Planing crosses

Matthias Frisch

1 Graphical genotypes **3**
1.1 Details of haplotype block building . . . 4
1.2 Graphical genotype of the best lines . . . 5

2 Marking reference genotypes **6**

3 Genetic distances of crossing partners **7**

4 Genome-wide marker effects **10**

5 Optimum contribution selection **11**

6 Optimum cross selection **13**
6.1 A complete observed-marker panel . . . 13
6.2 Joint selection of parents and crosses by mean . . . 13
6.3 Pairing at fixed parental contributions . . . 15
6.4 Simulation of the planned DH families . . . 16

7 Optimum cross selection using usefulness **17**
7.1 One score table for both criteria . . . 18
7.2 The usefulness criterion . . . 19

8 Optimum mate allocation **20**
8.1 Mating and parental bounds . . . 20
8.2 Mean-based mate allocation . . . 20
8.3 Usefulness-based mate allocation . . . 21

9 Proportional loss of expected genetic standard deviation **22**
9.1 A diversity constraint in cross selection . . . 23

10 Allocation of progeny to fixed crosses **24**
10.1 Comparison with simulated selected descendants . . . 27

11 Reproductive roles and parent capacities **27**

12 Joint choice of crosses and family sizes **31**

13 Allocation of produced and retained descendants **34**

14 Selection-aware contributions and selected DH genotypes **37**

A Mathematical definitions and implemented algorithms **47**

A.1 Marker prediction, segregation variance and usefulness . . . 47
A.2 Parental contributions and equal-sized integer families . . . 48
A.3 Continuous optimum mate allocation . . . 49
A.4 Covariance-based proportional loss of SD . . . 50
A.5 Integer progeny allocation for fixed families . . . 51
A.6 Finite-normal order statistics and discrete allocation . . . 52
A.7 Common population evaluation . . . 53
A.8 Reproductive roles and practical constraints . . . 54
A.9 Joint family activation and produced counts . . . 55
A.10 Two-budget allocation of produced and retained descendants . . . 56
A.11 Exact DH moments and selection-aware founder contributions . . . 58
A.12 Founder proxy, constraints and actual genomic metrics . . . 59
A.13 Actual-genotype scenarios, variance and inheritance . . . 60
A.14 Actual selected DH lines and a second crossing cycle . . . 61
A.15 Finite-bank search and paired future assessment . . . 62
A.16 Labelled gain–diversity frontier . . . 63

B Cross selection from external diploid inputs **63**

B.1 Family means and variances . . . 63
B.2 Three crossing plans . . . 64

1 Graphical genotypes

We consider 264 tropical maize lines from CIMMYT's Drought Tolerance Maize for Africa project that were analyzed with 1135 SNP markers. Data from Crossa et al. (2010). The original marker data are available from the Genetics webpage. Thanks to Dr. Jose Crossa and Dr. Raman Babu for providing the map data and the permission to use this data set as an example for data analysis. We load the package and the tropical maize data

```
> library("SelectionTools")
> st.set.info.level(0)
> data("v-tropmaize-vcf")
> data("v-tropmaize-phe")
> st.load.vcf.data(v.tropmaize.vcf)
> st.load.performance.data(v.tropmaize.phe)
```

The data are then preprocessed.

```
> st.restrict.marker.data( NoAll.MAX = 2    )
> st.restrict.marker.data( MaMis.MAX = 0.05 )
> st.restrict.marker.data( ExHet.MIN = 0.1  )
> st.restrict.marker.data( InMis.MAX = 0.1  )
```

We then make a copy of the preprocessed data for later.

```
> st.copy.marker.data("preprocessed","default")
```

Building haplotype blocks is a two-step procedure. In the first step the borders of the haplotypes are determined. Several options are available. We use here five adjacent markers.

```
> st.copy.marker.data( "hblocks01", "preprocessed" )
> h <- st.def.hblocks( hap      = 5,
+                      hap.unit = 1,
+                      data.set = "hblocks01" )
```

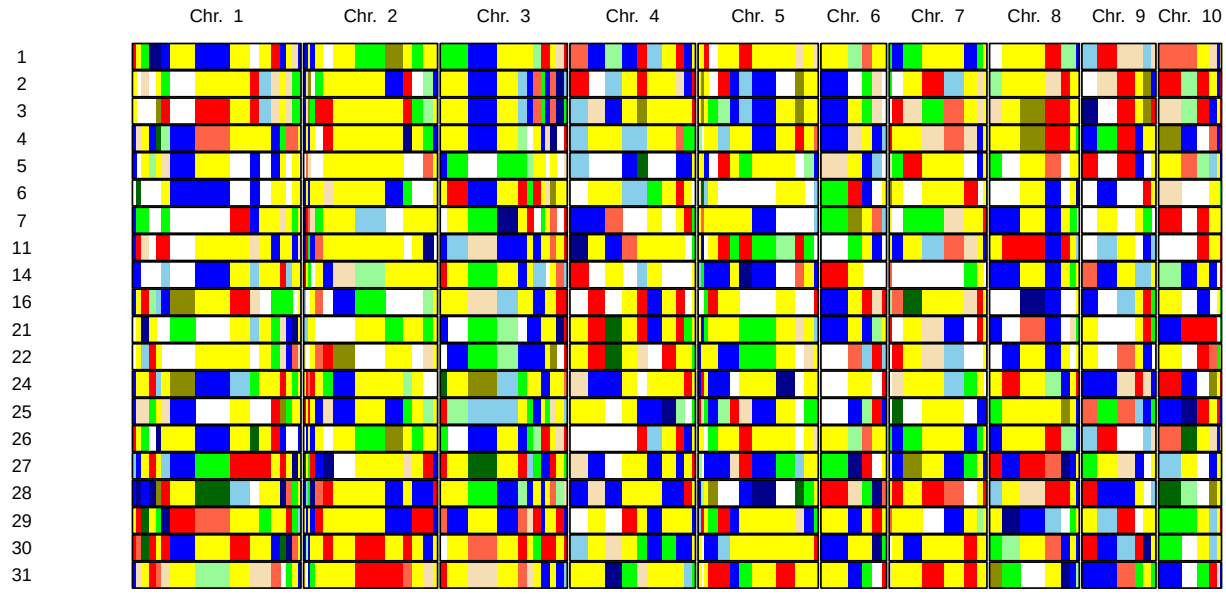
In the second step the variants of the haplotype blocks are determined. For inbred lines, the variants are determined with `st.recode.hil()`.

```
> st.recode.hil ( data.set="hblocks01" )
```

and the plot

```
> st.plot.ggt( f.ind    = 1,
+              l.ind    = 20,
+              data.set = "hblocks01")
```

Here are the graphical genotypes of the first 20 lines in the data set:



1.1 Details of haplotype block building

This is the list of haplotype blocks, five markers are combined to a block, the blocks are subsequently nameed started with `b000000`

```
> h.first <- h[1:10, ]
> h.first
```

```
   Chrom       Pos    Name Class
1      1   3.774747 b000000     b
2      1  10.654315 b000001     b
3      1  20.484429 b000002     b
4      1  36.397726 b000003     b
5      1  47.886392 b000004     b
6      1  55.773202 b000005     b
7      1  75.985652 b000006     b
8      1 145.383079 b000007     b
9      1 198.285440 b000008     b
10     1 213.593020 b000009     b
                                          Markers
1   PZA036131;PZA036142;PZA003931;PZA003934;PZA028692;
2   PZB019151;PZA035191;PZA035201;PZA035203;PZA035521;
3   PZA035583;PZA035571;PZA036631;PZA029219;PZB025153;
4   PZB020583;PZB016622;PZB016621;PZA001926;PZA037422;
5   PZA037421;PZB025171;PZA036732;PZA036731;PZA031681;
6   PZA031682;PZA037461;PZA003281;PZA035611;PZA035612;
7   PZA003789;PZA032404;PZA034072;PZA034073;PZA034074;
8  PHM1062129;PZB010671;PHM172534;PZA035802;PZB001492;
9   PZD000535;PZA034271;PZA034272;PZA036851;PZB011562;
10 PZA032182;PZA031931;PZA0028911;PZA037411;PZA036323;
```

Alternatively we could form haplotyype blocks using a fixed length of the chromosome stretches

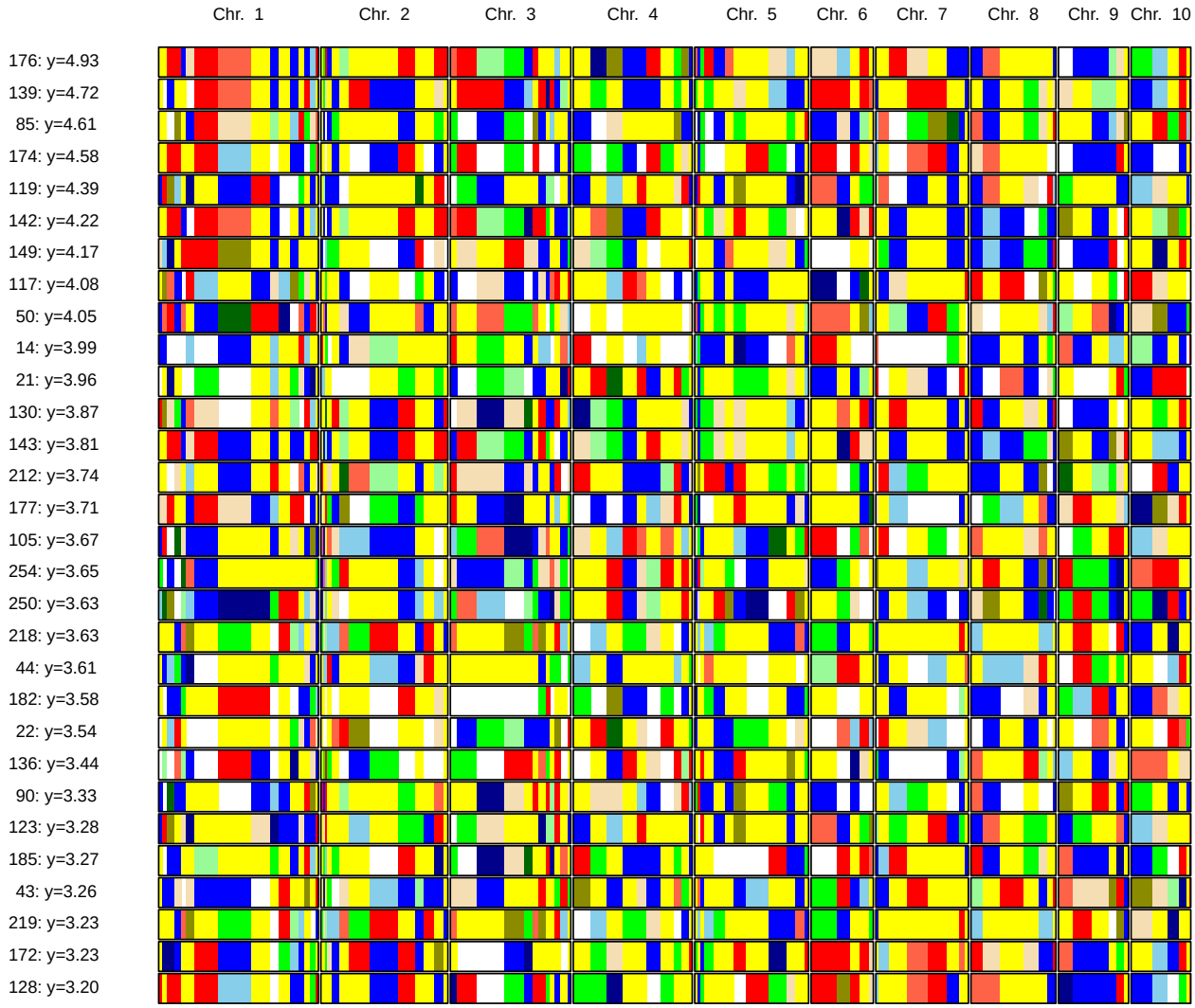
```
> st.copy.marker.data( "hblocks02", "preprocessed" )
> h2 <- st.def.hblocks( hap      = 5,
+                       hap.unit = 2,
+                       data.set = "hblocks02" )
> h2.first <- h2[1:10, ]
> h2.first
```

```
   Chrom        Pos    Name Class
1      1   5.999946 b000000     b
2      1  17.517206 b000001     b
3      1  31.194276 b000002     b
4      1  39.391025 b000003     b
5      1  46.893396 b000004     b
6      1  54.497620 b000005     b
7      1  74.244593 b000006     b
8      1  95.772987 b000007     b
9      1 137.777371 b000008     b
10     1 181.739428 b000009     b
                                                                                Markers
1                       PZA036131;PZA036142;PZA003931;PZA003934;PZA028692;PZB019151;
2  PZA035191;PZA035201;PZA035203;PZA035521;PZA035583;PZA035571;PZA036631;PZA029219;
3                                                    PZB025153;PZB020583;PZB016622;
4                                                    PZB016621;PZA001926;PZA037422;
5                                                    PZA037421;PZB025171;PZA036732;
6                                 PZA036731;PZA031681;PZA031682;PZA037461;PZA003281;
7                                           PZA035611;PZA035612;PZA003789;PZA032404;
8                                         PZA034072;PZA034073;PZA034074;PHM1062129;
9                                                          PZB010671;PHM172534;
10                                                         PZA035802;PZB001492;
```

1.2 Graphical genotype of the best lines

Performance values stored in the marker data set can be used directly by `st.plot.ggt`. With `sort.performance=TRUE`, the individuals are sorted according to their performance before `f.ind` and `l.ind` are applied. Thus, with decreasing sorting, `f.ind=1` and `l.ind=30` select the 30 best lines. With `show.performance=TRUE`, the performance value is added automatically to each individual label.

```
> st.plot.ggt(f.ind                 = 1,
+             l.ind                 = 30,
+             sort.performance      = TRUE,
+             show.performance      = TRUE,
+             performance.decreasing = TRUE,
+             data.set              = "hblocks01")
```



2 Marking reference genotypes

To find a suitable crossing partner, a line can be marked as a reference. Chromosome regions of a line that are similar to the reference line are yellow, other chromosome segments are blue. We first make a copy of the haplotyped data set

```
> st.copy.marker.data( "hblocks03", "hblocks01" )
```

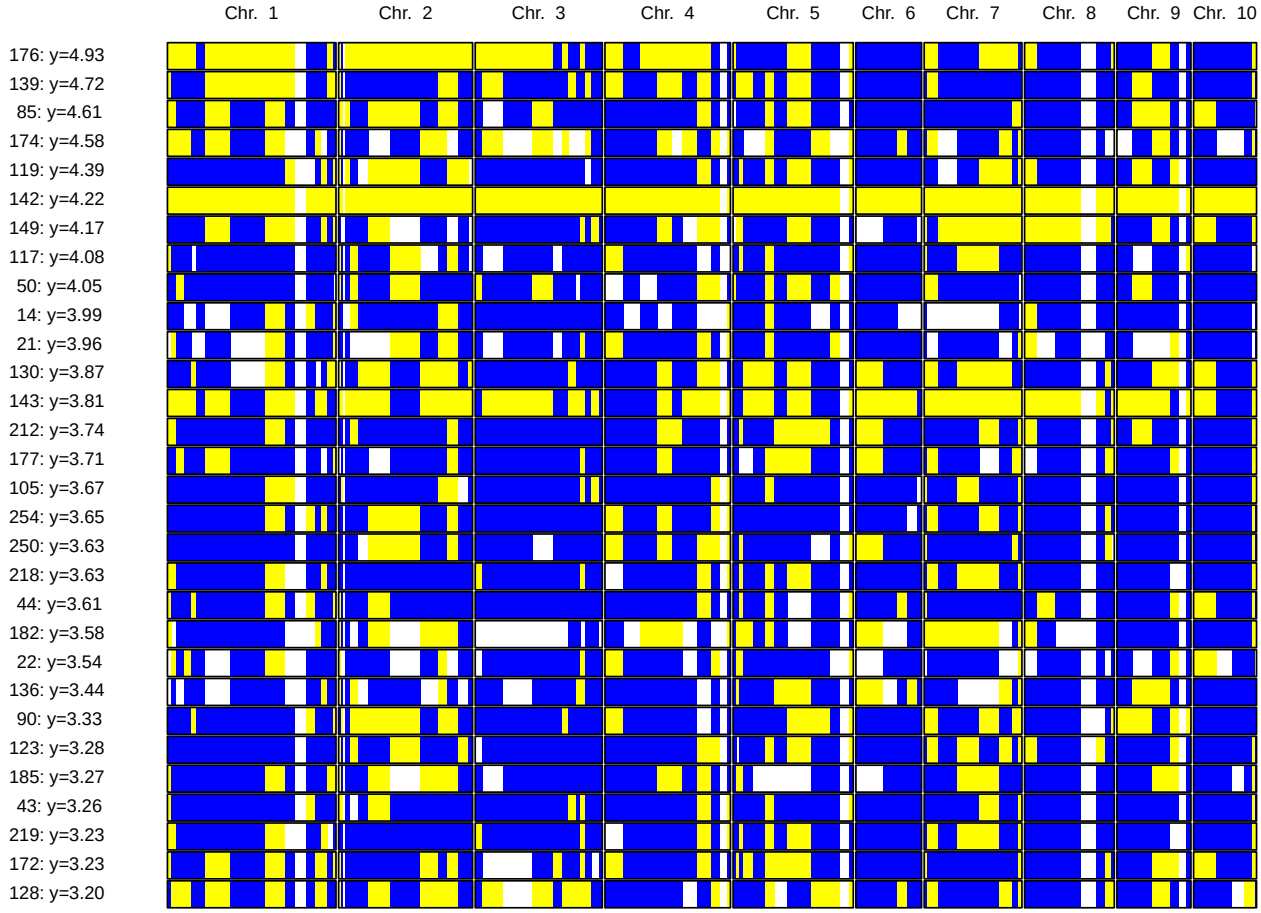
The reference individual can be specified directly by its identifier. Here the individual named `142` is used as reference.

```
> st.recode.ref( reference = "142",
+                data.set  = "hblocks03")
```

We then plot the 30 best lines again. Sorting and performance labels are obtained directly from the performance data stored in the data set.

```
> st.plot.ggt( f.ind                = 1,
+              l.ind                = 30,
+              sort.performance     = TRUE,
+              show.performance     = TRUE,
```

```
+              performance.decreasing = TRUE,
+              data.set               = "hblocks03")
```



3 Genetic distances of crossing partners

We make a copy of the preprocessed marker data individuals.

```
> st.copy.marker.data( "gd01", "preprocessed" )
```

The performance values stored in the SelectionTools data set can be used by `st.select.phen` to give a list of the best 50 lines

```
> best <- st.select.phen( n=50, data.set="gd01" )
```

