exe, depending on the type of selection required. A text editor is required to prepare input files and interpret some files created by OPSEL. While not absolutely necessary, most users will also find it helpful to have access to an Excel-compatible spreadsheet program for editing of pedigree and breeding-value data, converting these to text or comma-separated files for input to OPSEL, and manipulation of files created when OPSEL is run.

Becoming a registered user of OPSEL

OPSEL and its supporting tools are available from Skogforsk at www.skogforsk.se/opsel. A download link is provided to supply users with a zipped package of components and examples. While distribution of OPSEL is not restricted, registration is strongly encouraged. Periodic updates, bug fixes and lists of known issues will be circulated to all registered users as they are released.

A note on copyright, licensing and warranty

While copyright for OPSEL is held by Skogforsk, distribution is without fee for research and non-commercial use under general public licenses. Copies of these licenses must accompany distribution of OPSEL and work derived from it. In addition, there is ABSOLUTELY NO WARRANTY expressed or implied, and the user bears all the risk when using OPSEL. Users should be aware that OPSEL uses open-source software to perform some of its functions, including ECOS, CBC and dsOpt – copies of licenses for these products are distributed with the OPSEL package. In accepting the conditions of use on the opening screen, the user also agrees to the conditions of use for these components – please read these!

Distribution of OPSEL

OPSEL will usually be distributed as a "zip" file. The following files are provided in the package:

- opsel.exe – the executable binary for running OPSEL;
- filesocp.exe – prepares the data for the unequal contributions solver, ECOS;
- ecos-wrapper.exe – the ECOS solver using Second-Order Cone Programming;
- dsOpt.exe – the branch-and-bound solver for MILP, used by OPSEL;
- GNUv2_license.txt – GNU General Public License, Version 2, applies to OPSEL and ECOS;
- Eclipse_license.txt – Eclipse General Public License, Version 1, applies to CBC and dsOpt;
- SO_block.csv, SO_unequal.csv and BP_T11.csv – the example datasets discussed in this document, formatted as comma-separated values;
- SO_block.ctl, SO_unequal.ctl and BP_T11.ctl – control parameter files for running examples through the command line in Command Prompt window;
- OPSEL_Userguide_v2.pdf – this document; and
- Release Notes xx-xx-xxx.pdf – fixes and upgrades since publication of the Userguide

Installing OPSEL

The following procedure is recommended for installing the program, although file locations could be changed, once successful operation of the program has been verified.

1. Extract the zip file to a convenient folder on your machine, such as C:\OPSEL\. The "OPSEL.exe", "dsOpt.exe", "filesocp.exe", and "ecos-wrapper.exe" files may have had the ".exe" extensions stripped, to enable the sending of the zip file through institution firewalls; if so, you will need to restore the names by adding ".exe" to distributed file names. We recommend that all files in the package be placed in the folder where optimization is to occur, i.e., the same file as the user-prepared input files.
2. Test your installation by working through the optimization of example data sets provided with the package and detailed in the next sections.

Running the examples provided with OPSEL

THE CANDIDATE FILE

All optimization jobs in OPSEL require a "candidate file". The program is distributed with example candidate files, formatted as "comma-separated values" (.csv) files for optimized selection of populations with both equal and unequal contributions. The files describe the pedigree of the selection candidates and their ancestors, using the "Me-Mum-Dad" standard that will be familiar to most users of pedigree analysis tools and databases, and appends extra fields to describe the genetic values and frequency constraints in the selected population.

Open this file in a text editor or in Excel and note the following structure:

1. The first line of the file will be skipped, so can contain column headers to document the file.
2. Each subsequent line describes an individual with its ID, those of its Female and Male parents, its estimated breeding (or genetic) value, and finally the maximum (and optionally for unequal contributions selection the minimum) number of times that the individual can occur in the selected population.