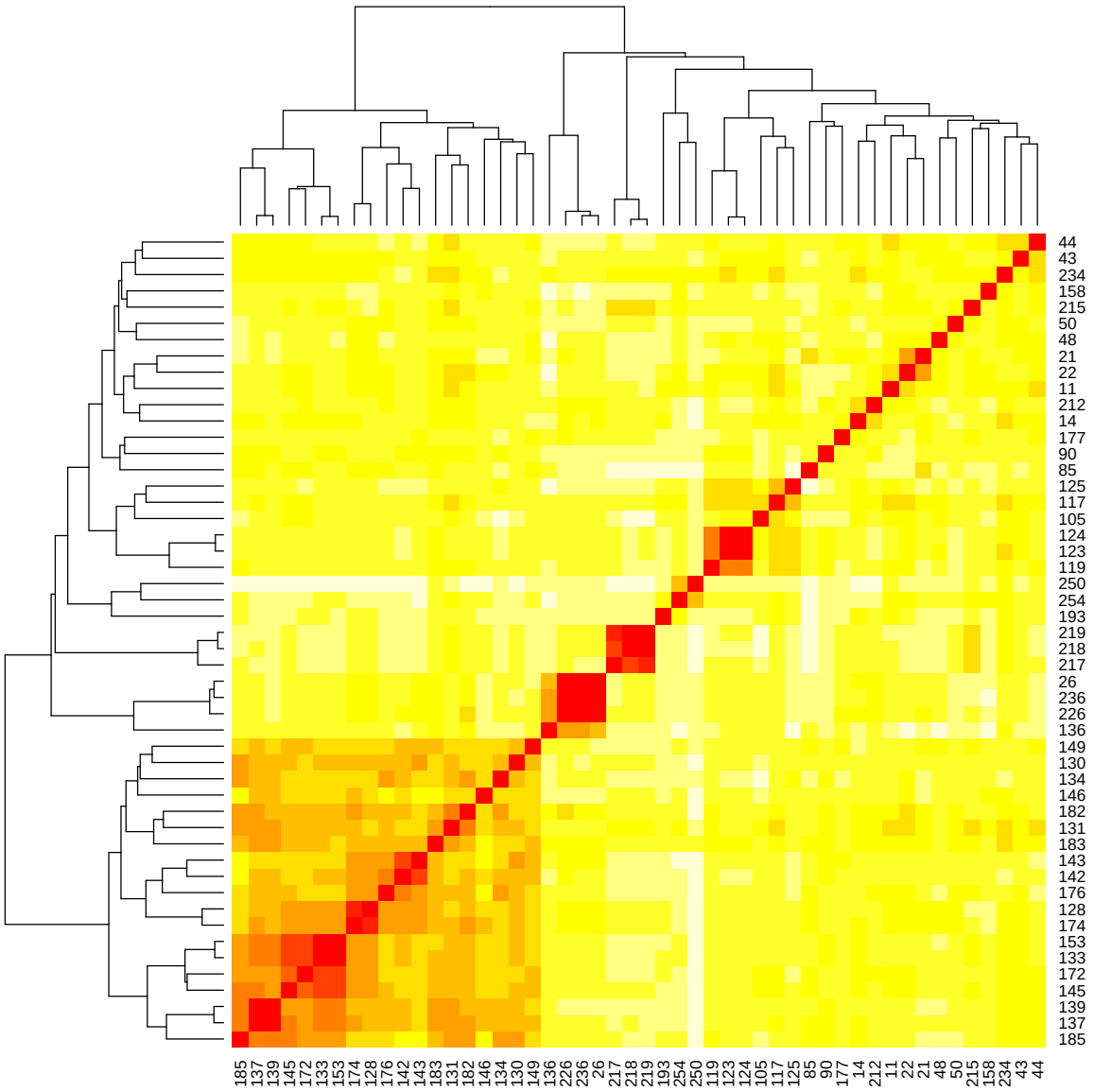
Then a new data set is generated with these 50 lines

```
> st.restrict.marker.data ( ind.list=best$i,
+                           data.set="gd01" )
```

The function `st.genetic.distances` calculates pariwise genetic distances of the genotypes in a a data set. We calculate the Rogers Distance with `measure ="rd"`. With `format="m"` a standard R `dist` object is returned and can be converted to a square matrix for plotting.

```
> dist.mat <- st.genetic.distances( measure  = "rd",
+                                   format   = "m",
```

```
+                                        data.set = "gd01" )
> dm <- as.matrix(dist.mat)
> heatmap( dm, scale="none", col=heat.colors(12) )
```



For ranking the crossing combinations we use the Modified Rogers distance. The pairwise distances are therefore recalculated with `gs.cross.eval.gd` using `dist="mrd"`, and the ranked crossing information is then retrieved with `gs.cross.info`.

```
> gs.cross.eval.gd ( dist="mrd", data.set="gd01" )
> crosses <- gs.cross.info ( sortby="gd", bestn=50, data.set="gd01" )
> distance.crosses <- crosses[1:10, ]
> distance.crosses
```

```
   P1No P2No P1Name P2Name        gd mu mi ma va es
1    10   50     85    254 0.6112693 NA NA NA NA NA
2    45   49    219    250 0.6074501 NA NA NA NA NA
3    34   49    174    250 0.6067799 NA NA NA NA NA
4    31   49    153    250 0.6061281 NA NA NA NA NA
5    44   49    218    250 0.6061281 NA NA NA NA NA
6    18   49    128    250 0.6049901 NA NA NA NA NA
```

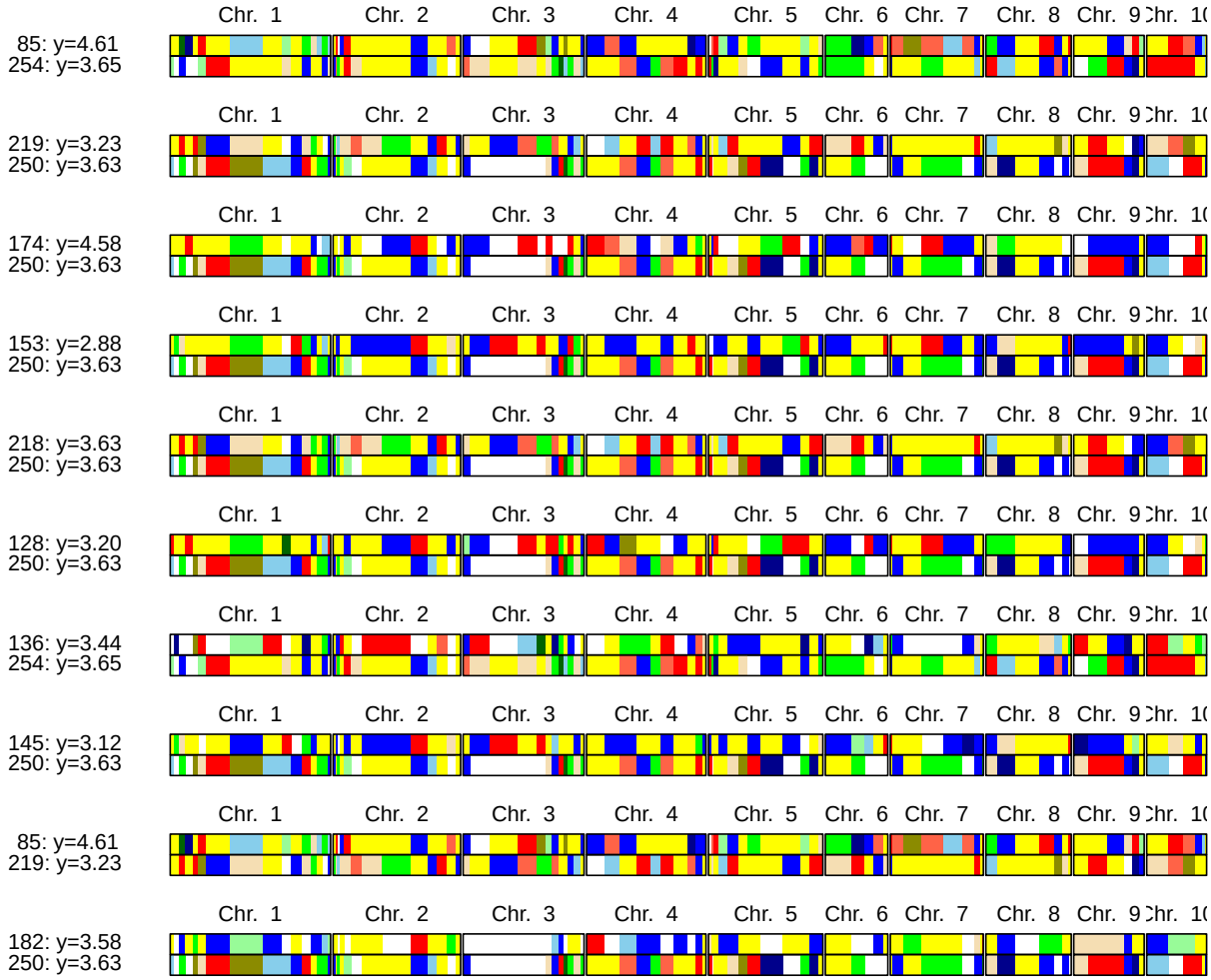
```
7    23   50    136    254 0.6038653 NA NA NA NA NA
8    28   49    145    250 0.6033532 NA NA NA NA NA
9    10   45     85    219 0.6025309 NA NA NA NA NA
10   37   49    182    250 0.6025062 NA NA NA NA NA
```

We now want to inspect the graphical genotypes of the pairs of lines with the greatest genetic distances and generate haplotype blocks

```
> st.def.hblocks( hap=5, data.set="gd01" )
> st.recode.hil ( data.set="gd01" )
```

and plot the 10 crosses with the greatest genetic distance:

```
> st.set.info.level(-2)
> for (i in seq_len(min(10,nrow(crosses)))) {
+     a <- crosses$P1Name[i]
+     b <- crosses$P2Name[i]
+     f.nme <- sprintf("ex-crs-07-%03i.pdf",100+i)
+     pdf(f.nme,width=8,height=0.8,pointsize=10)
+     st.plot.ggt(i.list=c(a,b),
+                 show.performance=TRUE,
+                 cex.chrom=1,
+                 p.t=FALSE,
+                 p.s=FALSE,
+                 data.set="gd01")
+     dev.off()
+ }
```



4 Genome-wide marker effects

We make a copy of the processed data set

```
> st.copy.marker.data( "effects01", "preprocessed" )
```

Then genome-wide marker effects are estimated from the marker and performance data. Here we use BLUP as implemented in `gs.esteff.rr`.

```
> gs.esteff.rr( method="BLUP", data.set="effects01" )
```

Once marker effects have been estimated, all possible crosses among the individuals in the data set can be evaluated using several complementary criteria. The expected cross mean is calculated with `gs.cross.eval.mu`. Lower and upper marker-effect based limits are calculated with the functions `gs.cross.eval.mi` and `gs.cross.eval.ma` for progeny values that can be assembled from the parental alleles.

```
> gs.cross.eval.mu( data.set="effects01" )
> gs.cross.eval.mi( data.set="effects01" )
> gs.cross.eval.ma( data.set="effects01" )
```

The segregation variance describes how strongly progeny from a cross are expected to differ genetically. It can be calculated for doubled-haploid or single-seed-descent progeny while taking linkage and recombination into account. The approach implemented here follows Osthushenrich et al. (2017).

```
> gs.cross.eval.va( pop.type="DH", data.set="effects01" )
```

The expected superior progeny criterion combines the expected cross mean with the amount of segregation expected among the progeny. It therefore favours crosses that have a high mean and, at the same time, enough genetic variation to generate outstanding descendants. For large populations the selected fraction is positive and no greater than one. With this fraction, the approach of Zhong and Jannink (2007) is used.

```
> gs.cross.eval.es( alpha=0.25, data.set="effects01" )
```

For small finite progeny populations, the numbers of selected individuals and evaluated individuals can be supplied directly; the corresponding selection-intensity correction follows Burrows (1972). Both counts are positive whole numbers, with `N` no greater than `G`. When all progeny are retained, `N=G` or `alpha=1`, the selection differential is zero and the criterion equals the cross mean. The mean and variance are evaluated before this criterion; a calculated zero variance is valid.

```
> gs.cross.eval.es( N=1, G=4, data.set="effects01" )
```

Genetic distance can be evaluated independently of the marker effects. In this example we use Rogers distance; properties of genetic distance measures used in plant breeding are discussed by Reif et al. (2005).

```
> gs.cross.eval.gd(dist="rd",data.set="effects01")
```

The criteria can be calculated for the same set of crosses and then summarized together. Criteria that have not been evaluated appear as `NA`, whereas an evaluated zero remains zero. Sorting requires an evaluated criterion. The lower marker-effect limit, `sortby="mi"`, is sorted from smallest to largest; the other scores are sorted from largest to smallest. Here the crosses are ranked by expected superior progeny value and the best 20 are returned.

```
> gs.cross.eval.gd(dist="rd",data.set="effects01")
> gs.cross.eval.mu(data.set="effects01")
> gs.cross.eval.ma(data.set="effects01")
> gs.cross.eval.mi(data.set="effects01")
> gs.cross.eval.va(pop.type="DH",data.set="effects01")
> gs.cross.eval.es(N=1,G=4,data.set="effects01")
> crosses <- gs.cross.info(sortby="es",bestn=20,data.set="effects01")
> formatted.crosses <- format(crosses,digits=2,nsmall=2)
> formatted.crosses
```

	P1No	P2No	P1Name	P2Name	gd	mu	mi	ma	va	es
1	123	153	142	176	0.148	3.62	-1.05	6.35	0.032	3.80
2	124	153	143	176	0.163	3.59	-0.97	6.40	0.025	3.76
3	123	124	142	143	0.093	3.55	-0.83	5.90	0.020	3.70
4	118	153	137	176	0.207	3.51	-1.43	6.75	0.029	3.69
5	118	123	137	142	0.212	3.47	-1.16	6.59	0.035	3.66
6	120	153	139	176	0.212	3.48	-1.47	6.75	0.029	3.66
7	120	123	139	142	0.217	3.44	-1.23	6.59	0.036	3.64
8	118	124	137	143	0.226	3.45	-1.28	6.59	0.025	3.61
9	111	153	130	176	0.194	3.43	-1.46	6.59	0.029	3.61
10	100	153	119	176	0.304	3.32	-1.83	6.85	0.080	3.61
11	111	123	130	142	0.200	3.39	-1.30	6.41	0.040	3.60
12	105	153	124	176	0.300	3.39	-1.73	6.78	0.037	3.59
13	109	153	128	176	0.186	3.42	-1.24	6.42	0.025	3.59
14	104	153	123	176	0.297	3.39	-1.66	6.78	0.034	3.58
15	109	123	128	142	0.168	3.38	-1.17	6.07	0.037	3.58
16	120	124	139	143	0.229	3.42	-1.31	6.60	0.024	3.58
17	105	123	124	142	0.317	3.35	-1.65	6.55	0.046	3.57
18	104	123	123	142	0.314	3.35	-1.62	6.52	0.047	3.57
19	151	153	174	176	0.176	3.41	-1.19	6.42	0.021	3.56
20	109	124	128	143	0.183	3.36	-1.30	6.05	0.032	3.54

5 Optimum contribution selection

The predicted genetic values can also be used to choose the proportion of genes contributed by each parent to the next generation. For an additive model, the expected genetic value of the resulting population is the contribution-weighted mean of these parental values. `st.ocs` maximizes this mean subject to an upper bound on group relatedness and lower and upper bounds on each contribution (Wellmann, 2019). These continuous proportions sum to one.

The following analysis uses the same tropical-maize lines and fitted marker effects as the cross evaluations. Predicted genetic values supply the named merit vector. The genomic relationship matrix, `A`, is centered using marker frequencies in this complete candidate set. The matrix and its reference remain fixed when contributions change. Here `K=A/2` is the kinship-scaled matrix, and group relatedness is the quadratic form of `K` with the contribution vector. In this example it is a genomic relatedness index relative to the candidate panel, rather than an absolute probability of identity by descent. Equal contributions to all candidates have an index approximately zero on this centered reference.

```
> values <- gs.estimate.gv(estimation.set="effects01",
+                          prediction.set="effects01")
> merit <- setNames(values$yhat, values$i)
> A <- st.get.A("effects01")
> merit <- merit[rownames(A)]
> K <- A/2
```

The chosen breeding scenario sets `theta=0.02` as the bound on group relatedness on this reference scale. Each contribution has a lower bound of zero and an upper bound of 0.1, so a single parent can supply at

most ten percent of the nominal gene contribution. The merit vector is ordered explicitly to match both dimensions of `K`. The returned `gain` is the contribution-weighted mean of the supplied merit values.

```
> contributions <- st.ocs(
+     merit = merit,                          # Parental genetic values
+     K = K,                                  # Kinship matrix
+     theta = 0.02,                           # Group relatedness limit
+     lower = 0,                              # Minimum parental contribution
+     upper = 0.1                             # Maximum parental contribution
+ )
> contributions
```

```
Optimum contribution selection
Criterion: maximum expected genetic merit
Status: optimal
Expected merit: 2.7698
Group relatedness: 0.0200; requested limit: 0.0200
Minimum relatedness: >-0.0001; lower bound: 0.0000
Gain upper bound: 2.7698; optimality gap: <0.0001
Maximum constraint residual: <0.0001

Parental contributions (236 parents, 64 positive)
  Parent  Contribution
  218           0.0575
  176           0.0433
  44            0.0399
  119           0.0376
  137           0.0368
  85            0.0350
  117           0.0344
  226           0.0325
  50            0.0289
  21            0.0282
  [221 table rows omitted]
  258           0.0000
  259           0.0000
  261           0.0000
  263           0.0000
  264           0.0000
```

Interpretation of the printed summary

The summary above the contribution table reports the objective and the numerical qualification of the solution.

Criterion and status. The criterion is the quantity maximized. `optimal` denotes a certified merit optimum; `maxit` denotes an unresolved iteration or numerical limit. An infeasible request has no solution under the supplied constraints.

Expected merit. This is the contribution-weighted mean of the supplied parental genetic values.

Group relatedness. This is the achieved $c'Kc$, compared with the requested limit `theta`.

Minimum relatedness and lower bound. These describe the separate search for the smallest attainable relatedness under the contribution bounds. Their difference measures its remaining numerical gap.

Gain upper bound and optimality gap. The bound limits the best attainable merit. Its difference from the achieved merit is the gap.

Maximum constraint residual. This is the largest numerical violation of the constraints; values close to zero indicate feasibility within numerical tolerance.

The contribution table identifies the parents and their gene shares, which sum to one. Its displayed rows include the greatest and smallest shares.

The result also reports the minimum-relatedness calculation for the supplied contribution bounds. A requested `theta` below the feasible range is infeasible. The returned status and diagnostics describe feasibility and numerical convergence. These contributions refer to the nominal gene shares before selection within progeny families. A crossing plan additionally specifies parental pairs and integer progeny counts; realized ancestry after selection can differ from the nominal contributions.

6 Optimum cross selection

`st.cross.plan` selects parent pairs for a specified number of families. The default joint mode chooses both parental usage and pairing to maximize the predicted additive population mean, subject to contribution limits and the group relatedness bound. The specified number of crosses refers to distinct unordered pairs of different parents. Here every family has the same progeny count.

6.1 A complete observed-marker panel

The execution example uses a separate panel comprising the first 30 lines in the processed data order. Markers with any missing call among these parents are excluded, leaving 243 markers with complete homozygous founder genotypes. The original intercept and the fitted effects of these retained markers define the prediction model for this example. Merit and the genomic relationship matrix are recalculated for precisely this panel. Its marker set and reference frequencies differ from the preceding 236-line analysis; the relatedness bound therefore has this smaller panel as its reference. Planning and simulation below share the same parents, markers and effect model.

```
> marker.effects <- gs.return.effects(data.set="effects01")
> st.copy.marker.data("cross.parents", "effects01")
> st.restrict.marker.data(ind.list=names(merit)[1:30],
+                         data.set="cross.parents")
> st.restrict.marker.data(MaMis.MAX=0, data.set="cross.parents")
> cross.map <- st.get.map(data.set="cross.parents")
```

Marker-data copies do not retain fitted effects. The marker names from the map identify the original effects attached by `gs.set.effects`, together with the fitted intercept.

```
> cross.effects <- marker.effects[
+   marker.effects$marker %in% c("mu", cross.map$Name), ]
> gs.set.effects(cross.effects, data.set="cross.parents")
> cross.values <- gs.estimate.gv(estimation.set="cross.parents",
+                                prediction.set="cross.parents")
```

The named merit vector follows the parent order of `cross.A`. Dividing this additive relationship matrix by two supplies the kinship scale used in the following plans.

```
> cross.merit <- setNames(cross.values$yhat, cross.values$i)
> cross.A <- st.get.A("cross.parents")
> cross.merit <- cross.merit[rownames(cross.A)]
> cross.K <- cross.A/2
```

6.2 Joint selection of parents and crosses by mean

The following breeding scenario contains 15 distinct families with 100 progeny each. A parental contribution of at most 0.1 permits a parent to occur in at most three of these equal-sized families. Group relatedness is bounded by `theta=0.02` on the 30-parent reference scale.

```
> cross.plan <- st.cross.plan(
+     merit = cross.merit,                    # Parental genetic values
+     K = cross.K,                            # Kinship matrix
+     n.crosses = 15,                         # Number of distinct parent pairs
+     n.progeny = 100,                        # Progeny per family
+     theta = 0.02,                           # Group relatedness limit
+     upper = 0.1                             # Maximum parental contribution
+ )
> cross.plan
```

```
Optimum cross selection (mean)
Criterion: maximum predicted mean
Status: best.found
Families: 15; progeny per family: 100; total progeny: 1500
Expected mean: 2.3854
Group relatedness: 0.0200; requested limit: 0.0200
Average within-cross relatedness: -0.0159
Objective bound: 2.3992; optimality gap: 0.0138
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship
       1  1         5              100  2.2508  -0.0689
       2  1         27             100  2.3523  -0.0447
       3  2         6              100  2.4583  -0.0311
       4  2         30             100  2.2902  -0.0290
       5  3         7              100  2.1269  -0.1213
       6  6         34             100  2.4984   0.0129
       7  11        14             100  2.7394  -0.1286
       8  11        22             100  2.6670   0.1692
       9  11        38             100  2.6248  -0.0542
      10  14        34             100  2.6336  -0.0500
      11  14        39             100  2.3035  -0.0657
      12  16        22             100  2.3135   0.0832
      13  29        31             100  2.0175   0.1063
      14  32        37             100  2.1386  -0.0825
      15  34        41             100  2.3654   0.0664
```

Interpretation of the printed summary

The summary above the chosen crosses describes the breeding scenario, objective and numerical qualification of the plan.

Criterion and status. The criterion names the objective. `best.found` denotes a feasible search result, whereas `optimal` also requires a bound certifying the optimum. `search.limit` means that the search has not found a feasible plan.

Families and progeny. These give the number of distinct crosses, the progeny count per family and the total progeny count.

Expected mean. This is the average predicted cross mean for these equal-sized families. With supplied cross scores, expected usefulness additionally includes the predicted gain from within-family selection; `accuracy` specifies the assumed accuracy of that ranking.

Group relatedness. This is $c'Kc$ for the pooled parental gene shares, compared with the requested limit.

Average within-cross relatedness. This is the average kinship between the two parents of each chosen cross. It concerns pairing and can change while pooled parental contributions remain fixed.

Objective bound and optimality gap. The bound is an upper bound for a maximized objective and a lower bound for the minimized pairing objective. The gap measures the remaining difference from that bound.

Maximum constraint residual. This is the largest numerical violation of the plan constraints.

The cross table gives the parental identifiers, progeny counts, predicted cross means and parental kinships. A usefulness column appears when the corresponding cross scores are supplied.

6.3 Pairing at fixed parental contributions

With `mode="fixed"`, the parental gene shares are supplied explicitly. Their additive predicted mean and group relatedness remain fixed, while the pairing objective is to minimize average within-cross relatedness. This example uses the gene shares of the preceding feasible plan, so they already correspond to integer parental-use counts for the 15 families. Other supplied proportions that cannot be realized by the requested integer family counts are reported as infeasible without automatic rounding. The preceding plan provides the starting mating list through `control=list(start=cross.plan)`.

```
> fixed.plan <- st.cross.plan(
+     merit = cross.merit,                        # Parental genetic values
+     K = cross.K,                                # Kinship matrix
+     n.crosses = 15,                             # Number of distinct parent pairs
+     n.progeny = 100,                            # Progeny per family
+     theta = 0.02,                               # Group relatedness limit
+     upper = 0.1,                                # Maximum parental contribution
+     contributions = cross.plan$contributions,   # Fixed parental gene shares
+     mode = "fixed",                             # Pairing at fixed contributions
+     control = list(                             # Search settings
+         start = cross.plan                      # Starting crossing plan
+     )
+ )
> fixed.plan
```

```
Fixed-contribution mating allocation
Criterion: minimum average within-cross relatedness
Status: best.found
Families: 15; progeny per family: 100; total progeny: 1500
Expected mean: 2.3854
Group relatedness: 0.0200; requested limit: 0.0200
Average within-cross relatedness: -0.1627
Objective bound: -0.1901; optimality gap: 0.0274
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship
       1  1         2              100  2.3328  -0.0856
       2  1         32             100  2.2892  -0.2974
       3  2         34             100  2.4219  -0.1611
       4  3         34             100  2.1981  -0.2146
       5  5         22             100  2.4391  -0.1884
       6  6         7              100  2.4272  -0.1139
       7  6         29             100  2.3693  -0.1475
       8  11        14             100  2.7394  -0.1286
       9  11        27             100  2.5472  -0.1045
      10  11        31             100  2.2524  -0.1580
      11  14        16             100  2.3860  -0.1202
      12  14        37             100  2.3940  -0.1685
      13  22        34             100  2.5612  -0.1926
      14  30        41             100  2.2336  -0.2419
      15  38        39             100  2.1889  -0.1171
```

6.4 Simulation of the planned DH families

The simulation uses complete homozygous founders. `st.set.simpop` initializes them from the marker data set. The parental names in the plan identify the individuals copied into the two temporary parental populations. `cross` creates one F1 from each pair, and `dh` produces the planned number of doubled-haploid descendants from that F1. The seed is set before the first cross.

```
> st.set.simpop(pop.name="Founders", data.set="cross.parents")
> rng.init(seed=1001)
> founder.names <- population.pedigree("Founders")$name
> planned.crosses <- cross.plan$crosses
> for (i in seq_len(nrow(planned.crosses))) {
+     population.copy("P1", "Founders",
+                     start=match(planned.crosses$P1Name[i], founder.names),
+                     n=1)
+     population.copy("P2", "Founders",
+                     start=match(planned.crosses$P2Name[i], founder.names),
+                     n=1)
+     cross("F1", "P1", "P2", NoPg=1, crossing.scheme=3)
+     dh(sprintf("DH%03d", i), "F1",
+        NoPg=planned.crosses$n.progeny[i])
+ }
> population.remove(c("P1", "P2", "F1"))
> families <- population.list()
> families
```

```
    PopName count
1     DH015   100
2     DH014   100
3     DH013   100
4     DH012   100
5     DH011   100
6     DH010   100
7     DH009   100
8     DH008   100
9     DH007   100
10    DH006   100
11    DH005   100
12    DH004   100
13    DH003   100
14    DH002   100
15    DH001   100
16 Founders    30
```

`population.list` reports the simulation populations and their sizes. The temporary parental and F1 populations have been removed. The 15 DH families contain 100 individuals each, giving 1,500 simulated DH individuals; `Founders` retains the 30 parental lines.

`population.concat` moves the individuals from each DH family into one population named `DH`. The separate family populations are removed, while their individuals and pedigrees are retained in the merged population.

```
> for (i in seq_len(nrow(planned.crosses))) {
+     population.concat("DH", sprintf("DH%03d", i))
+ }
> merged.populations <- population.list()
> merged.populations
```

```
   PopName count
1       DH  1500
2 Founders    30
```

The simulation uses the fitted marker effects and intercept transferred by `st.set.simpop`. Genotyping and evaluation provide genetic values for the founders and the merged DH population under this same model.

```
> genotype.population("Founders DH")
> evaluate.population("Founders DH", effName="effect")
> genetic.summary <- summarize.gvalue("Founders DH", effectfile="effect")
> genetic.summary
```

```
          Obs     Mean      SDev      Min      Q10      Med      Max
Founders   30 2.187315 0.2997672 1.653617 1.799855 2.210782 2.805240
DH       1500 2.383872 0.2514697 1.650580 2.040985 2.383192 3.035688
```

The selection gain is the difference between the mean of the simulated DH population and the equal-weight mean of the 30 founder lines. It reflects the choice of crossing parents before selection among the DH individuals, under the fitted additive model.

```
> selection.gain <- genetic.summary["DH", "Mean"] -
+                   genetic.summary["Founders", "Mean"]
> selection.gain
```

```
[1] 0.1965574
```

The plan predicts the mean of the unselected DH population. Its predicted selection gain uses the same founder mean as the simulated gain. The comparison therefore concerns the same genetic model and founder reference.

```
> prediction.comparison <- data.frame(
+     Predicted = c(cross.plan$gain,
+                   cross.plan$gain - genetic.summary["Founders", "Mean"]),
+     Simulated = c(genetic.summary["DH", "Mean"], selection.gain),
+     row.names = c("Population mean", "Selection gain")
+ )
> prediction.comparison$Difference <- prediction.comparison$Simulated -
+                                     prediction.comparison$Predicted
> prediction.comparison
```

```
               Predicted Simulated   Difference
Population mean 2.3853515 2.3838720 -0.001479536
Selection gain  0.1980369 0.1965574 -0.001479536
```

The difference is the simulated value minus the prediction. Finite Mendelian sampling can make the simulated population mean differ from its expectation. This comparison precedes selection among DH lines; the usefulness criterion describes a different, within-family selection scenario.

7 Optimum cross selection using usefulness

The preceding mean criterion can be compared with usefulness while retaining the same 15 families, 100 progeny per family, candidate pairs, parental-use limits and group-relatedness bound. In this example, ten descendants are retained from each family. Equal retained counts give each family the same weight in the predicted mean of the selected descendants.

7.1 One score table for both criteria

`gs.cross.eval.mu` and `gs.cross.eval.va` evaluate the means and DH segregation variances for the complete founder panel and its existing effect model. The `N=10` and `G=100` arguments specify ten retained individuals among 100 progeny. The resulting `es` score uses the finite-family approximation of Burrows (1972); it is not an exact expectation over finite normal order statistics.

```
> gs.cross.eval.mu(data.set="cross.parents")
> gs.cross.eval.va(pop.type="DH", data.set="cross.parents")
> gs.cross.eval.es(N=10, G=100, data.set="cross.parents")
> cross.scores <- gs.cross.info(data.set="cross.parents")
```

Both analyses receive the same values from `cross.scores`. The mean criterion maximizes the average `mu` of the selected crosses. The result also reports their average usefulness. Both searches receive the same earlier feasible `cross.plan` through `control`; each can retain or improve upon that starting crossing list.

```
> mean.plan <- st.cross.plan(
+     merit = cross.merit,                    # Parental genetic values
+     K = cross.K,                            # Kinship matrix
+     n.crosses = 15,                         # Number of distinct parent pairs
+     n.progeny = 100,                        # Progeny per family
+     theta = 0.02,                           # Group relatedness limit
+     upper = 0.1,                            # Maximum parental contribution
+     scores = cross.scores,                  # Precomputed cross scores
+     criterion = "mean",                     # Objective: maximum cross mean
+     control = list(                         # Search settings
+         start = cross.plan                  # Starting crossing plan
+     )
+ )
> mean.plan
```

```
Optimum cross selection (mean)
Criterion: maximum predicted mean
Status: best.found
Families: 15; progeny per family: 100; total progeny: 1500
Expected mean: 2.3854
Expected usefulness: 2.6241; accuracy: 1.0000
Group relatedness: 0.0200; requested limit: 0.0200
Average within-cross relatedness: -0.0159
Objective bound: 2.6318; optimality gap: 0.2464
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  1         5              100  2.2508  -0.0689      2.4677
       2  1         27             100  2.3523  -0.0447      2.5732
       3  2         6              100  2.4583  -0.0311      2.6836
       4  2         30             100  2.2902  -0.0290      2.5302
       5  3         7              100  2.1269  -0.1213      2.4293
       6  6         34             100  2.4984   0.0129      2.6800
       7  11        14             100  2.7394  -0.1286      2.9392
       8  11        22             100  2.6670   0.1692      2.8504
       9  11        38             100  2.6248  -0.0542      2.8665
      10  14        34             100  2.6336  -0.0500      2.8920
      11  14        39             100  2.3035  -0.0657      2.6186
      12  16        22             100  2.3135   0.0832      2.5542
      13  29        31             100  2.0175   0.1063      2.2565
      14  32        37             100  2.1386  -0.0825      2.4046
      15  34        41             100  2.3654   0.0664      2.6154
```

7.2 The usefulness criterion

The usefulness criterion maximizes the average cross mean plus `accuracy` times the expected selection gain, `es - mu`. The supplied value of one represents perfect within-family ranking under this fitted additive model. An accuracy of zero reduces the objective to the same mean criterion. The score computation, constraints and default search budget are identical in the two calls.

```
> usefulness.plan <- st.cross.plan(
+     merit = cross.merit,                    # Parental genetic values
+     K = cross.K,                            # Kinship matrix
+     n.crosses = 15,                         # Number of distinct parent pairs
+     n.progeny = 100,                        # Progeny per family
+     theta = 0.02,                           # Group relatedness limit
+     upper = 0.1,                            # Maximum parental contribution
+     scores = cross.scores,                  # Precomputed cross scores
+     criterion = "usefulness",               # Objective: maximum usefulness
+     accuracy = 1,                           # Within-family ranking accuracy
+     control = list(                         # Search settings
+         start = cross.plan                  # Starting crossing plan
+     )
+ )
> usefulness.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: best.found
Families: 15; progeny per family: 100; total progeny: 1500
Expected mean: 2.3753
Expected usefulness: 2.6667; accuracy: 1.0000
Group relatedness: 0.0196; requested limit: 0.0200
Average within-cross relatedness: -0.1004
Objective bound: 2.8596; optimality gap: 0.1930
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  1         22             100  2.4721  -0.0856      2.7121
       2  1         32             100  2.2892  -0.2974      2.5902
       3  2         21             100  2.3264  -0.0940      2.6401
       4  2         27             100  2.4013  -0.1014      2.6538
       5  3         34             100  2.1981  -0.2146      2.5346
       6  5         22             100  2.4391  -0.1884      2.7195
       7  6         27             100  2.4778   0.0098      2.7180
       8  6         30             100  2.3667   0.0192      2.5886
       9  7         34             100  2.3908  -0.0552      2.7169
      10  11        34             100  2.5678  -0.1328      2.8246
      11  11        39             100  2.2377  -0.0856      2.5555
      12  14        40             100  2.2893  -0.0594      2.6583
      13  14        41             100  2.5370  -0.0867      2.8071
      14  22        31             100  2.2458  -0.0605      2.5957
      15  29        38             100  2.3899  -0.0731      2.6850
```

Both results report the predicted population mean and usefulness, making the trade-off visible for the two feasible crossing lists. Their `best.found` status describes the search result rather than a certified global optimum. The group-relatedness limit still concerns nominal parental contributions before within-family selection. It does not specify the genomic ancestry of the retained descendants.

The same crossing commands apply to `usefulness.plan$crosses`. These expected selected-descendant scores describe the fitted additive model and the declared selection scenario; they do not establish an advantage in field performance.

8 Optimum mate allocation

Optimum mate allocation assigns a continuous fraction of the produced progeny to each allowed cross (Endelman, 2025). The parental contributions follow from those mating fractions. This differs from the preceding optimum cross-selection examples, which specify 15 distinct families with 100 progeny each. Continuous mate allocation has no fixed number of families or integer progeny counts.

8.1 Mating and parental bounds

The following analysis uses the same complete founder panel, kinship matrix and full-precision cross scores. The three columns of `oma.parents` identify each parent and its minimum and maximum gene contribution. The mating table identifies both parents, the cross mean to be optimized, and its allocation bounds. Here each parent and each mating can receive at most one tenth of the corresponding total.

```
> oma.parents <- data.frame(id=rownames(cross.K), min=0, max=0.1)
> oma.mean.matings <- data.frame(
+   parent1=cross.scores$P1Name, parent2=cross.scores$P2Name,
+   merit=cross.scores$mu, min=0, max=0.1)
```

8.2 Mean-based mate allocation

`base="RM"` uses the randomly mated reference, whose value is the mean of all entries in the supplied `cross.K`. The two `dF` values specify an interval for the first-generation average within-mating kinship, called `F1`. The upper endpoint also sets the bound on the second-generation group quantity, `F2`. On this centered reference, the reference value is approximately zero, so `dF=c(0,0.01)` bounds `F1` between zero and 0.01 and `F2` by approximately 0.0199. These are a chosen reference-scale scenario, not absolute probabilities of identity by descent or a recommended biological inbreeding rate.

```
> oma.mean <- st.oma(
+     parents = oma.parents,                  # Parent contribution bounds
+     matings = oma.mean.matings,             # Allowed crosses and scores
+     K = cross.K,                            # Kinship matrix
+     dF = c(0, 0.01),                        # F1 change range; F2 change cap
+     base = "RM"                             # Random-mating reference
+ )
> oma.mean
```

```
OMA: Optimum mate allocation
Status: optimal
Expected merit: 2.3987
Reference F0: >-0.0001; F1: 0.0020
Group F2: 0.0199; upper limit: 0.0199
dF1: 0.0020; dF2: 0.0100
Merit upper bound: 2.3987; gap: <0.0001
Maximum constraint residual: <0.0001

Continuous mating allocations
  Parent 1  Parent 2  Allocation   Merit
  1         2             0.0107  2.3328
  1         3             0.0019  2.1090
  1         4             0.0016  2.1884
  1         5             0.0063  2.2508
  1         6             0.0088  2.4093
  1         7             0.0048  2.3017
  1         11            0.0148  2.4787
  1         14            0.0367  2.5445
  1         16            0.0036  2.1252
  1         21            0.0032  2.2774
```

```
[420 table rows omitted]
38        40            0.0003  2.1747
38        41            0.0023  2.4224
39        40            0.0002  1.7876
39        41            0.0006  2.0352
40        41            0.0003  2.0211
```

Interpretation of the printed summary

Status. `optimal` means that the constraint residual and merit-bound gap satisfy the numerical tolerance. `maxit` denotes an unresolved iteration or numerical limit; `infeasible` denotes a request with no feasible allocation.

Expected merit. This is the allocation-weighted mean of the supplied mating scores: cross means in this analysis and usefulness in the following analysis.

Reference F0 and F1. `F0` is the fixed reference value. Here it is the mean of the complete kinship matrix. `F1` is the allocation-weighted kinship between the parents of the allowed matings.

Group F2 and upper limit. `F2` is $c'Kc$ for the parental gene contributions implied by the mating fractions. Its upper limit follows from the supplied `dF` values and reference.

dF1 and dF2. These express changes relative to `F0` as

$$dF_1 = \frac{F_1 - F_0}{1 - F_0}, \qquad dF_2 = 1 - \sqrt{\frac{1 - F_2}{1 - F_0}}.$$

The first follows the supplied change interval, and the second is bounded by its upper endpoint.

Merit upper bound and gap. The upper bound limits the best attainable merit. The gap is its difference from the achieved merit.

Maximum constraint residual. This reports the largest numerical constraint violation used to assess feasibility.

The allocation column gives fractions of the produced progeny, summing to one. For example, 0.0107 corresponds to approximately 1.07% of the progeny. The merit column contains each mating's supplied score. These continuous fractions are not integer family sizes.

Parental gene contributions are one half of the sum of mating allocations across their two parental slots. The same parent in both slots would supply the full contribution of that self mating; this candidate table contains only distinct parents.

Different mating fractions can give the same parental contributions and the same additive mean. The mean-based optimum can therefore have several equally good mating allocations.

8.3 Usefulness-based mate allocation

The second mating table retains every pair and its bounds. Only the merit column changes to the existing expected superior-progeny score. That score was calculated above for ten retained individuals among 100 produced DH progeny per family, using the same fitted effects and finite-family approximation. The method optimizes the fixed supplied scores; it does not recalculate the selection intensity from the allocation fractions. A later integer allocation with different family sizes would require scores appropriate to those sizes. The contribution constraints refer to nominal gene shares before selection within families.

```
> oma.usefulness.matings <- oma.mean.matings
> oma.usefulness.matings$merit <- cross.scores$es
```

The second call uses the same parent and mating bounds, kinship matrix, reference and `dF` interval as the mean analysis. Its weighted merit therefore represents expected usefulness under that declared scoring model.

```
> oma.usefulness <- st.oma(
+     parents = oma.parents,                    # Parent contribution bounds
+     matings = oma.usefulness.matings,         # Allowed crosses and scores
+     K = cross.K,                              # Kinship matrix
+     dF = c(0, 0.01),                          # F1 change range; F2 change cap
+     base = "RM"                               # Random-mating reference
+ )
> oma.usefulness
```

```
OMA: Optimum mate allocation
Status: optimal
Expected merit: 2.6831
Reference F0: >-0.0001; F1: <0.0001
Group F2: 0.0199; upper limit: 0.0199
dF1: <0.0001; dF2: 0.0100
Merit upper bound: 2.6831; gap: <0.0001
Maximum constraint residual: <0.0001

Continuous mating allocations
  Parent 1  Parent 2  Allocation   Merit
  1         2           <0.0001  2.5882
  1         3           <0.0001  2.3099
  1         4           <0.0001  2.3755
  1         5           <0.0001  2.4677
  1         6           <0.0001  2.6464
  1         7           <0.0001  2.5471
  1         11          <0.0001  2.6839
  1         14           0.0227  2.7570
  1         16          <0.0001  2.3367
  1         21          <0.0001  2.5356
  [420 table rows omitted]
  38        40          <0.0001  2.5211
  38        41          <0.0001  2.7087
  39        40          <0.0001  2.1226
  39        41          <0.0001  2.3380
  40        41          <0.0001  2.3849
```

The returned allocations are continuous fractions. An operational crossing plan additionally needs integer family and progeny counts satisfying its own constraints. These fractions are not rounded automatically and are not an integer family plan for the simulation in Section 6.4. The optimum cross-selection interface remains the corresponding choice when the requested result is a list of equal-sized integer families.