The pedigree follows the usual rule that parents **must** precede their progeny, and that all ancestors must be listed. Unknown Female and Male IDs are indicated with a 0 (zero). All other pedigree IDs are alphanumeric labels up to 256 characters in length. As spaces, commas and tab characters are used to delimit fields, these characters are not permitted in any field. Small examples of candidate files for breeding populations and seed orchard selection are illustrated in Tables 1 and 2, respectively.

Table 1. Small example of a candidate file for optimum selection of a fixed-size breeding population. The Max column will always contain either the value 1, indicating that the individual is a candidate, or 0, indicating that the individual is not a candidate, but is included as an ancestor to another individual (e.g., individual 2).

Indiv.	Female	Male	EBV	Max
1	0	0	90.4	1
2	0	0	0.0	0
3	0	0	75.5	1
4	1	2	100.4	1
5	1	3	130.5	1
6	2	4	150.3	1
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

Table 2. Small example of a candidate file for optimum selection of a seed orchard. The Max column will always contain a value >= 1 (indicating that the individual is a candidate up to the specified maximum frequency) or = 0, indicating that the individual is not a candidate, but is included as an ancestor to another individual (e.g., individual B). The last column is optional if a minimum frequency is to be specified; in this case individual C is to be included in the solution with a frequency of at least 5.

Indiv.	Female	Male	EBV	Max	Min.
A	0	0	200.4	25	0
B	0	0	0.0	0	0
C	0	0	180.4	25	5
D	A	B	223.5	10	0
E	A	C	165.3	10	0
F	B	C	225.7	10	0
G	A	F	300.2	15	0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

The 4th field for each record contains the genetic value (EBV) of the individual as a real number. The values in this column are ignored when the frequency constraint specified in field 5 is zero.

The 5th field for each record gives the number of times the individual is to be used in the selected population, as a positive integer. When selecting a population with equal contributions, this value should be 0 or 1 (see SO_block.csv); whereas an unequal contributions problem would specify the maximum contribution of the genotype, say as number of ramets in an orchard or maximum number of parent contributions in a crossing program (see SO_unequal.csv). If the individual is not regarded as a selection candidate (an ancestor or other relative), the value in the Max field should always be 0 (zero). Note, in SO_unequal.csv, we have specified a maximum of 2800 when the candidate is a founder (a large, older individual in an archive, capable of providing many scions for orchard grafting), whereas the F1 candidates are constrained to a maximum of 50 (as these individuals are much younger with smaller crowns).

The 6th field is optional and used when a minimum frequency is to be considered in the optimization of an unequal contribution problem. When used, the individual genotype will occur in the solution with a frequency greater than or equal to this minimum.

When preparing the input file, the user will normally use a spreadsheet program to organize and create the various fields. Simply save the spreadsheet as a comma-separated value (.csv) or tab-delimited (.txt) file before submitting to OPSEL. When submitted to OPSEL, any additional columns will be ignored and not transferred to the output file.

INTERACTIVE DIALOGUE OR COMMAND LINE

OPSEL is a Windows program. It can be started by double-clicking on the program icon (opsel.exe), which will take the user through an online dialogue window to declare the path to the candidate file, define the constraints and initiate the optimization. Alternatively, the user can run the program directly from the command-line in the Command Prompt window, declaring a control file as a parameter that defines the constraints and declares the location of the candidate file. The following sections describe the running of each example through both the online dialogue and via the command line.

OPTIMIZING SELECTION WITH EQUAL CONTRIBUTIONS – SO_BLOCK.CSV

Prepare the candidate list

In this example, we use a standard set of data available online from a clonally replicated population of loblolly pine in the southeastern USA (Resende et al. 2012) to illustrate the assembly of a 12-clone block that will be replicated as randomized complete blocks across a seed orchard, so that each genotype contributes equally. The population was derived by controlled crossing among 32 selected parents, consisting of 22 field-selected F0 plus-trees and ten selected F1 progeny. These parents were crossed in a partial-diallel mating design, and the 861 progeny propagated for field testing as rooted cuttings cuttings (Baltunis et al. 2007a, b). For this example, we declare that the 12-tree blocks replicated across the orchard must have a status number $N_S \geq 8$ (group coancestry $\theta \leq 0.0625$).