9 Proportional loss of expected genetic standard deviation

The published PropSD criterion of Fernández-González et al. (2026) describes proportional loss of expected genetic standard deviation relative to a fixed reference population. The expectation follows a Gaussian genetic model with covariance proportional to the supplied matrix. Its implementation uses exact moments of sample variance and a two-moment Gamma approximation to expected sample standard deviation. This quantity differs from the standard deviation of the fitted parental merits and from group relatedness. Appendix A gives the mathematical definitions.

The complete 30-parent matrix defines the reference sample here. Each family contributes two parental occurrences to the selected sample, including repeated occurrences of a parent used in several families. Such occurrences represent the same genetic random variable. They are not independent copies.

```
> parent.slots <- c(usefulness.plan$crosses$P1Name,
+                   usefulness.plan$crosses$P2Name)
> diversity.loss <- propsd(
+     G = cross.K,                              # Complete reference covariance
+     parents = parent.slots                    # Parental occurrences in families
+ )
> diversity.loss
```

```
Proportional loss of expected sample SD
PropSD: 0.0195
Expected SD: reference 1.0044; selected 0.9848
Sample occurrences: reference 30.0000; selected 30.0000
Gaussian moments; two-moment Gamma approximation
```

The loss is one minus the selected-to-reference ratio of approximate expected standard deviations. A value of 0.03 represents a 3% loss under this model; negative values are possible when the selected sample has greater expected variation. The reference remains fixed when the selected occurrences change. The reported approximation does not describe realized ancestry or variation after selection within the descendant families.

9.1 A diversity constraint in cross selection

The optional diversity list constrains the same numerical estimate during cross selection. The explicit `family.slots` declaration specifies the sample convention used above. The complete matrix can also contain reference parents outside the eligible crossing panel. The objective remains usefulness and the existing contribution and group-relatedness limits still apply.

```
> diversity.plan <- st.cross.plan(
+     merit = cross.merit,                      # Parental genetic values
+     K = cross.K,                              # Kinship matrix
+     n.crosses = 15,                           # Number of distinct parent pairs
+     n.progeny = 100,                          # Progeny per family
+     theta = 0.02,                             # Group relatedness limit
+     upper = 0.1,                              # Maximum parental contribution
+     scores = cross.scores,                    # Precomputed cross scores
+     criterion = "usefulness",                 # Objective: maximum usefulness
+     accuracy = 1,                             # Within-family ranking accuracy
+     diversity = list(                         # Additional diversity constraint
+         G = cross.K,                          # Complete reference covariance
+         max = 0.025,                          # Strict upper limit on PropSD
+         sample = "family.slots"               # Two parental occurrences per family
+     ),
+     control = list(                           # Search settings
+         starts = 4,                           # Number of search starts
+         maxit = 5000                          # Proposals per search start
+     )
+ )
> diversity.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: best.found
Families: 15; progeny per family: 100; total progeny: 1500
Expected mean: 2.3702
Expected usefulness: 2.6513; accuracy: 1.0000
Group relatedness: 0.0198; requested limit: 0.0200
PropSD (family slots): 0.0223; strict upper limit: 0.0250
Average within-cross relatedness: -0.0542
Objective bound: 2.8596; optimality gap: 0.2083
Maximum constraint residual: 0.0000
```

```
Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  1         32            100  2.2892  -0.2974      2.5902
       2  1         41            100  2.2763  -0.0154      2.5384
       3  2         5             100  2.2998  -0.0626      2.6027
       4  2         22            100  2.5211  -0.0164      2.7462
       5  3         37            100  1.9585  -0.1444      2.2316
       6  5         14            100  2.5115  -0.1087      2.7796
       7  6         11            100  2.6042  -0.0972      2.8350
       8  7         34            100  2.3908  -0.0552      2.7169
       9  11        33            100  2.4373  -0.0699      2.6973
      10  14        35            100  2.4060  -0.0877      2.6929
      11  14        40            100  2.2893  -0.0594      2.6583
      12  16        38            100  2.2713  -0.0772      2.5509
      13  21        22            100  2.4657   0.2835      2.7443
      14  27        34            100  2.4414   0.0685      2.7007
      15  29        38            100  2.3899  -0.0731      2.6850
```

The diversity line reports the computed loss and its strict upper limit. Feasibility requires the displayed estimate before rounding to be below 0.025. The reported search status retains its usual meaning: a feasible heuristic result does not establish a global optimum. The constraint applies to the declared parental sample and the Gamma estimate, rather than a guaranteed loss of genetic variation in a future selected population.

10 Allocation of progeny to fixed crosses

The parent pairs can remain fixed while the produced progeny counts vary. The following allocation distributes 1,500 progeny across the 15 families and retains ten individuals per family. The mean of the selected descendants therefore weights each family by its ten retained individuals. Selection intensity is recalculated for every proposed integer family size from the cross mean and segregation variance; an earlier score at 100 progeny is not reused at other family sizes. The default selection differential is the finite-family approximation of Burrows (1972).

```
> progeny.allocation <- st.cross.allocate(
+     plan = diversity.plan,                    # Fixed parent pairs and constraints
+     scores = cross.scores,                    # Cross means and segregation variances
+     total.progeny = 1500,                     # Total produced progeny budget
+     n.selected = 10,                          # Retained descendants per family
+     min.progeny = 10,                         # Minimum produced count per family
+     max.progeny = 1500,                       # Maximum produced count per family
+     accuracy = 1                              # Within-family ranking accuracy
+ )
> progeny.allocation
```

```
Fixed-pair progeny allocation
Status: optimal
Families: 15; produced: 1500; retained: 150
Predicted retained mean: 2.652; response: 0.2816
Objective upper bound: 2.652; response gap: 6.661e-16
Group relatedness, nominal retained: 0.01982; produced: 0.02027
Family: parents; produced / retained; predicted retained mean
1: 1 x 32; 106 / 10; 2.595
2: 1 x 41; 94 / 10; 2.534
3: 2 x 5; 106 / 10; 2.608
4: 2 x 22; 83 / 10; 2.734
5: 3 x 37; 98 / 10; 2.23
6: 5 x 14; 96 / 10; 2.777
7: 6 x 11; 85 / 10; 2.824
8: 7 x 34; 113 / 10; 2.728
9: 11 x 33; 94 / 10; 2.693
```

```
10: 14 x 35; 102 / 10; 2.694
11: 14 x 40; 126 / 10; 2.681
12: 16 x 38; 100 / 10; 2.551
13: 21 x 22; 99 / 10; 2.743
14: 27 x 34; 94 / 10; 2.696
15: 29 x 38; 104 / 10; 2.688
```

The family table gives produced and retained counts together with the recalculated expected selected-family means. The expected selected mean combines them using retained counts. The budget and integer bounds refer to produced counts. The optimum and numerical gap refer to allocation among these fixed pairs under the finite-family approximation; they do not establish a joint optimum over parent pairs and progeny numbers.

The inherited parental contribution and group-relatedness constraints use nominal retained-family gene shares. Produced-progeny shares and their group relatedness are separate reported quantities. Neither vector predicts the actual genomic ancestry after within-family selection. The family-slot diversity sample remains unchanged when only produced counts vary.

The low-level simulation commands in Section 6.4 already read the produced count of each family from `planned.crosses`. The assignment `planned.crosses <- progeny.allocation$crosses` therefore provides the variable counts to that same crossing and DH-generation loop. The equal-family execution adapter retains its equal-size requirement.

An alternative calculation uses finite-sample normal order statistics following Okoye et al. (2026, Equations 4–8). The same fixed families, retained counts and total budget are used below. The selection method changes the expected selection differential; it does not change the fitted cross means or segregation variances. The automatic allocator compares the predicted gain from one additional individual in each family.

```
> progeny.order.allocation <- st.cross.allocate(
+     plan = diversity.plan,                    # Fixed parent pairs and constraints
+     scores = cross.scores,                    # Cross means and segregation variances
+     total.progeny = 1500,                     # Total produced progeny budget
+     n.selected = 10,                          # Retained descendants per family
+     min.progeny = 10,                         # Minimum produced count per family
+     max.progeny = 1500,                       # Maximum produced count per family
+     accuracy = 1,                             # Within-family ranking accuracy
+     selection.method = "order.statistics",    # Finite-normal selection expectation
+     control = list(                           # Allocation settings
+         algorithm = "auto"                    # Automatic discrete allocator
+     )
+ )
> progeny.order.allocation
```

```
Fixed-pair progeny allocation
Selection: finite normal order statistics; allocator: greedy
Status: numerically.optimal
Families: 15; produced: 1500; retained: 150
Predicted retained mean: 2.652; response: 0.2817
Estimated objective upper bound: 2.652; response gap: 4.841e-14
Group relatedness, nominal retained: 0.01982; produced: 0.02027
Family: parents; produced / retained; predicted retained mean
1: 1 x 32; 106 / 10; 2.595
2: 1 x 41; 94 / 10; 2.534
3: 2 x 5; 106 / 10; 2.608
4: 2 x 22; 83 / 10; 2.734
5: 3 x 37; 98 / 10; 2.23
6: 5 x 14; 96 / 10; 2.777
7: 6 x 11; 85 / 10; 2.825
8: 7 x 34; 113 / 10; 2.728
9: 11 x 33; 94 / 10; 2.693
10: 14 x 35; 102 / 10; 2.694
11: 14 x 40; 126 / 10; 2.681
```

```
12: 16 x 38; 100 / 10; 2.551
13: 21 x 22; 99 / 10; 2.744
14: 27 x 34; 94 / 10; 2.696
15: 29 x 38; 104 / 10; 2.688
```

The selection-method and algorithm lines identify the scored objective and the allocator used. Produced counts again sum to 1,500 and each family retains ten individuals. All 15 produced counts equal those of the Burrows-based allocation above. The predicted retained mean now uses the expected mean of the ten greatest values in a finite normal sample. The status `numerically.optimal` means that the estimated numerical gap meets the requested tolerance. Quadrature error estimates are not rigorous interval bounds. The mathematical integer optimum is conditional on fixed pairs, fixed retained counts and the stated normal model; it does not establish biological accuracy or a joint optimum over pairs and counts.

A common population evaluation summarizes the same fixed family counts. The saved selection method and prediction accuracies are inherited from the allocation. The kinship reference remains the named matrix used in planning, and the inherited constraints are checked again from the supplied counts and parents. G1 denotes the selected DH descendants of the current crosses. G2 would follow an additional cycle of crossing selected DH lines, DH derivation and selection.

```
> cross.population <- st.cross.evaluate(
+     plan = progeny.order.allocation,          # Fixed families and produced counts
+     K = cross.K                               # Original named kinship reference
+ )
> cross.population
```

```
Cross-plan population evaluation
Status: evaluated; selection: order.statistics
Produced: 1500 in 15 families; retained: 150 in 15 families
G1 retained mean: 2.652; retained-weight baseline: 2.37; response: 0.2817
Retained between-family variance: 0.01875; retained total variance:
unavailable
Produced marginal variance, within / between / total: 0.02753 / 0.0206 /
0.04812
Nominal relatedness, family / produced / retained: 0.01982 / 0.02027 /
0.01982
Selected genomic diversity and G2 forecast: unavailable without
genetic.model
Family: parents; produced / retained; predicted retained mean
1: 1 x 32; 106 / 10; 2.595
2: 1 x 41; 94 / 10; 2.534
3: 2 x 5; 106 / 10; 2.608
4: 2 x 22; 83 / 10; 2.734
5: 3 x 37; 98 / 10; 2.23
6: 5 x 14; 96 / 10; 2.777
7: 6 x 11; 85 / 10; 2.825
8: 7 x 34; 113 / 10; 2.728
9: 11 x 33; 94 / 10; 2.693
10: 14 x 35; 102 / 10; 2.694
11: 14 x 40; 126 / 10; 2.681
12: 16 x 38; 100 / 10; 2.551
13: 21 x 22; 99 / 10; 2.744
14: 27 x 34; 94 / 10; 2.696
15: 29 x 38; 104 / 10; 2.688
```

The status `evaluated` identifies completion of this population calculation. The produced marginal variance, 0.04812, describes the genetic value of a randomly sampled candidate from the mixture of families. It combines the within-family term 0.02753 and the between-family term 0.02060. The selected G1 mean, approximately 2.652, instead weights each family by its retained count. Its response, 0.2817, uses the unselected family means on these same retained-count weights as the baseline, approximately 2.370.

The family, produced and retained contribution bases describe nominal parental shares. Equal retained counts make the family and retained relatedness equal here, approximately 0.01982, while the produced-count weights give 0.02027. These values use the original kinship matrix. Within-family selection can alter genomic ancestry even when the retained family counts remain fixed. The selected between-family term is 0.01875, but selected within-family and total variance require additional moment calculations. The genetic value and segregation variance after crossing the selected DH lines for a further breeding cycle remain unavailable. Evaluation checks the supplied population; it does not optimize the plan or alter its counts.

10.1 Comparison with simulated selected descendants

A separate repeated-simulation check uses the same 30 founders, 243 markers, fitted effects and fixed pairs as `progeny.allocation`. The table retains the Burrows-based predictions and allocation above, with equal families as the comparison. Each of 200 paired cycles produces 1,500 DH descendants under each allocation and retains the ten highest genetic values per family. Each paired cycle uses the same seed under both allocations, with seeds 740001 through 740200. The table reports the predicted selected mean, simulated mean across cycles, and Monte Carlo standard error (MCSE). The difference uses the paired cycles.

```
> allocation.validation
```

```
               Quantity   Predicted    Simulated         MCSE
1        Equal families 2.651319231 2.6477827455 0.0005485178
2    Allocated families 2.651797776 2.6483499520 0.0005351892
3 Allocation difference 0.000478545 0.0005672066 0.0003372739
```

The simulated allocation difference is positive. Its approximate 95% Monte Carlo interval runs from −0.000094 to 0.001228 and includes zero. This run therefore does not establish a positive allocation benefit at that precision. The equal-count and allocated simulated means are approximately 0.00354 and 0.00345 below their Burrows predictions, respectively. That discrepancy concerns the selection approximation for this fitted-effect model, rather than a failure of the integer allocation optimum. Re-evaluation of these same equal and allocated counts with finite-normal order statistics increases their predicted means by approximately 0.0000317340 and 0.0000317367, respectively. These changes are much smaller than the discrepancy from simulation, whose explanation therefore remains open. The same marker effects score the predictions and simulations, so the comparison assesses model consistency and does not provide independent field validation. These repeated simulations are validation work, separate from the ordinary short companion example.

11 Reproductive roles and parent capacities

The same maize markers and fitted effects also support a smaller example with the first six parents of the crossing panel. Marker-data-set copying and restriction keep the scores specific to those six candidates. The kinship submatrix retains the scale of the original 30-parent reference; its allele-frequency reference is not re-estimated after subsetting.

```
> practical.ids <- names(cross.merit)[1:6]
> practical.K <- cross.K[practical.ids, practical.ids]
> st.copy.marker.data(
+     "practical.parents",                    # Six-parent copy for this policy
+     "cross.parents"                         # Existing real marker panel
+ )
> st.restrict.marker.data(
+     ind.list = practical.ids,               # Six parents from the real panel
+     data.set = "practical.parents"          # Six-parent marker data set
+ )
> gs.set.effects(
```

```
+     cross.effects,                          # Fitted effects for the same markers
+     data.set = "practical.parents"          # Six-parent marker data set
+ )
> gs.cross.eval.mu(
+     data.set = "practical.parents"          # Recomputed cross means
+ )
> gs.cross.eval.va(
+     pop.type = "DH",                        # Segregation among derived DH lines
+     data.set = "practical.parents"          # Six-parent marker data set
+ )
> gs.cross.eval.es(
+     N = 2,                                  # Retained individuals per family
+     G = 20,                                 # Produced individuals per family
+     data.set = "practical.parents"          # Six-parent marker data set
+ )
> practical.scores <- gs.cross.info(
+     data.set = "practical.parents"          # Scores for exactly the six parents
+ )
```

The first three candidates form an operational seed-parent pool, and the other three form an operational pollen-parent pool. These imposed crossing roles are management choices for this example, rather than biological sex categories. The requested plan uses exactly five parents: two seed parents and three pollen parents. Equal family use applies to the selected seed parents, without requiring all eligible seed parents to be selected.

```
> practical.parents <- data.frame(
+     parent = practical.ids,                 # Actual parent identifiers
+     pool = rep(c("seed", "pollen"), each=3),   # Operational parent groups
+     seed = rep(c(TRUE, FALSE), each=3),     # First group eligible as seed parent
+     pollen = rep(c(FALSE, TRUE), each=3)    # Second group eligible as pollen parent
+ )
> practical.limits <- data.frame(
+     parent = NA_character_,                 # Group-level parent counts
+     pool = c(NA, "seed", "pollen"),         # All parents or the named group
+     role = c("all", "seed", "pollen"),      # Count parents in the stated role
+     basis = "family.slots",                 # Distinct-family participation
+     measure = "parents",                    # Number of distinct used parents
+     lower = c(5, 2, 3),                     # Exact total, seed and pollen counts
+     upper = c(5, 2, 3)                      # Equal lower and upper bounds
+ )
> practical.equal.use <- data.frame(
+     pool = "seed",                          # Seed-parent group
+     role = "seed",                          # Seed use only
+     basis = "family.slots"                  # Equal integer family use
+ )
```

The following search considers four distinct families with twenty produced descendants each. The existing usefulness scores correspond to retaining two descendants per family. The relatedness ceiling is nonrestrictive here, so the example emphasizes the explicitly declared practical restrictions.

```
> practical.plan <- st.cross.plan(
+     merit = cross.merit[practical.ids],     # Merits of the six eligible parents
+     K = practical.K,                        # Original-scale kinship submatrix
+     n.crosses = 4,                          # Four distinct biological families
+     n.progeny = 20,                         # Equal produced count at this stage
+     theta = max(diag(practical.K)),         # Nonrestrictive relatedness ceiling
+     scores = practical.scores,              # Scores for the six-parent panel
+     criterion = "usefulness",               # Expected selected-family performance
+     constraints = list(                     # Intersecting practical restrictions
+         parents = practical.parents,        # Pool labels and role eligibility
+         limits = practical.limits,          # Exact used-parent counts
```

```
+         equal.use = practical.equal.use        # Equal use among selected seed parents
+     ),
+     control = list(                            # Search settings
+         method = "exact"                       # Bounded exhaustive cross search
+     )
+ )
> practical.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: optimal
Practical constraints: feasible
Checks: 10 pass; 0 fail; 0 unresolved; 0 unavailable
Seed/pollen directions: 4
Families: 4; progeny per family: 20; total progeny: 80
Expected mean: 2.3390
Expected usefulness: 2.5743; accuracy: 1.0000
Group relatedness: 0.1785; requested limit: 1.0405
Average within-cross relatedness: -0.0544
Objective bound: 2.5743; optimality gap: <0.0001
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  1         5              20  2.2508  -0.0689      2.4556
       2  1         6              20  2.4093  -0.1318      2.6332
       3  2         4              20  2.2375   0.0140      2.5376
       4  2         6              20  2.4583  -0.0311      2.6710
```

All ten practical checks pass. The selected seed parents 1 and 2 each occur in two families, and the pollen parents 4, 5 and 6 cover the four families. The exact five-parent total includes two seed and three pollen parents; parent 3 remains eligible but unused. The saved direction map identifies reproductive roles independently of the canonical order of the unordered biological parent pair. Status `optimal` qualifies the completed exact search for this declared model and its supplied scores.