OPSEL will submit the constraint data and candidate list to dsOpt.exe and interpret the output. The time required for the optimization is difficult to predict, but it is recom-

mended to truncate the candidate list in a sensible way, in order to reduce the size of the problem and decrease the time required to find a solution. This particular example is already a short list of candidates and runs quite quickly, so truncation is not required. For longer pedigrees, truncation to speed up execution could be done through a combination of techniques such as:

1. Limiting the number of full-sibs, such that the candidate list contains only the best sibs in each family; and/or
2. Apply a breeding value threshold, below which candidates are dropped from the list (their Max value set to zero).

The candidate file must conform to the format and sort order described earlier, and must contain the following 5 columns (other columns in this file are ignored):

Indiv	*Female*	*Male*	*EBV*	*Max*

The estimated breeding values (EBVs) would normally be extracted, together with the appropriate pedigree entries, from a third-party database, such as DATAPLAN®.

Running SO_block.csv through the online dialogue

The following is a detailed description of an OPSEL job to optimize selection with equal contributions, illustrated by processing SO_block.csv:

1. Start OPSEL by double-clicking on OPSEL.exe
2. Accept the conditions of use on the splash screen by typing "y", then pressing Enter.
3. The description on the next screen is a summary of the information in this user guide on how to prepare the pedigree file. Note that the "root" name of the input file, in this case "SO_block", is used by OPSEL when referencing all other files, using different extensions appended to the root.
4. Enter the name of the candidate file containing the pedigree and breeding-value data: SO_block.csv.
5. Specify that OPSEL is to use the "equal contributions" option by entering "0".
6. OPSEL reports the current working directory and asks if the MILP solver "dsOpt.exe" is in this folder. If so, just answer "y", otherwise enter "n" and OPSEL will ask you to specify the full path location of dsOpt.exe.
7. Specify the desired census size of the selected population, the number of genotypes in the breeding population or the orchard block, in this case 50.
8. OPSEL now needs to know the constraint on diversity of the selected population. This can be specified either as group coancestry, or as Status Number. In this case, specify the constraint on maximum group coancestry as 0.0625 (which is equivalent to a Status Number of 1/(2 * 0.0625) = 8.0).

9. The solver will work on the problem until it finds the optimum solution, or until the time threshold is reached. The default threshold is 7 200 seconds and is appropriate for this example.

10. The branch-and-bound search is limited by default to 10 000 nodes and is appropriate for this example.

11. OPSEL will continue the optimization until the solution is within a specified "gap" from the theoretical optimum. A smaller gap will require more time to execute, but there is no guarantee that a better solution will be found. The default of 0.5% is appropriate for this example.

12. By default, OPSEL will store the solution in a text file, but you can specify comma-separated values here, if you wish; doing so will make it easier to read the solution with Excel.

13. OPSEL now proceeds to formulate the MILP and to build the input for dsOpt. A call to dsOpt.exe is made, and you will see its progress in a separate DOS command-line window.

14. Once dsOpt has completed the optimization, control is returned to OPSEL which interprets the output. A summary of the solution statistics is given. OPSEL will give you the option to delete the temporary work files and will then ask if you want to store your responses to the online dialogue to run later from the command line; provide a filename, such as SO_block.ctl.

15. Check the folder where you started OPSEL and you will find the following additional files:

    SO_block_in.txt – the input prepared by OPSEL for dsOpt.
    SO_block_in.sol – the raw solution generated by dsOpt.
    SO_block_solution.csv or SO_block_solution.txt – the solution combined with the initial candidate list, in either comma-separated format or as a text file, as specified by the user. The columns in the file are:

    - Individual – ID of the individual genotype.
    - Female – ID of the individual's mother.
    - Male – ID of the individual's father.
    - EBV – the genetic value of the individual (supplied by user).
    - Freq – selected individuals are indicated by 1, all others by 0.
    - Max – the constraint on the maximum contribution for each individual (either 0 or 1).

    SO_block_log.txt – a log file that contains summary statistics about the problem and its solution.

Running SO_block.csv from the command line

OPSEL can also be run from the command line by specifying the various job parameters in a separate text file. The simplest way to do this is to run through the online dialogue and then save the parameters in a text file, such as (see step 14 above). A sample version of SO_block.ctl is provided in the OPSEL distribution package and can be modified with any text editor.

Open a Command Prompt window -- usually found in the Accessories folder. Navigate to the location of opsel.exe and the parameter file. Launch OPSEL by entering opsel.exe, followed by the path to the parameter file, for example:

C:\opsel>opsel.exe SO_block.ctl

OPTIMIZING SELECTION OF A SEED ORCHARD WITH UNEQUAL CONTRIBUTIONS – SO_UNEQUAL.CSV

In this example, based on the Dag Lindgren Seed Orchard in northern Sweden, we use data that were available from the Swedish pine breeding program to select a seed orchard where the frequency of individual genotypes optimizes the number of ramets of each. The orchard site has 2800 planting positions. The candidate population consists of founder plus-trees and their F1 progeny, for which EBVs are available.

Prepare the candidate list

Not all candidates are capable of producing large numbers of scions, so there are limitations on the number of grafts that can be made for any given individual: some founders no longer exist, so they cannot be used in the orchard; the F1 progeny are still rather small and can produce perhaps no more than 50 grafts; while other founders are large and well represented in existing orchards and archives, and so can contribute virtually unlimited numbers of grafts. The 5th column of the candidate list file contains the constraint on the maximum number of contributions permitted for each individual; in this case 0, 50 or 2800, as appropriate.

We can also "force" an individual to be represented a minimum number of times. For this example, we have identified individual 294 as a key genotype that we plan to use in the future as a reference pollen source, so we definitely want to include in this orchard. In the optional 6th column of the candidate list, we specify a minimum contribution from this individual of 25 ramets. There is no such minimum constraint for the other individuals, so their minima are all set to 0.

The candidate file must conform to the format and sort order described earlier, and must contain the following 5 columns (the 6th column "*Min*" is optional):

Indiv	*Female*	*Male*	*EBV*	*Max*	*Min*

The estimated breeding values (EBVs) would normally be extracted, together with the appropriate pedigree entries, from a third-party database, such as DATAPLAN®.

Running SO_unequal.csv through the online dialogue

The following is a detailed description of an OPSEL job selecting the genotypes and optimizing their contributions to a 2800-tree seed orchard, illustrated by processing SO_unequal.csv:

1. Start OPSEL by double-clicking on OPSEL.exe
2. Accept the conditions of use on the splash screen by typing "y", then pressing Enter.
3. The description of OPSEL on the next screen is a summary of the information in this user guide on how to prepare the pedigree file. Note that the "root" name of the input file, in this case "SO_unequal", is used by OPSEL when referencing all other files, using different extensions appended to the root.
4. Type the candidate-list file name containing the pedigree and breeding-value data: SO_unequal.csv, and press enter.
5. In very special cases, you may know that the pedigree is properly sorted and that the identities are integers starting at 1. In all other cases, you will need to complete pedigree recoding and verification. So, answer "n" to the question on whether or not to skip this step.
6. Specify that OPSEL is to use the "unequal contributions" option by entering 1.
7. OPSEL reports the current working directory and asks if the SOCP solver files are in this folder. If so, just answer "y", otherwise enter "n" and OPSEL will ask you to specify the full path location.
8. Specify the desired census size of the selected population, in this case 2800.
9. OPSEL now needs to know the constraint on diversity of the selected population. This can be specified either as group coancestry, or as Status Number. In this case, let's answer "y", then specify our constraint as Status Number 14 (which is equivalent to a group coancestry of 1/(2 * 14) = 0.0357).
10. OPSEL now asks if the input file contains a specification for a minimum constraint (column 6). In this example, we have set a minimum of 25 ramets from individual 294, so here we answer "y".
11. By default, the optimum contribution proportions will be multiplied by the desired size of the orchard (2800 ramets), to give an integer solution. Due to rounding, the sum of the integer contributions may not sum exactly to 2800. If an exact total orchard size is important, the user can force OPSEL to make adjustments as required, using one of two available algorithms. For this example, let's accept the default rounding to see how close we get; enter 0 (zero).
12. By default, OPSEL will store the solution in a text file, but you can specify comma-separated values here, if you wish. Doing so will make it easier to read the solution with Excel; enter "y".
13. OPSEL now checks to make sure that the pedigree specification is legal (all individuals accounted for, parents precede progeny, etc.).