The next allocation adds a produced-capacity limit of thirty descendant slots for each pollen parent. Produced counts remain variable while the four chosen families, reproductive directions and two retained individuals per family remain fixed. A produced slot counts one descendant receiving that parent's pollen contribution; it does not count laboratory crossing operations.

```
> practical.capacity <- data.frame(
+     parent = practical.ids[4:6],               # Individual pollen parents
+     pool = NA_character_,                      # Individual selectors need no pool filter
+     role = "pollen",                           # Pollen contribution only
+     basis = "produced",                        # Produced descendant slots
+     measure = "uses",                          # Integer incidences before division
+     lower = 0,                                 # No minimum produced pollen use
+     upper = 30                                 # At most thirty slots per pollen parent
+ )
> practical.allocation <- st.cross.allocate(
+     plan = practical.plan,                     # Fixed families and declared directions
+     scores = practical.scores,                 # Cross means and segregation variances
+     total.progeny = 80,                        # Total produced descendant budget
+     n.selected = 2,                            # Retained descendants per family
+     min.progeny = 2,                           # Minimum produced count per family
+     max.progeny = 40,                          # Maximum produced count per family
+     accuracy = 1,                              # Within-family ranking accuracy
+     selection.method = "order.statistics",     # Finite-normal selection expectation
+     constraints = list(                        # Additional produced-capacity restriction
+         limits = practical.capacity            # Individual pollen-parent capacities
+     )
```

```
+ )
> practical.allocation
```

```
Fixed-pair progeny allocation
Selection: finite normal order statistics; allocator:
constrained.enumeration
Status: numerically.optimal
Families: 4; produced: 80; retained: 8
Predicted retained mean: 2.574; response: 0.2352
Estimated objective upper bound: 2.574; response gap: 4.408e-14
Group relatedness, nominal retained: 0.1785; produced: 0.1744
Practical checks: feasible; pass 13, fail 0, unresolved 0, unavailable 0
Family: parents; produced / retained; predicted retained mean
1: 1 x 5; 21 / 2; 2.459
2: 1 x 6; 15 / 2; 2.614
3: 2 x 4; 29 / 2; 2.571
4: 2 x 6; 15 / 2; 2.653
```

The produced family counts 21, 15, 29 and 15 sum to eighty; two individuals per family give eight retained descendants. Pollen parent 6 contributes to the second and fourth families, using $15 + 15 = 30$ produced slots and exactly reaching its capacity. Pollen parents 4 and 5 use 29 and 21 slots, respectively. All thirteen practical checks pass.

With twenty descendants per family, pollen parent 6 would have forty produced uses, exceeding the new capacity. The earlier plan with equal family sizes therefore has a different feasible set; a difference in its predicted selected mean is not a gain comparison against a feasible baseline under the new restriction.

This produced-capacity constraint couples family sizes. The reported allocator is `constrained.enumeration`; the independent family-increment theorem in the appendix does not certify this constrained allocation. Status `numerically.optimal` uses the estimated objective bound, with response gap approximately 4.4×10^{-14} here. The qualification applies to the stated fixed-pair problem. Parent-count and equal-family-use restrictions inherited from the crossing plan remain in force alongside the new produced-capacity rows.

```
> practical.population <- st.cross.evaluate(
+     plan = practical.allocation               # Independent inherited-constraint check
+ )
> practical.population
```

```
Cross-plan population evaluation
Status: evaluated; selection: order.statistics
Produced: 80 in 4 families; retained: 8 in 4 families
G1 retained mean: 2.574; retained-weight baseline: 2.339; response: 0.2352
Retained between-family variance: 0.005243; retained total variance:
unavailable
Produced marginal variance, within / between / total: 0.02308 / 0.008781 /
0.03186
Nominal relatedness, family / produced / retained: 0.1785 / 0.1744 / 0.1785
Practical checks: feasible; pass 13, fail 0, unresolved 0, unavailable 0
Selected genomic diversity and G2 forecast: unavailable without
genetic.model
Family: parents; produced / retained; predicted retained mean
1: 1 x 5; 21 / 2; 2.459
2: 1 x 6; 15 / 2; 2.614
3: 2 x 4; 29 / 2; 2.571
4: 2 x 6; 15 / 2; 2.653
```

The evaluation recomputes all thirteen inherited and additional practical checks from the actual family counts and declared directions; all pass. This feasibility result is separate from the optimizer's source status.

The G1 retained mean is approximately 2.574, with response 0.2352 above the retained-weight baseline of approximately 2.339.

The produced marginal variance is 0.03186. The retained between-family term, 0.005243, is not a retained total variance; that total remains unavailable. Nominal role, family, produced and retained shares retain their declared denominators. They do not predict genomic ancestry after selection; selected genomic diversity and the additional G2 breeding-cycle forecast remain unavailable.

12 Joint choice of crosses and family sizes

The preceding allocation changes produced counts for a fixed set of families. The `progeny` argument of `st.cross.plan` allows the chosen biological families and their produced counts to vary together. The retained count remains four per active family here. Selection acts within each family, so all three families contribute equally to the selected G1 mean even when their produced counts differ.

This comparison uses the same six maize founders and their fifteen unordered candidate pairs, with ninety produced DH descendants and twelve retained descendants. Every calculation uses the same cross means, segregation variances, finite-normal selection expectation, accuracy and original kinship reference. The seed and pollen policy from Section 11 is not imposed in this separate example. Thus the candidate graph is identical across the three designs. The stored `es` column was computed for another family size; the new `progeny` branch recomputes selection response from `mu`, `va` and the actual counts instead of using that column.

The equal-family baseline chooses three pairs with thirty produced descendants each. The fixed-pair calculation then redistributes the same ninety produced descendants over those pairs, within bounds four to fifty. The joint calculation allows the pairs and counts to vary under those same bounds. The equal-family design is feasible for both broader problems, and the fixed-pair allocation is supplied as a checked starting incumbent for the joint search. The production bounds declared by the standalone allocator replace the source design's equal-size bounds; inherited parental, pairing and practical restrictions still apply.

```
> joint.equal.time <- system.time({
+     joint.equal <- st.cross.plan(
+         merit = cross.merit[practical.ids],     # Merits of the same six founders
+         K = practical.K,                        # Original-scale named kinship reference
+         n.crosses = 3,                          # Three distinct biological families
+         theta = max(diag(practical.K)),         # Nonrestrictive family-slot ceiling
+         scores = practical.scores,              # Means and variances for all fifteen pairs
+         criterion = "usefulness",               # Expected selected G1 mean
+         accuracy = 1,                           # Within-family ranking accuracy
+         progeny = list(                         # Explicit common production contract
+             total.progeny = 90,                 # Total produced DH descendants
+             n.selected = 4,                     # Retained descendants per active family
+             min.progeny = 30,                   # Equal families in this baseline
+             max.progeny = 30,                   # Exactly thirty produced per family
+             selection.method =                  # Finite-normal selection expectation
+                 "order.statistics",
+             control = list(                     # Shared allocation settings
+                 algorithm = "auto"              # Qualified automatic integer allocator
+             )
+         ),
+         control = list(                         # Family-set search settings
+             method = "exact"                    # Bounded exhaustive family-set search
+         )
+     )
+ })
> joint.equal
```

```
Joint cross and progeny allocation
Criterion: usefulness; status: numerically.optimal
Families: 3.0000; produced: 90.0000; retained per family: 4.0000
Selection: order.statistics; produced bounds per active family: 30.0000 to
```

```
  30.0000
Baseline mean: 2.4146; selected response: 0.2112; selected mean: 2.6258
Group relatedness (family slots): 0.2940; limit: 1.0405
Objective upper bound: 2.6258; centered criterion gap: <0.0001
Solver: exact; evaluated family/direction sets: 1.0000
Cumulative allocation work: 0.0000; ceiling: 100000000.0000
Original-input feasibility: passed
Finite-normal integration errors and objective bounds are estimated, not
  rigorous interval certificates.

Chosen crosses and actual counts
  Family  Parent 1  Parent 2  Produced  Retained  Selected mean  Response
       1  1         6               30         4         2.6219    0.2126
       2  2         6               30         4         2.6603    0.2019
       3  5         6               30         4         2.5954    0.2191
```

The three selected pairs are 1×6, 2×6 and 5×6. Each produces thirty descendants and retains four, giving the stated budgets of ninety and twelve. The retained-weight baseline is approximately 2.414646; selection increases it to approximately 2.625834. The family-slot relatedness, 0.2940, is below the original-scale ceiling 1.0405, and all original-input checks pass.

Status `numerically.optimal` qualifies the completed search and its estimated finite-normal errors. The evaluated-set count records actual inner allocation solves; valid bounds can exclude other family-search branches. It is not the size of the allowed candidate graph.

```
> joint.fixed.time <- system.time({
+     joint.fixed <- st.cross.allocate(
+         plan = joint.equal,                     # The three baseline families remain fixed
+         scores = practical.scores,              # Identical means and segregation variances
+         total.progeny = 90,                     # Identical total production budget
+         n.selected = 4,                         # Identical retention per family
+         min.progeny = 4,                        # Every retained line has a produced line
+         max.progeny = 50,                       # At most fifty produced per family
+         accuracy = 1,                           # Identical ranking accuracy
+         selection.method = "order.statistics", # Identical finite-normal scoring
+         control = list(                         # Fixed-family allocation settings
+             algorithm = "auto"                  # Qualified automatic integer allocator
+         )
+     )
+ })
> joint.fixed
```

```
Fixed-pair progeny allocation
Selection: finite normal order statistics; allocator: greedy
Status: numerically.optimal
Families: 3; produced: 90; retained: 12
Predicted retained mean: 2.626; response: 0.2112
Estimated objective upper bound: 2.626; response gap: 5.695e-14
Group relatedness, nominal retained: 0.294; produced: 0.2951
Family: parents; produced / retained; predicted retained mean
1: 1 x 6; 30 / 4; 2.622
2: 2 x 6; 29 / 4; 2.658
3: 5 x 6; 31 / 4; 2.598
```

The fixed-pair allocator changes the produced counts to thirty, twenty-nine and thirty-one. It moves one produced descendant from 2×6, the pair with the smallest segregation variance of these three, to 5×6, the pair with the largest. All three families still retain four descendants. Their retained weights and baseline therefore remain unchanged. Only the within-family response changes in this comparison.

The nominal retained relatedness remains 0.2940, while the produced-count value becomes approximately 0.2951. These are different incidence bases, not predictions of genomic diversity after selection.

The estimated response gap qualifies the fixed-family count solution; it does not by itself qualify the choice of those families.

```
> joint.plan.time <- system.time({
+     joint.plan <- st.cross.plan(
+         merit = cross.merit[practical.ids],     # The same six eligible founders
+         K = practical.K,                        # The same kinship reference and scale
+         n.crosses = 3,                          # Identical number of active families
+         theta = max(diag(practical.K)),         # Identical family-slot relatedness ceiling
+         scores = practical.scores,              # The same fifteen candidate families
+         criterion = "usefulness",               # Expected selected G1 mean
+         accuracy = 1,                           # Identical ranking accuracy
+         progeny = list(                         # Joint pair and production decisions
+             total.progeny = 90,                 # Identical total production budget
+             n.selected = 4,                     # Identical retention per active family
+             min.progeny = 4,                    # Lower produced count per family
+             max.progeny = 50,                   # Upper produced count per family
+             selection.method =                  # Identical finite-normal scoring
+                 "order.statistics",
+             control = list(                     # Shared allocation settings
+                 algorithm = "auto"              # Qualified automatic integer allocator
+             )
+         ),
+         control = list(                         # Family-set search settings
+             method = "exact",                   # Bounded exhaustive family-set search
+             start = joint.fixed                 # Checked feasible allocation as incumbent
+         )
+     )
+ })
> joint.plan
```

```
Joint cross and progeny allocation
Criterion: usefulness; status: numerically.optimal
Families: 3.0000; produced: 90.0000; retained per family: 4.0000
Selection: order.statistics; produced bounds per active family: 4.0000 to
  50.0000
Baseline mean: 2.4146; selected response: 0.2112; selected mean: 2.6259
Group relatedness (family slots): 0.2940; limit: 1.0405
Objective upper bound: 2.6259; centered criterion gap: <0.0001
Solver: exact; evaluated family/direction sets: 20.0000
Cumulative allocation work: 1560.0000; ceiling: 100000000.0000
Original-input feasibility: passed
Finite-normal integration errors and objective bounds are estimated, not
  rigorous interval certificates.

Chosen crosses and actual counts
  Family  Parent 1  Parent 2  Produced  Retained  Selected mean  Response
       1  1         6               30         4         2.6219    0.2126
       2  2         6               29         4         2.6580    0.1997
       3  5         6               31         4         2.5977    0.2214
```

The joint search returns the same three pairs and the same counts $(30, 29, 31)$. This example therefore has no additional predicted gain from changing the pairs after fixed-pair allocation. The exact outer search confirms that allocation within the full fifteen-pair candidate graph, subject to the stated numerical qualification. All original-input checks pass. The cumulative work is shared across the evaluated family sets, rather than reset for each proposal.

The following comparison uses `st.cross.evaluate` to recompute the three population summaries from their actual counts and stored scoring method. Its additional digits make the small predicted allocation difference visible.

```
> joint.equal.eval <- st.cross.evaluate(
+     plan = joint.equal                          # Recomputed equal-family population
```

```
+ )
> joint.fixed.eval <- st.cross.evaluate(
+     plan = joint.fixed                          # Recomputed fixed-pair population
+ )
> joint.plan.eval <- st.cross.evaluate(
+     plan = joint.plan                           # Recomputed joint population
+ )
> joint.metrics <- c("retained.baseline", "response", "retained.mean")
> joint.comparison <- rbind(
+     Equal = unlist(joint.equal.eval$population[joint.metrics]),
+     Fixed = unlist(joint.fixed.eval$population[joint.metrics]),
+     Joint = unlist(joint.plan.eval$population[joint.metrics])
+ )
> colnames(joint.comparison) <- c("Baseline", "Response", "Selected")
> joint.comparison <- cbind(
+     joint.comparison,
+     Seconds = c(joint.equal.time["elapsed"], joint.fixed.time["elapsed"],
+                 joint.plan.time["elapsed"])
+ )
> joint.comparison
```

```
      Baseline  Response Selected Seconds
Equal 2.414646 0.2111881 2.625834   0.003
Fixed 2.414646 0.2112196 2.625866   0.038
Joint 2.414646 0.2112196 2.625866   0.061
```

All three designs use ninety produced and twelve retained descendants, with the same fitted moments, ranking accuracy and allowed graph. Moving from equal counts to optimized counts increases the predicted selected mean by approximately 3.1411×10^{-5}; the joint result matches the fixed-pair result here. The improvement is wholly in the within-family response because the pairs and retained weights remain the same.

The seconds column records one elapsed-time observation for each optimizer call, excluding the shared data preparation and subsequent evaluation. The values describe this small example. The genetic gain is a prediction under the supplied finite-normal model; the simulation checks above and the later selected-genomic and G2 models address separate biological questions. The joint planner at this stage optimizes G1 with a fixed retained count per active family.

13 Allocation of produced and retained descendants

The preceding joint planner retains four descendants from each selected family. The `total.selected` option instead fixes the total retained population and allows its distribution across families to vary. Production remains fixed at ninety descendants, and retention remains fixed at twelve. Selection still acts within each family. A pooled selection of the twelve highest scores across all ninety produced descendants is a different rule.

The comparison uses the same six maize founders, fifteen candidate pairs, cross moments, ranking accuracy, finite-normal scoring and original kinship reference as Section 12. Each active family produces between four and fifty descendants. Every family retains at least one descendant, with the additional constraint that retention cannot exceed its own production. The fixed-retention solution therefore remains feasible and supplies a checked starting plan, including its actual produced and retained counts. The family-slot parental and relatedness restrictions retain their original meanings; no reproductive-role policy is added here.

The selected-individual weights are now $r_e/12$. A change in retention can therefore change the retained-weight baseline as well as the within-family selection response. The nominal retained founder shares can also differ from family-slot shares. These differences are reported separately from selected genomic ancestry, which requires the later selection-aware model.

The search below has explicit work limits. Its result may be a checked best found plan with a remaining gap; the broader two-budget problem is not qualified by the earlier fixed-retention optimum.

```
> retention.time <- system.time({
+     retention.plan <- st.cross.plan(
+         merit = cross.merit[practical.ids],     # The same six eligible maize founders
+         K = practical.K,                        # Original-scale named kinship reference
+         n.crosses = 3,                          # Three distinct produced families
+         theta = max(diag(practical.K)),         # The same family-slot relatedness ceiling
+         upper = 0.5,                            # The same family-slot parental bound
+         scores = practical.scores,              # The same fifteen cross means and variances
+         criterion = "usefulness",               # Retained-individual G1 mean
+         accuracy = 1,                           # Identical within-family ranking accuracy
+         progeny = list(                         # Fixed production and retention totals
+             total.progeny = 90,                 # Ninety produced DH descendants
+             total.selected = 12,                # Twelve retained DH descendants
+             min.progeny = 4,                    # Lower produced count for each active pair
+             max.progeny = 50,                   # Upper produced count for each active pair
+             min.selected = 1,                   # At least one retained from each family
+             max.selected = 12,                  # Retained upper bound before r <= n
+             selection.method =                  # Identical finite-normal scoring
+                 "order.statistics",
+             control = list(                     # Cumulative allocation settings
+                 algorithm = "auto",             # Supported bounded integer allocation
+                 max.transitions = 1e7           # Shared work ceiling for the whole call
+             )
+         ),
+         control = list(                         # Reproducible bounded family search
+             method = "heuristic",               # Best-found search over family choices
+             start = joint.plan,                 # Actual produced and retained counts
+             seed = 1,                           # Local search seed
+             starts = 1,                         # One start from the checked incumbent
+             maxit = 2000                        # Ceiling on outer search iterations
+         )
+     )
+ })
> retention.plan
```

```
Joint production and retention planning
Criterion: usefulness; status: best.found
Produced families: 3.0000; produced descendants: 90.0000; retained
  descendants: 12.0000
Families with retained descendants: 3.0000; selection: order.statistics
Default produced bounds: 4.0000 to 50.0000; retained bounds: 1.0000 to
  12.0000
Baseline mean: 2.4392; selected response: 0.2046; selected mean: 2.6439
Family-slot relatedness: 0.2940; family-slot limit: 1.0405
Actual produced / retained relatedness: 0.2957 / 0.3162
Objective upper bound: 2.7513; centered criterion gap: 0.1074
Solver: heuristic; evaluated family/direction sets: 267.0000
Cumulative allocation work: 10000000.0000; ceiling: 10000000.0000
Original-input feasibility: passed
Finite-normal integration and weighted objective bounds are estimated, not
  rigorous interval certificates.

Produced families and actual n / r
  Family  Parent 1  Parent 2  Produced  Retained  Selected mean  Response
       1  1         6               30         3         2.6389    0.2296
       2  2         6               50         8         2.6520    0.1937
       3  5         6               10         1         2.5937    0.2174
```

The selected pairs remain 1×6, 2×6 and 5×6, with produced counts $(30, 50, 10)$ and retained counts $(3, 8, 1)$. Both budgets and all original-input checks pass. The 2×6 family reaches the produced upper bound of fifty and supplies eight of the twelve retained descendants; 5×6 supplies one. The printed means therefore receive weights $3/12$, $8/12$ and $1/12$, rather than equal family weights.

The retained-weight baseline increases to approximately 2.439239 while the within-family response decreases to approximately 0.204619. Their sum, approximately 2.643859, is greater than the fixed-retention prediction. The higher baseline more than offsets the lower response; variable retention does not improve the result merely by increasing every family's selection intensity.

Status `best.found` remains the optimizer's qualification. The cumulative ten-million work ceiling is reached, and the reported global upper bound leaves a gap of approximately 0.1074. This is the optimizer's remaining estimated bound gap, not a measured biological prediction error. The checked feasible incumbent is retained when that work limit stops the search.

```
> retention.population <- st.cross.evaluate(
+     plan = retention.plan                     # Independent actual-count population check
+ )
> retention.population
```

```
Cross-plan population evaluation
Status: evaluated; selection: order.statistics
Produced: 90 in 3 families; retained: 12 in 3 families
G1 retained mean: 2.644; retained-weight baseline: 2.439; response: 0.2046
Retained between-family variance: 0.0002599; retained total variance:
unavailable
Produced marginal variance, within / between / total: 0.01791 / 0.0009007 /
0.01881
Nominal relatedness, family / produced / retained: 0.294 / 0.2957 / 0.3162
Selected genomic diversity and G2 forecast: unavailable without
genetic.model
Family: parents; produced / retained; predicted retained mean
1: 1 x 6; 30 / 3; 2.639
2: 2 x 6; 50 / 8; 2.652
3: 5 x 6; 10 / 1; 2.594
```

The independent evaluation reproduces the G1 mean and confirms ninety produced and twelve retained descendants in three families. Family-slot relatedness remains approximately 0.2940 because the three pairs are unchanged. Nominal retained relatedness instead increases to approximately 0.3162, and produced relatedness is approximately 0.2957. The original family-slot ceiling is still checked on its stated basis.