14. The input file for the SOCP solver is prepared by the SOCP wrapper and submitted externally to ECOS. You will see its progress in a separate DOS command-line window (although execution is VERY fast!).

15. Once ECOS has completed the SOCP optimization, control is returned to OPSEL, which interprets the output and provides an on-screen summary. If the run fails, OPSEL should report an error code and brief message to assist in debugging the problem.

16. OPSEL now asks permission before deleting any of the temporary work files. Unless these are required for debugging, you can go ahead to delete these (answer "y").

17. OPSEL will ask you if you want to store your responses to the online dialogue to run later from the command line; answer "y" and provide a filename, such as SO_unequal.ctl.

18. Check the folder where you started OPSEL and you will find the following additional files:

    SO_unequal_solution.csv or SO_unequal_solution.txt
    – the solution combined with the initial candidate list, in either comma-separated format or as a text file, as specified by the user. The columns in the file are:

    Individual – ID of the individual genotype.
    Female – ID of the individual's mother.
    Male – ID of the individual's father.
    EBV – the genetic value of the individual (supplied by user)
    Freq – the contributions from selected individuals are indicated by whole numbers
    Proportion – the contribution from each individual indicated as a proportion.
    Max – the constraint on the maximum contribution from each individual.
    Min – (if requested) the constraint on the minimum contribution from each individual.

    SO_unequal_log.txt – a log file that contains summary statistics about the problem and its solution.
    ecos-info.txt – a log file produced by the ECOS solver giving specifics on the optimization.

Running SO_unequal.csv from the command line

OPSEL can also be run from the command line by specifying the various job parameters in a separate text file. The simplest way to do this is to run through online dialogue and save the parameters in a text file, such as (see step 18 above). A sample version of SO_unequal.ctl is provided in the OPSEL distribution package and can be modified with any text editor.

Open a Command Prompt window – usually found in the Accessories folder. Navigate to the location of opsel.exe and the parameter file. Launch OPSEL by entering opsel.exe, followed by the path to the parameter file, for example:

C:\opsel>opsel.exe SO_unequal.ctl

OPTIMIZING SELECTION OF GENOTYPES CONTRIBUTING UNEQUALLY TO A CROSSING PROGRAM – BP_T11.CSV

In general, allowing genotypes to mate unequally will allow the optimization to find a better solution for a breeding population with greater average breeding value. In this example, taken from Mullin and Persson (2017), we wish to select a group of mates to produce 100 crosses for the 3rd cycle of breeding of a Scots pine breeding population. Since 2 parents are required for each cross, we need to identify a total of 200 parent contributions to complete the crossing program.

Prepare the candidate list

The candidate file must be conform to the format and sort order described earlier, and must contain the following 5 columns (the 6th column "Min" is optional):

Indiv	*Female*	*Male*	*EBV*	*Max*	*Min*

The estimated breeding values (EBVs) would normally be extracted, together with the appropriate pedigree entries, from a third-party database, such as DATAPLAN®. The first three columns list the usual Me-Mum-Dad identification data, and the fourth column lists the EBV.

There may be reasons to exclude some records as candidates; the tree may no longer exist, or is otherwise inaccessible. In our example, all individuals across all generations are considered as candidates. Operationally, there may be reasons to restrict the number of crosses made with a given parent, but in this case, we will allow the optimization to find the absolute best contributions, so we set the maximum to the number of crosses, 100.

There is no reason to make much effort to "truncate" the candidate list, as the optimization by SOCP goes very quickly, even with many thousands of candidates. The candidate file BP_T11.csv includes over 48 000 identities, spanning 3 cycles of breeding, of which about 35 000 are available as selection candidates. Despite the large size of the candidate file, execution requires only a few minutes.

If some relevant crossing has already been done, we might "force" a candidate to be represented a specified minimum number of times; this will ensure that the previous crossing effort is represented in the new solution. For this example, we have not specified any minima, so the 6th column is left blank.

Running BP_T11.csv through the online dialogue

The following is a detailed description of an OPSEL job selecting the genotypes and optimizing their contributions to a 100-cross (200 parent contribution) mating program, illustrated by processing BP_T11.csv:

1. Start OPSEL by double-clicking on OPSEL.exe
2. Accept the conditions of use on the splash screen by typing "y", then pressing Enter.
3. The description of OPSEL on the next screen is a summary of the information in this user guide on how to prepare the pedigree file. Note that the "root" name of the input file, in this case "BP_T11", is used by OPSEL when referencing all other files, using different extensions appended to the root.

4. Type the candidate-list file name containing the pedigree and breeding-value data: BP_T11.csv, and press enter.

5. In very special cases, you may know that the pedigree is properly sorted and that the identities are integers starting at 1. In all other cases, you will need to complete pedigree recoding and verification. So, answer "n" to the question on whether or not to skip this step.

6. Specify that OPSEL is to use the "unequal contributions" option by entering 1.

7. OPSEL reports the current working directory and asks if the SOCP solver files are in this folder. If so, just answer "y", otherwise enter "n" and OPSEL will ask you to specify the full path location.

8. Specify the desired census size of the selected population. Here, to assemble parent contributions for a 100-cross mating design, we require 100 x 2 = 200 total contributions.

9. OPSEL now needs to know the constraint on diversity of the selected population. This can be specified either as group coancestry, or as Status Number. The specification of an appropriate level of diversity is discussed in some detail by Mullin and Persson (2017). In this case, let's answer "y", then specify our constraint as Status Number 20 (which is equivalent to a group coancestry of 1/(2 * 20) = 0.025).

10. OPSEL now asks if the input file contains a specification for a minimum constraint (column 6). As no minima are used in this example, we answer "n".

11. By default, the optimum contribution proportions will be multiplied by the desired total number of contributions (200 parent contributions), to give an integer solution. Due to rounding, the sum of the calculated integer contributions may not sum exactly to 200, without application of an adjustment during the rounding. OPSEL has two algorithms available for this adjustment:

    Method 1. Adjust the rounding point up or down from 0.5, as required; or

    Method 2. Add or subtract, as required, from those genotypes with the highest contributions.

    You can of course try either method – both will produce an exact total of 200 contributions, but will likely violate the object function very slightly. For this example, let's enter "1".

12. By default, OPSEL will store the solution in a text file, but you can specify comma-separated values here, if you wish. Doing so will make it easier to read the solution with Excel; enter "y".

13. OPSEL now checks to make sure that the pedigree specification is legal (all individuals accounted for, parents precede progeny, etc.) and recodes the identities in the pedigree.

14. The input file for the SOCP solver is prepared by the SOCP wrapper and submitted externally to ECOS. You will see its progress in a separate DOS command-line window.

15. Once ECOS has completed the SOCP optimization, control is returned to OPSEL, which interprets the output and provides an on-screen summary. If the run fails, OPSEL should report an error code and brief message to assist in debugging the problem.

16. OPSEL now asks permission before deleting any of the temporary work files. Unless these are required for debugging, you can go ahead to delete these (answer "y").

17. OPSEL will ask you if you want to store your responses to the online dialogue to run later from the command line; answer "y" and provide a filename, such as BP_T11.ctl.