The actual retained counts give nominal gene shares $1/8$, $1/3$, $1/24$ and $1/2$ for parents 1, 2, 5 and 6, respectively. Family-slot shares for parents 1, 2 and 5 are each $1/6$, while parent 6 occurs in every family and has share $1/2$. These differences describe the declared allocation. They do not predict the marker-level ancestry of the selected DH lines. Selected genomic diversity, selected total variance and the additional G2 cycle therefore retain their separate unavailable status here.

```
> retention.comparison <- rbind(
+     Fixed = c(
+         unlist(joint.plan.eval$population[joint.metrics]),
+         joint.plan.eval$relatedness$family.slots,
+         joint.plan.eval$relatedness$retained,
+         unname(joint.plan.time["elapsed"])
+     ),
+     Variable = c(
+         unlist(retention.population$population[joint.metrics]),
+         retention.population$relatedness$family.slots,
+         retention.population$relatedness$retained,
+         unname(retention.time["elapsed"])
+     )
+ )
> colnames(retention.comparison) <- c(
+     "Baseline", "Response", "Selected", "Family.q", "Retained.q", "Seconds"
+ )
> retention.comparison
```

```
         Baseline  Response Selected  Family.q Retained.q Seconds
Fixed    2.414646 0.2112196 2.625866 0.2939532  0.2939532   0.061
Variable 2.439239 0.2046192 2.643859 0.2939532  0.3161700   0.565
```

Allowing variable retention increases the predicted selected mean by approximately 0.017993 under the same total resources. The baseline rises by approximately 0.024593 while the response falls by approximately 0.006600. The family-slot metric stays unchanged, whereas the nominal retained metric reflects the unequal retained weights.

An independent numerical reference enumerated 455 family sets and 83,950 admissible two-budget assignments per set, or 38,197,250 candidates. It matches this selected plan and objective. This additional validation does not change the optimizer's printed `best.found` status or its own bound gap. Agreement concerns the numerical objective under the supplied normal model and fitted marker effects.

The seconds column contains one elapsed-time observation for each optimizer call. It excludes the common data preparation and subsequent evaluation. The variable-retention implementation uses the equivalent small-count normal integrals and shared cache described in the appendix; its work ceiling and requested objective remain explicit.

14 Selection-aware contributions and selected DH genotypes

Unequal retained counts change the nominal parental shares shown above. Selection within each family can also change which parent's chromosome segments occur in its retained descendants. The optional `genetic.model` combines the fitted additive effects, effect-allele coding, marker map and a declared within-family ranking law. It supports an analytic contribution prediction and an explicit assessment of actual DH origin paths under that same fixed genetic model.

The example uses the same six candidate founders, fifteen allowed pairs and production and retention budgets as Section 13. The marker data set `cross.parents` supplies the 243 complete model markers and all thirty rows for the fixed genomic-reference frequencies. Only the six eligible founders become candidate parents. Uniform weights describe origin averaged over all 243 markers, including the 109 markers with identical alleles across those six founders. They do not identify unobserved physical genome lengths.

The declared population is an immediate DH generation from the F1, with a cM Haldane map and no additional intermating. The index model combines standardized genetic value with independent normal noise at the requested accuracy. Here accuracy one ranks the fitted genetic values directly, with neutral random resolution of ties. The original supplied kinship matrix retains its scale and reference.

The first calculation reports selected contributions without adding a contribution cap. Its count-search problem and settings match the preceding calculation. The requested genotype scenarios run once after the final plan is accepted; they do not run for every candidate allocation.

```
> genetic.time <- system.time({
+     genetic.plan <- st.cross.plan(
+         merit = cross.merit[practical.ids],    # The same six candidate founders
+         K = practical.K,                       # The original named kinship reference
+         n.crosses = 3,                         # Three produced biological families
+         theta = max(diag(practical.K)),        # The original family-slot ceiling
+         upper = 0.5,                           # The original family-slot parent bound
+         scores = practical.scores,             # The same precise cross means and variances
+         criterion = "usefulness",              # Expected selected G1 mean
+         accuracy = 1,                          # Ranking by the fitted genetic value
+         progeny = list(                        # The same production and retention budgets
+             total.progeny = 90,                # Produced DH descendants
+             total.selected = 12,               # Retained DH descendants
+             min.progeny = 4,                   # Minimum produced from each active pair
+             max.progeny = 50,                  # Maximum produced from each active pair
+             min.selected = 1,                  # At least one retained from each family
+             max.selected = 12,                 # Retention also remains below production
+             selection.method =                 # Finite-normal selection prediction
```

```
+                 "order.statistics",
+             control = list(                     # Cumulative count-search settings
+                 algorithm = "auto",             # Supported bounded integer search
+                 max.transitions = 1e7           # Shared count-work ceiling
+             )
+         ),
+         genetic.model = list(                   # Optional additive genotype model
+             data.set = "cross.parents",         # Thirty-row fixed genomic reference
+             population = "DH",                  # Immediate F1-derived DH descendants
+             t = 0,                              # No additional intermating generation
+             map.function = "Haldane",           # cM map with independent crossovers
+             index.model =                       # Declared independent-noise index law
+                 "trait.independent.noise",
+             marker.weights = NULL,              # Uniform weights on all model markers
+             selected.founder.theta = NULL,      # Contributions reported without a new cap
+             scenarios = list(                   # Final genotype assessment only
+                 replicates = 250,               # Independent retained-population replicates
+                 seed = 12052026                 # Reproducible private scenario bank
+             ),
+             control = list(                     # Whole-call genetic resource settings
+                 max.cells = 1e7,                # Genetic workspace and snapshot budget
+                 max.work = 1e8                  # Cumulative genetic-work ceiling
+             )
+         ),
+         control = list(                         # The same reproducible family search
+             method = "heuristic",               # Bounded best-found family search
+             start = joint.plan,                 # The same actual starting counts
+             seed = 1,                           # The same local optimizer seed
+             starts = 1,                         # One checked starting plan
+             maxit = 2000                        # The same outer iteration ceiling
+         )
+     )
+ })
> genetic.plan
```

```
Joint production and retention planning
Criterion: usefulness; status: best.found
Produced families: 3.0000; produced descendants: 90.0000; retained
  descendants: 12.0000
Families with retained descendants: 3.0000; selection: order.statistics
Default produced bounds: 4.0000 to 50.0000; retained bounds: 1.0000 to
  12.0000
Baseline mean: 2.4392; selected response: 0.2046; selected mean: 2.6439
Family-slot relatedness: 0.2940; family-slot limit: 1.0405
Actual produced / retained relatedness: 0.2957 / 0.3162
Objective upper bound: 2.7513; centered criterion gap: 0.1074
Solver: heuristic; evaluated family/direction sets: 267.0000
Cumulative allocation work: 10000000.0000; ceiling: 10000000.0000
Original-input feasibility: passed
Selected-founder proxy (UCPC): 0.3398; cap: NA; qualified
Actual-genotype scenarios: 250.0000 of 250.0000 completed; complete
Finite-normal integration and weighted objective bounds are estimated, not
  rigorous interval certificates.

Produced families and actual n / r
  Family  Parent 1  Parent 2  Produced  Retained  Selected mean  Response
       1  1         6               30         3         2.6389    0.2296
       2  2         6               50         8         2.6520    0.1937
       3  5         6               10         1         2.5937    0.2174
```

The report-only model returns the same families, produced counts $(30, 50, 10)$ and retained counts $(3, 8, 1)$ as the preceding calculation. Its optimization status and estimated gap remain `best.found` and approximately 0.1074. All original-input checks pass.

The selected-founder proxy is approximately 0.3398. The printed missing cap means no additional contribution ceiling was requested; the proxy is a qualified available approximation here. It differs from the nominal retained relatedness, approximately 0.3162, because the analytic model predicts shifts within families as well as unequal retained weights. All 250 requested genotype replicates complete. Their estimates remain separate from the analytic search criterion and its bound.

```
> genetic.population <- st.cross.evaluate(
+     plan = genetic.plan                       # Frozen core inherited; no scenario request
+ )
> genetic.population
```

```
Cross-plan population evaluation
Status: evaluated; selection: order.statistics
Produced: 90 in 3 families; retained: 12 in 3 families
G1 retained mean: 2.644; retained-weight baseline: 2.439; response: 0.2046
Retained between-family variance: 0.0002599; retained total variance:
unavailable
Produced marginal variance, within / between / total: 0.01791 / 0.0009007 /
0.01881
Nominal relatedness, family / produced / retained: 0.294 / 0.2957 / 0.3162
Selected founder proxy: 0.3398; state 1;
joint.normal.regression.approximation
G2 forecast: unavailable without genetic.model$future
Family: parents; produced / retained; predicted retained mean
1: 1 x 6; 30 / 3; 2.639
2: 2 x 6; 50 / 8; 2.652
3: 5 x 6; 10 / 1; 2.594
```

The evaluator inherits the frozen marker/effect/map core and recomputes the analytic summaries for the actual current counts. Its selected-founder proxy agrees with the plan. This call makes no scenario request, so it does not rerun the source's 250 replicates. The missing analytic selected variance is distinct from the finite-genotype variance estimates in the explicit scenario group shown next. G2 still requires a further declared breeding cycle.

```
> genetic.actual <- genetic.plan$scenarios$population[
+     , c("metric", "estimate", "SE")
+ ]
> genetic.actual
```

```
                metric    estimate           SE
1             response 0.205584555 0.0014418686
2         selected.mean 2.644824036 0.0014418686
3 finite.panel.variance 0.003403229 0.0001184327
4       sample.variance 0.003712614 0.0001291993
5     marginal.variance 0.003922976           NA
6             genomic.q 0.412149773 0.0018921202
7        gene.diversity 0.129969593 0.0004966000
```

The scenarios estimate a selected mean of approximately 2.64482 with MCSE 0.001442. Its difference from the normal prediction, approximately 2.64386, is about 0.67 MCSE in this bank. The response row uses the same retained-weight baseline as the analytic calculation.

The expected finite-panel variance is approximately 0.003403; the expected empirical sample variance, with divisor eleven, is approximately 0.003713. The marginal one-member variance is approximately 0.003923, including variation between replicate means. Its SE is unavailable in this implementation, rather than zero.

The actual allele-based `genomic.q` estimate is approximately 0.4121 with MCSE 0.001892. It uses all retained pairs including diagonals and the fixed thirty-row reference. The founder proxy uses mean marker-origin fractions with the supplied founder matrix. Those definitions explain why its value, approximately

0.3398, is a separate quantity. Population gene diversity is approximately 0.12997 with MCSE 0.000497; each individual DH remains homozygous.

```
> origin.prediction <- genetic.plan$selected.contributions$families[
+     , c("P1Name", "P2Name", "P1")
+ ]
> origin.scenario <- genetic.plan$scenarios$families[
+     genetic.plan$scenarios$families$metric == "P1.origin",
+     c("P1Name", "P2Name", "estimate", "SE")
+ ]
> origin.comparison <- merge(
+     origin.prediction,
+     origin.scenario,
+     by = c("P1Name", "P2Name")
+ )
> names(origin.comparison) <- c(
+     "Parent1", "Parent2", "Predicted", "Simulated", "SE"
+ )
> origin.comparison$Difference <- origin.comparison$Simulated -
+                                 origin.comparison$Predicted
> origin.comparison
```

```
  Parent1 Parent2 Predicted Simulated          SE    Difference
1       1       6 0.4534461 0.4537888 0.003469925  0.0003426681
2       2       6 0.4517716 0.4530556 0.002221298  0.0012839967
3       5       6 0.4705766 0.4587819 0.006522184 -0.0117947327
```

The comparison joins the analytic and scenario parent-1 origins by the actual unordered family identifiers. Selection favors parent 6 in all three families under the analytic model. The three observed discrepancies are within approximately 1.9 MCSE in this bank. This is agreement for these counts and fitted model, while the UCPC formula retains its regression qualification. Values inside $[0,1]$ alone do not establish exact ancestry prediction; the appendix identifies the approximation and its references. Uniform weights describe the declared marker panel.

The selected-founder ceiling adds a constraint on the analytic `ucpc.founder.proxy`. The intermediate fixed-retention allocation inherits the frozen core while declaring its own count domain. Its actual pair/count table supplies a feasible start for the new capped joint problem. Source scenario results do not become a new execution request. The original family-slot and practical restrictions remain.

```
> genetic.fixed <- st.cross.allocate(
+     plan = genetic.plan,                        # Inherited frozen model for these pairs
+     scores = practical.scores,                  # The same precise family moments
+     total.progeny = 90,                         # The same total production budget
+     n.selected = 4,                             # A fixed-retention feasible comparator
+     min.progeny = 4,                            # The same produced lower bound
+     max.progeny = 50,                           # The same produced upper bound
+     accuracy = 1,                               # The same ranking accuracy
+     selection.method = "order.statistics"       # The same finite-normal prediction
+ )
> genetic.cap <- 0.33
> genetic.feasible.start <- genetic.fixed$crosses
> genetic.cap.time <- system.time({
+     genetic.capped.plan <- st.cross.plan(
+         merit = cross.merit[practical.ids],     # The same six candidate founders
+         K = practical.K,                        # The original named kinship reference
+         n.crosses = 3,                          # Three produced biological families
+         theta = max(diag(practical.K)),         # The original family-slot ceiling
+         upper = 0.5,                            # The original family-slot parent bound
+         scores = practical.scores,              # The same precise cross means and variances
+         criterion = "usefulness",               # Expected selected G1 mean
+         accuracy = 1,                           # Ranking by the fitted genetic value
```

```
  +         progeny = list(                         # The same production and retention budgets
  +             total.progeny = 90,                 # Produced DH descendants
  +             total.selected = 12,                # Retained DH descendants
  +             min.progeny = 4,                    # Minimum produced from each active pair
  +             max.progeny = 50,                   # Maximum produced from each active pair
  +             min.selected = 1,                   # At least one retained from each family
  +             max.selected = 12,                  # Retention also remains below production
  +             selection.method =                  # Finite-normal selection prediction
  +                 "order.statistics",
  +             control = list(                     # Cumulative count-search settings
  +                 algorithm = "auto",             # Supported bounded integer search
  +                 max.transitions = 1e7           # Shared count-work ceiling
  +             )
  +         ),
  +         genetic.model = list(                   # Optional additive genotype model
  +             data.set = "cross.parents",         # Thirty-row fixed genomic reference
  +             population = "DH",                  # Immediate F1-derived DH descendants
  +             t = 0,                              # No additional intermating generation
  +             map.function = "Haldane",           # cM map with independent crossovers
  +             index.model =                       # Declared independent-noise index law
  +                 "trait.independent.noise",
  +             marker.weights = NULL,              # Uniform weights on all model markers
  +             selected.founder.theta =            # Verified real-data proxy ceiling
  +                 genetic.cap,
  +             scenarios = NULL,                   # No genotype sampling during this call
  +             control = list(                     # Whole-call genetic resource settings
  +                 max.cells = 1e7,                # Genetic workspace and snapshot budget
  +                 max.work = 1e8                  # Cumulative genetic-work ceiling
  +             )
  +         ),
  +         control = list(                         # The same reproducible family search
  +             method = "heuristic",               # Bounded best-found family search
  +             start = genetic.feasible.start,     # Independently checked feasible cap incumbent
  +             seed = 1,                           # The same local optimizer seed
  +             starts = 1,                         # One checked starting plan
  +             maxit = 2000                        # The same outer iteration ceiling
  +         )
  +     )
  + })
  > genetic.capped.plan
```

```
Joint production and retention planning
Criterion: usefulness; status: best.found
Produced families: 3.0000; produced descendants: 90.0000; retained
  descendants: 12.0000
Families with retained descendants: 3.0000; selection: order.statistics
Default produced bounds: 4.0000 to 50.0000; retained bounds: 1.0000 to
  12.0000
Baseline mean: 2.4352; selected response: 0.2070; selected mean: 2.6422
Family-slot relatedness: 0.2940; family-slot limit: 1.0405
Actual produced / retained relatedness: 0.2953 / 0.2993
Objective upper bound: 2.7513; centered criterion gap: 0.1090
Solver: heuristic; evaluated family/direction sets: 275.0000
Cumulative allocation work: 10000000.0000; ceiling: 10000000.0000
Original-input feasibility: passed
Selected-founder proxy (UCPC): 0.3250; cap: 0.3300; qualified
Finite-normal integration and weighted objective bounds are estimated, not
  rigorous interval certificates.

Produced families and actual n / r
  Family  Parent 1  Parent 2  Produced  Retained  Selected mean  Response
       1  1         6               32         4         2.6264    0.2171
       2  2         6               50         7         2.6605    0.2022
       3  5         6                8         1         2.5774    0.2011
```

The fixed-retention comparator has proxy approximately 0.3225, below 0.33; the uncapped variable-retention result above has proxy approximately 0.3398. The capped search returns produced counts $(32, 50, 8)$ and retained counts $(4, 7, 1)$, with proxy approximately0.3250 and selected mean approximately 2.6422. Its actual-count and proxy checks pass. The result remains `best.found`, with its printed remaining gap.

The ceiling changes the feasible set and its predicted-mean tradeoff. Its metric is the analytic founder proxy on the original supplied K; the allele-based scenario metric retains its own definition. The model and cap remain attached for subsequent source evaluation or allocation, where current counts receive a new assessment.

A further crossing and selection cycle

The next breeding cycle begins with the actual selected DH lines, rather than only their predicted means or average founder contributions. Twelve selected G1 lines form six random disjoint pairs. Every line is used once, and each new F1 produces twenty G2 DH candidates from which two are retained. Thus the second cycle produces 120 DH lines and retains twelve. Distinct lines remain distinct possible mates even when their genotypes coincide or they come from the same original family.

The current decision retains the same six eligible founders, fifteen allowed pairs, three produced families, ninety produced descendants and twelve retained descendants. The original named kinship reference, family-slot ceiling and parental bounds remain. As in the preceding joint examples, the separate seed/pollen policy is not imposed on this undirected founder graph. Such original-founder rules would describe G1 operations and would not silently define roles for the new DH lines. All three designs have these same aggregate resources and restrictions. The equal-count reference has thirty produced and four retained per family; the G1-focused and future-focused searches use the common variable-count bounds. The latter two have the same decision domain.