18. Check the folder where you started OPSEL and you will find the following additional files:

    BP_T11_solution.csv or BP_T11_solution.txt – the solution combined with the initial candidate list, in either comma-separated format or as a text file, as specified by the user.
    This file can be used directly as input to XDesign for automated generation of a mating design that follows positive-assortative or random assortment mates, while avoiding excessive relatedness between crossing mates (Mullin 2017).
    The columns in the file are:

    Individual – ID of the individual genotype.
    Female – ID of the individual's mother.
    Male – ID of the individual's father
    EBV – the genetic value of the individual (supplied by user)
    Freq – the contributions from selected individuals are indicated by whole numbers
    Proportion – the contribution from each individual indicated as a proportion.
    Max – the constraint on the maximum contribution from each individual.
    Min – (if requested) the constraint on the minimum contribution from each individual.

    BP_T11_log.txt – a log file that contains summary statistics about the problem and its solution.

    ecos-info.txt – a log file produced by the ECOS solver giving specifics on the optimization.

Running BP_T11.csv from the command line

As for the earlier examples, OPSEL can be run from the command line by specifying the various job parameters in a separate text file. A sample version of BP_T11.ctl is provided in the OPSEL distribution package and can be modified with any text editor.

Open a Command Prompt window – usually found in the Accessories folder. Navigate to the location of opsel.exe and the parameter file. Launch OPSEL by entering opsel.exe, followed by the path to the parameter file, for example:

C:\opsel>opsel.exe BP_T11.ctl

When things go wrong!

The user should be aware that OPSEL is in a continuing state of development. While it has already seen considerable operational application, the limitations have not been thoroughly explored, as this will happen naturally as users attempt to apply it. Unexpected things can happen! OPSEL may provide messages that give some information on failed runs or problems with data input. If you cannot rectify a problem, please note the messages received and send your input file and problem parameters (census size and constraint on group coancestry) to: tim.mullin@skogforsk.se

Acknowledgements

I owe a debt to my coauthors of scientific papers that form the backbone of the selection algorithms used in OPSEL, namely, Professor Makoto Yamashita, Dr. Pietro Belotti, and Ms. Sena Safarina. Also acknowledged are the important contributions of Drs. Torgny Persson and Curt Almqvist (Skogforsk, Sävar and Uppsala, respectively). Funding from (1) Föreningen Skogsträdsförädling (The Tree Breeding Association, Sweden), (2) the European Union Horizon 2020 Research and Innovation Program under grant agreement no. 676876 (Project GenTree), and (3) Skogforsk (The Swedish Forestry Research Institute). All trademark terms are the property of their respective owners.

References

Ahlinder, J., Mullin, T.J., and Yamashita, M. 2014. Using semidefinite programming to optimize unequal deployment of genotypes to a clonal seed orchard. Tree Genet. Genom. 10: 27-34. doi: DOI 10.1007/s11295-013-0659-z.

Baltunis, B., Huber, D., White, T., Goldfarb, B., and Stelzer, H. 2007a. Genetic gain from selection for rooting ability and early growth in vegetatively propagated clones of loblolly pine. Tree Genet. Genom. 3(3): 227-238. doi: 10.1007/s11295-006-0058-9.

Baltunis, B.S., Huber, D.A., White, T.L., Goldfarb, B., and Stelzer, H.E. 2007b. Genetic analysis of early field growth of loblolly pine clones and seedlings from the same full-sib families. Can. J. For. Res. 37(1): 195-205. doi: 10.1139/x06-203.

Bondesson, L., and Lindgren, D. 1993. Optimal utilization of clones and genetic thinning of seed orchards. Silvae Genet. 42(4-5): 157-163.

Cockerham, C.C. 1967. Group inbreeding and coancestry. Genetics, 56: 89-104.

Danusevičius, D., and Lindgren, D. 2008. Strategies for optimal deployment of related clones into seed orchards. Silvae Genet. 57(3): 119-127.

Domahidi, A., Chu, E., and Boyd, S. 2013. ECOS: An SOCP solver for embedded systems. In Control Conference (ECC), 2013 European. pp. 3071-3076.

Hallander, J., and Waldmann, P. 2009a. Optimization of selection contribution and mate allocations in monoecious tree breeding populations. BMC Genetics 10(1): 1-17. doi: 10.1186/1471-2156-10-70.