The `future` criterion targets the selected G2 mean on a fixed training bank of actual genotype paths. The current G1 usefulness remains an analytic quantity with its existing finite-normal definition. The declared index law ranks actual fitted genetic values at accuracy one in both cycles. New F1s between the selected DH mosaics supply G2 segregation; a homozygous DH itself has no segregation variance.

```
> future.equal <- st.cross.allocate(
+     plan = joint.equal,                          # Actual equal-production baseline pairs
+     scores = practical.scores,                   # The same precise cross means and variances
+     total.progeny = 90,                          # Identical total produced descendants
+     n.selected = 4,                              # Explicit retained count in each family
+     min.progeny = 30,                            # Equal production in all three families
+     max.progeny = 30,                            # Exactly thirty candidates per family
+     accuracy = 1,                                # The same current ranking accuracy
+     selection.method = "order.statistics"        # The same analytic G1 selection method
+ )
> future.progeny <- list(
+     total.progeny = 90,                          # Current produced DH descendants
+     total.selected = 12,                         # Current retained DH descendants
+     min.progeny = 4,                             # Minimum produced per active family
+     max.progeny = 50,                            # Maximum produced per active family
+     min.selected = 1,                            # At least one current retained line
+     max.selected = 12,                           # Retention also stays below production
+     selection.method = "order.statistics",       # Analytic G1 selection expectation
+     control = list(                              # Current integer-count settings
+         algorithm = "auto",                      # Supported bounded count search
+         max.transitions = 1e7                    # Cumulative count-work ceiling
+     )
+ )
> future.policy <- list(
+     policy = "random.disjoint",                  # Every retained G1 line is mated once
+     n.progeny = 20,                              # Produced DHs from each new G2 F1
+     n.selected = 2,                              # Retained DHs from each G2 family
+     accuracy = 1,                                # G2 ranking by fitted genetic value
```

```
+     training = list(                          # Fixed bank used by future search
+         replicates = 32,                      # Complete replicates per candidate
+         seed = 0                              # Reproducible private training seed
+     ),
+     evaluation = list(                        # Separate held-out assessment bank
+         replicates = 256,                     # Requested whole common replicates
+         seed = 1                              # Role-separated evaluation seed
+     ),
+     comparators = list(                       # Actual equal-resource designs
+         Equal = future.equal,                 # Evaluated actual n=30/r=4 reference
+         G1 = genetic.plan                     # Existing G1-focused actual plan
+     )
+ )
> future.model <- list(
+     data.set = "cross.parents",               # Original thirty-row genomic reference
+     population = "DH",                        # Immediate F1-derived DH candidates
+     t = 0,                                    # No extra intermating generation
+     map.function = "Haldane",                 # cM map and independent crossovers
+     index.model =                             # Independent-noise ranking law
+         "trait.independent.noise",
+     marker.weights = NULL,                    # Uniform weights on all 243 markers
+     selected.founder.theta = NULL,            # No additional G1 founder-proxy cap
+     scenarios = NULL,                         # No separate single-cycle scenario bank
+     future = future.policy,                   # Actual two-cycle request declared above
+     control = list(                           # Whole-call genetic resource settings
+         max.cells = 1e7,                      # Shared genetic workspace ceiling
+         max.work = 2e8                        # Search and final assessment together
+     )
+ )
> future.time <- system.time({
+     future.plan <- st.cross.plan(
+         merit = cross.merit[practical.ids],   # The same six candidate founders
+         K = practical.K,                      # The original named kinship reference
+         n.crosses = 3,                        # Current produced biological families
+         theta = max(diag(practical.K)),       # The original family-slot ceiling
+         upper = 0.5,                          # The original parental family-slot bound
+         scores = practical.scores,            # Precise scores for all fifteen pairs
+         criterion = "future",                 # Actual selected G2 mean on training bank
+         accuracy = 1,                         # Current within-family ranking accuracy
+         progeny = future.progeny,             # Current production and retention budgets
+         genetic.model = future.model,         # Explicit core and next-cycle request
+         control = list(                       # Bounded joint family and count search
+             method = "heuristic",             # Best-found finite-bank search
+             start = genetic.plan,             # Actual checked G1 pairs and n/r counts
+             seed = 1,                         # Reproducible search seed
+             starts = 1,                       # Requested starting schedule
+             maxit = 2000                      # Current outer iteration ceiling
+         )
+     )
+ })
> future.plan
```

```
Joint planning for selected G2 descendants
Criterion: future; horizon: 2; status: best.found
G1 families / produced / retained: 3.0000 / 90.0000 / 12.0000
G2 families / produced / retained: 6.0000 / 120.0000 / 12.0000
Analytic G1 baseline / selected mean: 2.4392 / 2.6427
Training G1 selected mean / additional G2 response / G2 mean: 2.6457 /
  0.1535 / 2.7992
Training bank: complete; 32.0000 of 32.0000 complete
Held-out bank: complete; 256.0000 of 256.0000 complete
Held-out G2 selected mean / SE: 2.7926 / 0.0017
Actual held-out G2 genomic q / gene diversity: 0.4648 / 0.1156
Family-slot relatedness / limit: 0.2940 / 1.0405
```

```
Original-input G1 feasibility: passed
Selected-founder proxy (UCPC): 0.3398; cap: NA; qualified
Training-bank bound / gap: NA / NA

Chosen G1 crosses and actual counts
  Family  Parent 1  Parent 2  Produced  Retained
       1  1         6               34         3
       2  2         6               50         8
       3  5         6                6         1

Paired held-out contrasts (selected minus comparator)
  Comparator    Metric            Difference      SE  Reason
  G1.reference  G1.selected.mean     -0.0004  0.0004  available
  G1.reference  G2.selected.mean     -0.0004  0.0006  available
  G1.reference  G2.response         >-0.0001  0.0005  available
  Equal         G1.selected.mean      0.0195  0.0012  available
  Equal         G2.selected.mean      0.0057  0.0020  available
  Equal         G2.response          -0.0138  0.0016  available
  G1            G1.selected.mean     -0.0004  0.0004  available
  G1            G2.selected.mean     -0.0004  0.0006  available
  G1            G2.response         >-0.0001  0.0005  available
```

The printed objective concerns the complete training bank and horizon two. Its unselected G1 baseline, actual G1 response and additional G2 response have the component identity given in the appendix. Existing G1 gain and usefulness fields retain their analytic meaning. The returned search status and unknown expectation gap remain part of the result.

In the printed family order, produced counts are $(34, 50, 6)$ and retained counts are $(3, 8, 1)$. All 32 training replicates and 256 evaluation replicates complete. The search reaches its training-work allowance and retains `best.found` status. Completion of the evaluation bank does not close the optimization gap.

The separate evaluation bank compares the chosen design with the actual equal-count and G1-focused designs. Its whole common replicates generate both generations for every qualified comparator. This assessment does not reuse the earlier single-cycle scenario numbers or rank training proposals by held-out performance.

```
> future.comparison <- future.plan$future$population[
+     future.plan$future$population$design %in% c("Equal", "G1", "selected") &
+         future.plan$future$population$metric == "selected.mean",
+     c("design", "generation", "estimate", "SE")
+ ]
> future.comparison
```

```
     design generation estimate          SE
2  selected         G1 2.640892 0.001405756
9  selected         G2 2.792630 0.001744427
30    Equal         G1 2.621357 0.001376855
37    Equal         G2 2.786932 0.001768145
44       G1         G1 2.641251 0.001403634
51       G1         G2 2.793023 0.001823118
```

The rows compare selected G1 and selected G2 means under the same fitted model and declared next-cycle policy. Their SEs describe Monte Carlo precision across whole evaluation replicates. The analytic G1 prediction and these actual-genotype estimates remain separate quantities.

The future-focused design has G1 mean approximately 2.64089 and G2 mean 2.79263. The G1-focused design has corresponding means 2.64125 and 2.79302; the equal-count reference has means 2.62136 and 2.78693. Each design uses the same 256 held-out replicates.

```
> future.differences <- future.plan$future$contrasts[
+     future.plan$future$contrasts$comparator %in% c("Equal", "G1"),
```

```
+     c("comparator", "metric", "estimate", "SE")
+ ]
> future.differences
```

comparator	metric	estimate	SE
4 Equal	G1.selected.mean	1.953539e-02	0.0012363589
5 Equal	G2.selected.mean	5.698354e-03	0.0020294291
6 Equal	G2.response	-1.383703e-02	0.0016101313
7 G1	G1.selected.mean	-3.593500e-04	0.0003930399
8 G1	G2.selected.mean	-3.930633e-04	0.0005759057
9 G1	G2.response	-3.371328e-05	0.0004931103

Every difference is the future-focused plan minus the named comparator. The SE comes from the corresponding difference inside each common replicate, rather than combining unrelated aggregate SEs. The additional G2 response is the selected G2 mean minus the selected G1 mean for that same design. Interpretation of an advantage follows the actual paired estimate and its precision; a training winner need not improve the held-out expectation estimate.

Relative to the G1-focused design, current selected mean falls by approximately 0.000359 and the additional G2 response falls by 0.000034. The net G2 difference is -0.000393 with MCSE 0.000576, approximately 0.68 MCSE below the G1-focused comparator. The held-out assessment therefore provides no demonstrated advantage over the G1-focused design. The difference from equal counts is $+0.005698$ with MCSE 0.002029, approximately 2.8 MCSE above zero under this fitted model and declared simulation policy.

The training score is approximately 2.7992, while the held-out mean is 2.79263. This illustrates the risk of overfitting a small fixed training bank. The search optimizes its sampled objective; its selected design and numerical status do not certify the optimum of the unknown expectation. The seeds, requested bank sizes and constraints remain those declared before this comparison.

```
> future.pair.moments <- future.plan$future$pair.moments[
+     future.plan$future$pair.moments$design %in% c("Equal", "G1", "selected"),
+     c("design", "metric", "estimate", "SE")
+ ]
> future.pair.moments
```

design	metric	estimate	SE
1 selected	mean.variance	0.009226384	9.840492e-05
2 selected	mean.sqrt.variance	0.093664940	5.049969e-04
5 Equal	mean.variance	0.010722676	1.150516e-04
6 Equal	mean.sqrt.variance	0.101411655	5.589114e-04
7 G1	mean.variance	0.009218595	9.610039e-05
8 G1	mean.sqrt.variance	0.093710904	4.959518e-04

The variance is conditional on each actual selected-DH pair and describes the DH progeny of its new F1. Mean conditional variance and mean conditional SD summarize different aspects of that next-cycle segregation. The square root of mean variance does not replace mean SD or the actual G2 calculation.

For the selected design, mean conditional variance is approximately 0.009226 and mean conditional SD is 0.093665. The square root of the first number is approximately 0.09605, showing the distinction in this actual example.

```
> future.genomics <- future.plan$future$population[
+     future.plan$future$population$design %in% c("Equal", "G1", "selected") &
+         future.plan$future$population$generation == "G2" &
+         future.plan$future$population$metric %in% c("genomic.q", "gene.diversity"),
+     c("design", "metric", "estimate", "SE")
+ ]
> future.genomics
```

```
     design         metric  estimate           SE
13 selected      genomic.q 0.4648129 0.0020094849
14 selected gene.diversity 0.1156186 0.0005125546
41    Equal      genomic.q 0.4462531 0.0021661623
42    Equal gene.diversity 0.1226934 0.0005404730
55       G1      genomic.q 0.4639611 0.0020087833
56       G1 gene.diversity 0.1160426 0.0004912943
```

The G2 genomic quantities use realized selected allele frequencies and the original thirty-row reference. They are evaluated within each finite replicate before averaging. Population gene diversity remains distinct from individual DH heterozygosity, which is zero. Neither metric is a guaranteed consequence of a current founder-contribution prediction.

The future-focused G2 panel has estimated genomic coancestry 0.46481 and gene diversity 0.11562. The G1-focused panel gives 0.46396 and 0.11604. These are fixed-reference finite-panel estimates; the objective and current constraints do not impose a hard G2 genomic ceiling.

The same future objective can be assessed at declared G1 family-slot relatedness limits. Each frontier point replaces `theta` while retaining the other current constraints and budgets. Independent starts use the recorded standalone starting rules. One cumulative genetic-work budget covers the entire call, including every point's final assessment.

```
> future.theta.limits <- c(0.25, 0.30, 0.35)
> future.frontier.model <- future.model
> future.frontier.model$future$comparators <- NULL
> future.frontier.time <- system.time({
+     future.frontier <- st.cross.frontier(
+         merit = cross.merit[practical.ids],    # The same six candidate founders
+         K = practical.K,                       # Original-scale fixed kinship reference
+         n.crosses = 3,                         # Three current produced families
+         theta = max(diag(practical.K)),        # Replaced by each declared theta limit
+         upper = 0.5,                           # The same parental family-slot bound
+         scores = practical.scores,             # The same fifteen actual candidate pairs
+         criterion = "future",                  # Actual selected G2 training objective
+         accuracy = 1,                          # The same current ranking accuracy
+         progeny = future.progeny,              # The same current count domain and totals
+         genetic.model = future.frontier.model, # Same core and banks; no extra comparators
+         limits = future.theta.limits,          # Ordered G1 family-slot theta ceilings
+         metric = "theta",                      # Axis concerns nominal G1 family slots
+         warmstart = FALSE,                     # Each point keeps its standalone start
+         control = list(                        # The same declared search schedule
+             method = "heuristic",              # Bounded finite-bank search at each cap
+             seed = 1,                          # The same recorded search seed
+             starts = 1,                        # The same requested starting schedule
+             maxit = 2000                       # Per-point outer iteration ceiling
+         )
+     )
+ })
> future.frontier
```

```
Gain-diversity frontier
Criterion: future; horizon: 2.0000; limit metric: theta
Points: 3; recorded search status applies to each point
  Point   Limit  Status      Feasible  Training G2  Held-out G2      SE
      1  0.2500  best.found  TRUE           2.7919       2.7893  0.0018
      2  0.3000  best.found  TRUE           2.7963       2.7930  0.0018
      3  0.3500  best.found  TRUE           2.7992       2.7926  0.0017

Actual held-out G2 metrics and bank completion
  Point  G2 genomic q  G2 gene diversity  Completed  Requested
      1        0.4478             0.1199   256.0000   256.0000
```

```
      2         0.4640              0.1160   256.0000   256.0000
      3         0.4648              0.1156   256.0000   256.0000
```

The horizontal limit concerns nominal G1 family-slot relatedness on the original supplied kinship matrix. Actual selected G2 genomic coancestry and gene diversity retain their separate estimates. An alternative `selected.founder.theta` axis would concern the qualified G1 UCPC founder proxy and would also require that explicit label. Neither axis is a hard bound on actual G2 genomic coancestry. Infeasible, unavailable and work-limited points remain explicit; bounded-search and sample-based points need not form a smooth or monotone tradeoff curve.

All three points complete their 256 held-out replicates, while all retain `best.found` and a training-limit reason. Their held-out means are approximately 2.7893, 2.7930 and 2.7926 in the declared limit order. The estimates are not monotone in the cap. Their individual SEs do not supply a paired cross-point contrast or identify an optimum between the displayed limits. The first and last points have estimated G2 genomic coancestry approximately 0.4478 and 0.4648, respectively.

The immutable genetic core can be inherited for a new explicit future assessment. Execution banks, comparator commands and historical G2 results are request-specific. Ordinary source evaluation without a new future request executes no future bank and copies no stale assessment. Planner starts supply counts and pairs rather than an implicit genetic model. The forecast remains conditional on fitted marker effects and one declared additional cycle; Monte Carlo precision does not establish field performance or quantify marker-effect estimation uncertainty.

A Mathematical definitions and implemented algorithms

The definitions below distinguish parental merit, within-family segregation variance, nominal parental contribution and covariance-based diversity. All relationship matrices retain the reference population used to construct them. Numerical convergence describes the stated model and supplied scores; it does not establish the accuracy of the fitted marker effects. Equation numbers below refer to the cited publications. The sample conventions, constraints and numerical safeguards specific to SelectionTools are identified separately from the established statistical formulas and algorithmic principles.

A.1 Marker prediction, segregation variance and usefulness

For the biallelic additive model, $z_{i\ell} \in \{0,1,2\}$ is the dosage of the effect allele at marker ℓ in parent i, a_0 is the fitted intercept and a_ℓ is its marker effect. The parental prediction and expected cross mean are

$$g_i = a_0 + \sum_\ell a_\ell z_{i\ell}, \qquad \mu_{ij} = a_0 + \frac{1}{2}\sum_\ell a_\ell (z_{i\ell} + z_{j\ell}).$$

The cross mean is the dosage-notation form of Osthushenrich et al. (2017), Equations (1) and (4). Thus $\mu_{ij} = (g_i + g_j)/2$ for the same additive model. The score table retains the cross evaluator's supplied values; the planner does not replace them by values reconstructed from exported parental predictions.

For two completely homozygous parents, the marker contribution difference is $\delta_{ij,\ell} = a_\ell(z_{i\ell} - z_{j\ell})/2$. The linked segregation variance used for DH or completely homozygous SSD lines is

$$v_{ij} = \sum_c \sum_{\ell,k \text{ on } c} \rho_{\ell k}\, \delta_{ij,\ell}\, \delta_{ij,k}, \qquad \rho_{\ell k} = 2q_{\ell k} - 1.$$

This covariance contraction rewrites Osthushenrich et al. (2017), Equations (2) and (13), using the matching-origin relation in their Equation (16). The sum includes both orders of each marker pair and the diagonal. Different chromosomes have zero covariance. Here $q_{\ell k}$ is the probability that the two loci have matching parental origin. With recombination frequency r and t additional random-intermating generations, the implemented correlations are

$$\rho_{\mathrm{DH}}(r,t) = (1-2r)(1-r)^t, \qquad \rho_{\mathrm{SSD}}(r,t) = \frac{1-2r}{1+2r}(1-r)^t.$$

The DH and SSD correlations are $2q-1$ from Osthushenrich et al. (2017), Equations (20) and (23), respectively; their Equations (19) and (22) give the underlying linkage disequilibria. The SSD expression concerns the limit under repeated selfing; t is not the number of selfing generations. The map distance d in Morgans gives $r=(1-\exp(-2d))/2$ under Haldane or $r=\tanh(2d)/2$ under Kosambi. The first is the inverse of Osthushenrich et al. (2017), Equation (24), following Haldane (1919). The second is Kosambi (1943), Equation (6), p. 173. The unlinked inbred-parent calculation reduces to $\sum_\ell \delta_{ij,\ell}^2$. The C implementation forms the chromosome-specific recombination and origin-probability matrices and contracts them with the parental marker contributions, following Osthushenrich et al. (2017).

Normal truncation and its finite-family correction follow Burrows (1972, pp. 1091–1100). For retention fraction $p\in(0,1)$, normal truncation intensity is

$$i(p)=\frac{\phi\{\Phi^{-1}(1-p)\}}{p},$$

where ϕ and Φ are the standard normal density and distribution function. With R retained individuals from G produced individuals, the implemented Burrows (1972) finite-family approximation is

$$i_B(R,G)=i(R/G)-\frac{1-R/G}{2R(1+1/G)i(R/G)},\qquad 1\le R<G.$$

The boundary definitions are $i(1)=i_B(G,G)=0$. Family usefulness with selection accuracy $a\in[0,1]$ is

$$U_{ij}=\mu_{ij}+a\,i_B(R,G)\sqrt{v_{ij}}.$$

The usefulness criterion of Schnell and Utz (1975) combines the family mean and a within-family selection differential, as described by Osthushenrich et al. (2017, Introduction, pp. 1–2). Zhong and Jannink (2007, Introduction) discuss the corresponding superior-progeny criterion. The accuracy-scaled form above also appears in Fernández-González et al. (2026), Equation (16), with the intensity here supplied by the Burrows calculation. The SD, rather than the variance, enters the differential. The infinite population option uses $i(p)$ instead. The fixed-score planner uses $\mu_{ij}+a(es_{ij}-\mu_{ij})$ from its common supplied table. Changing its common progeny count does not recompute es_{ij}. Variable-size allocation uses μ_{ij} and v_{ij} to recompute the differential for each size.

A.2 Parental contributions and equal-sized integer families

For marker frequencies p_ℓ in the fixed reference, the genomic relationship matrix in this example is

$$W_{i\ell}=z_{i\ell}-2p_\ell,\qquad A=\frac{WW'}{2\sum_\ell p_\ell(1-p_\ell)},\qquad K=A/2.$$

This is the first genomic relationship construction of VanRaden (2008, p. 4416; unnumbered first-method formula), written for dosage coding $0,1,2$ rather than $-1,0,1$. The denominator must be positive. No diagonal adjustment is applied. The matrix can be singular and its off-diagonal entries can be negative. The quadratic form $Q(c)=c'Kc$ is a reference-dependent group-relatedness index. Dividing A by two does not itself give an absolute probability of identity by descent; Endelman (2025, Discussion following Equation (22)) also distinguishes genomic correlations from identity-by-descent probabilities.

The merit and coancestry terms in optimum contribution selection follow Meuwissen (1997), Equations [1] and [3], with $K=A/2$. His sex-specific normalizations in Equation [2] become one simplex for the supplied plant-parent pool here, with the stated lower and upper bounds. The problem is

$$\max_c g'c\quad\text{subject to}\quad \mathbf{1}'c=1,\quad l\le c\le u,\quad Q(c)\le\theta.$$

The inner C solver minimizes $Q(c)-2tg'c$ over the box-constrained simplex. This scalar tradeoff has the Lagrangian form of Meuwissen (1997), Equation [4]; the reduced-coordinate solver described here is the SelectionTools implementation. An active-set method uses a basis of free directions whose components sum to zero, an eigendecomposition of the reduced quadratic form, and a bounded line search. Rank-deficient

directions and active bounds receive explicit treatment. The outer search brackets and bisects the merit–relatedness tradeoff. A separate minimum-relatedness problem tests feasibility; the minimum-relatedness boundary also has an affine rank-based calculation.