Hallander, J., and Waldmann, P. 2009b. Optimum contribution selection in large general tree breeding populations with an application to Scots pine. Theor. Appl. Genet. 118(6): 1133-1142. doi: 10.1007/s00122-009-0968-7.

Henderson, C.R. 1976. A simple method for computing the inverse of a numerator relationship matrix used in the prediction of breeding values. Biometrics, 32: 69-83.

Hinrichs, D., Wetten, M., and Meuwissen, T.H.E. 2006. An algorithm to compute optimal genetic contributions in selection programs with large numbers of candidates. J. Anim. Sci. 84(12): 3212-3218. doi: 10.2527/jas.2006-145.

Kerr, R.J., Goddard, M.E., and Jarvis, S.F. 1998. Maximising genetic response in tree breeding with constraints on group coancestry. Silvae Genet. 47(2-3): 165-173.

Land, A., and Doig, A. 1960. An automatic method of solving discrete programming problems. Econometrica 28(3): 497-520.

Lindgren, D., Danusevičius, D., and Rosvall, O. 2009. Unequal deployment of clones to seed orchards by considering genetic gain, relatedness and gene diversity. Forestry, 82(1): 17-28. doi: 10.1093/forestry/cpn033.

Lindgren, D., Libby, W.S., and Bondesson, F.L. 1989. Deployment to plantations of numbers and proportions of clones with special emphasis on maximizing gain at a constant diversity. Theor. Appl. Genet. 77(6): 825-831. doi: 10.1007/bf00268334.

Lindgren, D., and Matheson, A.C. 1986. An algorithm for increasing the genetic quality of seed from seed orchards by using the better clones in higher proportions. Silvae Genet. 35(5-6): 173-177.

Lindgren, D., and Mullin, T.J. 1997. Balancing gain and relatedness in selection. Silvae Genet. 46(2-3): 124-129.

Meuwissen, T.H.E. 1997. Maximizing the response of selection with a predefined rate of inbreeding. J. Anim. Sci. 75(4): 934-940.

Mullin, T.J. 2014. OPSEL 1.0: A Computer Program for Optimal Selection in Forest Tree Breeding. Arbetsrapport från Skogforsk Nr 814-2014.

Mullin, T.J. 2017. XDesign 1.0: a computer program for generation of complex mating designs with optimal contributions. Arbetsrapport från Skogforsk Nr 956-2017.

Mullin, T.J., and Belotti, P. 2016. Using branch-and-bound algorithms to optimize selection of a fixed-size breeding population under a relatedness constraint. Tree Genet. Genom. 12(4).

Mullin, T.J., and Persson, T. 2017. Assembling optimum breeding populations for the Swedish Scots pine breeding program. Arbetsrapport från Skogforsk Nr 951-2017.

Pong-Wong, R., and Woolliams, J. 2007. Optimisation of contribution of candidate parents to maximise genetic gain and restricting inbreeding using semidefinite programming (<i>Open Access publication</i>). Genetics Selection Evolution 39(1): 1-23. doi: 10.1186/1297-9686-39-1-3.

Quaas, R.L. 1976. Computing the diagonal elements and inverse of a large numerator relationship matrix. Biometrics, 32: 949-953.

Resende, M.F.R., Muñoz, P., Resende, M.D.V., Garrick, D.J., Fernando, R.L., Davis, J.M., Jokela, E.J., Martin, T.A., Peter, G.F., and Kirst, M. 2012. Accuracy of genomic selection methods in a standard data set of loblolly pine (*Pinus taeda* L.). Genetics, 190(4): 1503-1510. doi: 10.1534/genetics.111.137026.

White, T.L., Adams, W.T., and Neale, D.B. 2007. Forest genetics. CABI Publishing, Wallingford, Oxfordshire, UK.

Yamashita, M., Mullin, T.J., and Safarina, S. 2017. An efficient second-order cone programming approach for optimal selection in tree breeding. Optimization Letters (in press).