For any candidate x, $\lambda \geq 0$, and box-constrained simplex C, convexity gives the merit upper bound

$$g'c \leq \lambda\{\theta + Q(x)\} + \max_{y \in C}(g - 2\lambda Kx)'y.$$

The remaining maximization is a bounded linear allocation. Feasibility residuals and the difference between such a bound and the achieved merit govern the reported numerical qualification; an iteration limit does not establish an optimum. The displayed bound follows by applying the first-order convexity inequality of Boyd and Vandenberghe (2004), Equation (3.2), p. 69, to Q and using $Q(c) \leq \theta$.

For M equal-sized distinct non-selfed families, $x_e \in \{0,1\}$ selects an allowed unordered pair $e = (i_e, j_e)$. The parental uses and nominal gene contributions are

$$\sum_e x_e = M, \qquad d_i = \sum_e x_e\{\mathbf{1}(i_e = i) + \mathbf{1}(j_e = i)\}, \qquad c_i = \frac{d_i}{2M}.$$

Joint planning maximizes $M^{-1}\sum_e x_e h_e$, with h_e the additive cross mean or supplied usefulness, subject to contribution bounds and $Q(c) \leq \theta$. Without cross scores its mean objective equals $g'c$. Fixed-contribution planning instead minimizes $M^{-1}\sum_e x_e K_{i_e j_e}$ while preserving every d_i. This is the kinship-minimizing mate-allocation objective described by Wellmann (2019, section "Mate allocation"; unnumbered objective), with the distinct-pair and integer-usage conventions stated here. A continuous contribution vector is therefore insufficient unless all its uses are integer-realizable in the allowed distinct-pair graph.

The exact search uses the branch-and-bound principle of Land and Doig (1960, Section 3), implemented here by include/exclude branches in score order. Available incident edges and remaining parental capacity prune impossible branches; sums of the best remaining edge scores bound objective values. Continuous contribution relaxations provide additional bounds when applicable. Automatic method selection compares an estimated enumeration workload with the node budget. The alternative heuristic combines multiple initial constructions with local edge changes and a local random-number stream. Fixed-contribution changes rewire two or three pairs while preserving all parental uses. Every retained incumbent is rechecked from its complete selected pairs and original matrix. The exact method can establish optimality after a completed search; a feasible heuristic incumbent has status `best.found`. An unresolved boundary or exhausted budget does not establish infeasibility. Relatedness is recomputed using integer uses and the original matrix, with numerical allowance

$$\varepsilon \max\left\{|\theta|, \sum_{i,j}|K_{ij}|c_i c_j\right\},$$

where ε is `control$tolerance`. Its computed roundoff enclosure must satisfy the bound including this allowance. A zero limit additionally requires a row-residual check independent of ε. These tolerance, roundoff and status rules are implementation safeguards.

A.3 Continuous optimum mate allocation

The continuous model follows the diploid mate-allocation framework of Endelman (2025, section "Convex optimization"). For allowed pairs e, mating fractions x_e are continuous and sum to one. The parental incidence matrix has entries $B_{ie} = \{\mathbf{1}(i_e = i) + \mathbf{1}(j_e = i)\}/2$, so $c = Bx$. This is the allowed-pair incidence representation of Endelman (2025), Equation (8). A self mating, when present in this interface, contributes one to its parent's incidence entry. With supplied mating scores h_e, `st.oma` maximizes $h'x$ under the supplied bounds on x and c. Its two kinship quantities are

$$F_1 = \sum_e x_e K_{i_e j_e}, \qquad F_2 = (Bx)'K(Bx).$$

These are the diploid specializations of Endelman (2025), Equations (14) and (19), with random mating in the following generation for F_2. The complete supplied $N \times N$ matrix defines the fixed reference

$$F_0 = \frac{1}{N}\sum_i (2K_{ii} - 1) \quad \text{for } \texttt{base="current"}, \qquad F_0 = \frac{1}{N^2}\sum_{i,j} K_{ij} \quad \text{for } \texttt{base="RM"}.$$

The current-population expression follows the diploid form of Endelman (2025), Equation (22), with $A = 2K$. The random-mating reference specializes his Equation (16) to uniform contributions. Using the complete supplied reference before restriction to eligible parents is the stated interface convention. The reference must be below one. For `dF=c(dL,dU)`, the constraints are

$$F_0 + (1-F_0)d_L \le F_1 \le F_0 + (1-F_0)d_U,$$
$$F_2 \le F_0 + (1-F_0)d_U(2-d_U).$$

A scalar `dF` sets an equality for F_1. The reported changes are

$$dF_1 = \frac{F_1 - F_0}{1-F_0}, \qquad dF_2 = 1 - \sqrt{\frac{1-F_2}{1-F_0}}.$$

The bounds and rate expressions are algebraic rearrangements of Endelman (2025), the one- and two-generation recurrences in Equations (10)–(11). The linear F_1 and quadratic F_2 restrictions remain distinct.

The C implementation uses a primal–dual interior-point method with the predictor–corrector principle of Mehrotra (1992). Positive inequality slacks and multipliers enter the perturbed complementarity equations. An affine predictor determines the centering parameter; the corrector includes the predictor's second-order product. A fraction-to-boundary rule and residual line search control the update. The Newton calculation uses a symmetric system of parental dimension, with the quadratic-constraint multiplier represented explicitly, and residual refinement. Scaling and rank decisions aid this calculation; feasibility is checked against the original supplied matrix and limits. Separate merit bounds and constraint residuals govern the status. The parent-dimension Newton formulation, residual checks and status qualification describe its implementation for this problem. Continuous mating fractions are not rounded into integer families.

A.4 Covariance-based proportional loss of SD

The `propsd` evaluator follows the published PropSD estimator of Fernández-González et al. (2026), Equations (14)–(15) and Appendix C, Equations (C1)–(C6). For the complete reference covariance-proportional matrix G, the model is $u \sim N(0, \tau^2 G)$. A literal selected sample $s = (s_1, \ldots, s_m)$ has covariance $G_s = G[s,s]$. Repeated identifiers repeat the same random genetic value; they do not create independent copies. The marker-frequency reference and the complete G remain fixed.

For any $n \times n$ covariance matrix V, $n \ge 2$, the centered matrix and ordinary sample variance with denominator $n-1$ are

$$H_n = I_n - \mathbf{1}\mathbf{1}'/n, \qquad C = H_n V H_n, \qquad S^2 = \frac{z' H_n z}{n-1}, \quad z \sim N(0, V).$$

The standard Gaussian quadratic-form identities of Magnus (1978), Lemmas 2.2–2.3, pp. 203–204, applied to $H_n/(n-1)$, give the exact moments

$$\mu_V = E(S^2) = \frac{\mathrm{tr}(C)}{n-1}, \qquad \nu_V = \mathrm{Var}(S^2) = \frac{2\sum_{i,j} C_{ij}^2}{(n-1)^2}.$$

These are also Fernández-González et al. (2026), Equation (15) and Appendix C, Equations (C1)–(C4), in centered-matrix notation. Their two-moment Gamma approximation, Equation (C5), has shape $\alpha_V = \mu_V^2/\nu_V$ and scale $\beta_V = \nu_V/\mu_V$. Its half moment approximates the expected sample SD:

$$e(V) = \sqrt{\beta_V}\,\frac{\Gamma(\alpha_V + 1/2)}{\Gamma(\alpha_V)}, \qquad \widehat{\mathrm{PropSD}} = 1 - \frac{e(G_s)}{e(G)}.$$

The Gamma half moment follows from Euler's integral (NIST DLMF, Equation 5.2.1), as worked out in Fernández-González et al. (2026), Equation (C6) and its following simplification. The common τ cancels. A general Gaussian quadratic form is a weighted sum of chi-squares, so Gamma moment matching is an approximation; it is exact when all nonzero eigenvalues of C are equal, including rank-one contrasts. The estimator is a ratio of expected SDs, rather than the expectation of a random ratio.

Integer multiplicities avoid materializing a replicated covariance matrix. With counts $k_i \geq 0$, $m = \sum_i k_i \geq 2$, $w = k/m$, $r = Gw$, and $D_{ij} = G_{ij} - r_i - r_j + w'Gw$, the same moments are

$$\mu_s = \frac{\sum_i k_i D_{ii}}{m-1}, \qquad v_s = \frac{2\sum_{i,j} k_i k_j D_{ij}^2}{(m-1)^2}.$$

These count expressions are an algebraic regrouping of Equation (15) of Fernández-González et al. (2026), with identical repeated-parent rows grouped together. Only positive-count parents enter the selected-sample sums. The total m matters as well as the normalized counts. Continuous contribution fractions alone therefore do not specify this statistic.

The C evaluator checks positive semidefiniteness. Each reference or selected sample uses its own active parental support to form anchored contrasts directly from the original supplied covariance entries, followed by occurrence-weighted centering and local contrast scaling. This preserves small selected-sample contrasts even when the reference contains much larger variances. The reference covariance model remains fixed; the local numerical scaling does not reconstruct marker frequencies. Extended-precision moment sums and separate reference and selected scale factors reduce cancellation and overflow. Moments reported in original covariance units can still exceed the numeric range of R. The SD calculation is

$$e(V) = \sqrt{\mu_V}\exp\{\ell(\alpha_V)\}, \qquad \ell(\alpha) = \log\Gamma(\alpha+1/2) - \log\Gamma(\alpha) - \tfrac{1}{2}\log\alpha.$$

For $\alpha > 128$, the implemented large-shape expansion is $\ell(\alpha) \simeq -1/(8\alpha) + 1/(192\alpha^3) - 1/(640\alpha^5)$; these coefficients specialize NIST DLMF, Equation 5.11.8, by subtracting the $h = 0$ expansion from the $h = 1/2$ expansion. The cutoff at 128 is an implementation choice; smaller shapes use the log-Gamma functions. A positive reference contrast variance is required. A selected sample with zero contrast variance has $e(G_s) = 0$ and loss one; a one-occurrence sample has undefined sample SD. Negative loss is retained when the selected expected SD exceeds the reference expected SD. Common positive covariance scaling and addition of a common random-intercept covariance leave the ratio unchanged when the covariance matrices remain valid.

The optional cross-planner convention `sample="family.slots"` assigns one occurrence to each parental slot of every selected family, so $k_i = d_i$ and $m = 2M$. The complete supplied diversity matrix defines the reference, including any additional reference parents. The constraint is the strict comparison $\widehat{\text{PropSD}} < b$, where b is `diversity$max`. The direction and strictness of this cutoff follow Fernández-González et al. (2026), Equation (18). It constrains the computed Gamma estimate; it is not an interval proof of biological diversity loss. Trait merits, $c'Kc$ and realized genomic ancestry after within-family selection remain different quantities. The exact search checks the criterion at complete candidate plans; objective bounds obtained by omitting it remain valid relaxations. The joint heuristic includes both relatedness and diversity violations when seeking feasibility and rechecks the criterion on retained plans. Fixed-contribution rewiring leaves the family-slot multiplicities, and hence this diversity criterion, unchanged.

A.5 Integer progeny allocation for fixed families

The produced counts G_e vary while parent pairs and retained counts R_e remain fixed. With total budget T, the integer constraints are

$$L_e \leq G_e \leq U_e, \qquad G_e \geq R_e \geq 1, \qquad \sum_e G_e = T.$$

Repeated unordered pairs are pooled into one biological family before selection intensity is calculated. Their retained counts and produced bounds are summed, and they have one common prediction accuracy a_e. The retained-population mean is

$$\overline{U}(G) = \sum_e \frac{R_e}{R_+}\{\mu_e + a_e\sqrt{v_e}\, i_B(R_e, G_e)\}, \qquad R_+ = \sum_e R_e.$$

For equal retained counts this is the mean-family usefulness objective of Fernández-González et al. (2026), Equation (17). The displayed retained-count weights and pooling convention define the allocation problem implemented here. The weighting follows retained individuals rather than produced counts. The constant

$\sum_e R_e\mu_e/R_+$ is excluded from allocation comparisons, which protects small response differences against large common merit offsets.

For family response utility $u_e(g) = (R_e/R_+)a_e\sqrt{v_e}\,i_B(R_e,g)$, exact dynamic programming uses the finite-stage principle of optimality of Bellman (1954, Section 2 and the basic recurrence in Equation (3.3), pp. 504–505), specialized to the remaining progeny budget:

$$D_0(0) = 0, \quad D_0(b \neq 0) = -\infty, \qquad D_e(b) = \max_{L_e \leq g \leq U_e} \{D_{e-1}(b-g) + u_e(g)\}.$$

The C recurrence works with additional counts above the family minima. Reachable budget ranges restrict states. Two objective rows and a family-by-budget traceback table recover the integer allocation. With $T' = T - \sum_e L_e$, the worst-case time is $O(M(T'+1)W)$ for maximum count-range width W; traceback storage is $O(M(T'+1))$. No concavity assumption is needed for this enumeration of feasible counts. The shared C intensity kernel also serves the existing finite-family cross evaluator. Upward-rounded recurrence sums give an upper bound on the finite computed utility table; a downward sum for the traced allocation gives the corresponding numerical gap. These bounds do not make the normal-selection or Burrows approximation exact. A completed calculation reports `optimal` only when its gap satisfies the numerical tolerance; an unresolved gap has status `numerically.unresolved`. A resource cap retains a deterministic feasible allocation with `resource.limit` and an infinite unresolved bound.

The same diploid parental-incidence relation as Endelman (2025), Equation (8), gives the nominal retained and produced shares separately

$$c_i^{\text{ret}} = \frac{\sum_e R_e\{\mathbf{1}(i_e = i) + \mathbf{1}(j_e = i)\}}{2R_+}, \qquad c_i^{\text{prod}} = \frac{\sum_e G_e\{\mathbf{1}(i_e = i) + \mathbf{1}(j_e = i)\}}{2T}.$$

The inherited parental bounds and relatedness limit apply to c^{ret}. Both relatedness values are reported; the produced one is a diagnostic in this allocation model. A family-slot PropSD constraint remains fixed because the parent pairs remain fixed. Neither nominal vector predicts the genomic ancestry of selected descendants. The optimum is conditional on the supplied pairs and retained counts, rather than a joint global optimum over pairs and progeny resources.

A.6 Finite-normal order statistics and discrete allocation

For independent standard-normal observations, $Z_{(r:g)}$ denotes the rth smallest value in a sample of size g. The optional `selection.method="order.statistics"` replaces $i_B(R,g)$ by

$$A(g,R) = \frac{1}{R}\sum_{r=g-R+1}^{g} E\{Z_{(r:g)}\}, \qquad A(R,R) = 0.$$

Writing ϕ and Φ for the standard-normal density and distribution function, respectively, the expectations are

$$E\{Z_{(r:g)}\} = \frac{g!}{(r-1)!(g-r)!}\int_{-\infty}^{\infty} z\phi(z)\Phi(z)^{r-1}\{1-\Phi(z)\}^{g-r}\,dz.$$

These are the finite-normal order-statistic and selected-mean definitions in Okoye et al. (2026), Equations (4)–(8). With fixed selected counts, the implemented objective is

$$\overline{U}_{\text{normal}}(G) = \sum_e \frac{R_e}{R_+}\{\mu_e + a_e\sqrt{v_e}A(G_e,R_e)\}.$$

The factor a_e is the within-family selection accuracy. For phenotypic selection with heritability H_e^2, it equals H_e. The expectation is exact for the stated independent normal-sampling model; numerical integration and a finite-locus genetic population remain separate qualifications.

The integer allocator uses the actual finite increment

$$\Delta_e(g) = \frac{R_e}{R_+}a_e\sqrt{v_e}\{A(g+1,R_e) - A(g,R_e)\}.$$

T_g denotes the Rth greatest observation among the original g draws, and X is an independent additional draw. The increase in the sum of the R greatest observations is $(X - T_g)_+$. Under a coupling that preserves the original observations, $T_{g+1} \geq T_g$, so its expected positive increment is nonnegative and decreases as g increases. Equivalently, with $Q(z) = 1 - \Phi(z)$,

$$A(g+1,R) - A(g,R) = \frac{1}{R}\int_{-\infty}^{\infty} Q(z)\Pr\{\mathrm{Bin}(g,Q(z)) \leq R-1\}\,dz.$$

This is the discrete diminishing-return property underlying the monotonicity argument of Okoye et al. (2026), Appendix 2, Equations (10)–(20). Starting from each lower bound L_e, selection of the greatest available $\Delta_e(g)$ gives a global optimum of this separable integer allocation problem: each family's decreasing increment sequence is selected in order, with upper bounds truncating the sequence. Ties can yield several optimal allocations. Continuous derivatives of an extension of $A(g,R)$ are not generally equal to these one-individual increments; the implementation ranks finite increments.

The fixed-retention C evaluator uses beta-quantile transformations of these integrals and adaptive quadrature split at the midpoint of the unit interval. For $g > R$, the nonnegative score representation is

$$A(g,R) = \frac{g}{R}E_{B\sim\mathrm{Beta}(R,g-R)}[\phi\{\Phi^{-1}(B)\}].$$

Normal symmetry gives $A(g,R) = (g-R)A(g,g-R)/R$, reducing near-full-retention calculations to the smaller selected count. Direct finite-increment evaluation avoids subtracting nearly equal selected means. A maximum heap retains the currently greatest family increment, and cached evaluations serve repeated count and retained-count combinations. The `greedy` allocator uses this discrete rule and `dp` uses the same normal-scoring objective in the budget recurrence above. The `auto` option starts with greedy allocation and can use the same objective in dynamic programming when the estimated gap remains unresolved and the resource limits permit it. Normal integrals and tail calculations are evaluated numerically. Their error estimates and floating-point uncertainty qualify the reported gap. Status `numerically.optimal` identifies an estimated gap meeting tolerance, rather than an interval-certified bound for the exact normal integrals. Otherwise the result reports `numerically.unresolved` or `resource.limit`, as appropriate. An order-statistic request never changes silently to Burrows scoring. The discrete theorem requires fixed pairs, fixed retained counts, equal cost per produced individual and separable count bounds. It does not extend automatically to coupled produced-share constraints or joint optimization of pairs and resources.

A.7 Common population evaluation

The evaluator pools repeated unordered parent pairs before scoring their produced counts G_e and retained counts R_e. Stored or explicitly supplied means, variances and accuracies must agree within a pooled family. The totals are $T = \sum_e G_e$ and $R_+ = \sum_e R_e$. A family with $R_e = 0$ has no weight in the selected summaries and no selection intensity is evaluated for it. With $T > 0$, the produced-family weights and marginal moments are

$$p_e = \frac{G_e}{T}, \qquad M_0 = \sum_e p_e\mu_e, \qquad W_0 = \sum_e p_e v_e,$$

$$B_0 = \sum_e p_e(\mu_e - M_0)^2, \qquad V_0 = W_0 + B_0.$$

This law-of-total-variance decomposition describes the genetic value of one uniformly sampled produced individual under the supplied family moments. It is not the expected empirical variance of all produced individuals or a segregation variance for an additional breeding cycle.

With $R_+ > 0$, the retained-count baseline and G1 mean are

$$M_{1,\mathrm{base}} = \sum_e \frac{R_e}{R_+}\mu_e, \qquad J_1 = \sum_{e:R_e>0}\frac{R_e}{R_+}U_e, \qquad R_1 = J_1 - M_{1,\mathrm{base}},$$

where U_e uses the explicitly resolved Burrows or finite-normal method. Thus R_1 retains the same family weights in the selected mean and its baseline. A difference from M_0, whose weights are G_e/T, is a separate

contrast. The available selected between-family term is

$$B_1 = \sum_{e:R_e>0} \frac{R_e}{R_+}(U_e - J_1)^2.$$

The analytic evaluator leaves selected within-family and total marginal variance unavailable. Explicit genotype scenarios in Section 14 estimate separately defined finite-genotype quantities under the declared model. They require second moments of the selected normal order statistics and a declared joint genetic-value/selection-score model. The expected empirical variance of the finite retained set further requires covariances among selected individuals. For $R_+ > 1$ its identity would be

$$E(S_1^2) = \frac{R_+}{R_+ - 1}\{V_{1,\text{marginal}} - \text{Var}(\overline{G}_1)\}.$$

The currently unavailable covariance terms are not replaced by an independence assumption. For $R_+ = 1$ sample variance is undefined. Zero total production or retention similarly yields unavailable summaries with an explicit reason rather than an assumed zero genetic value.

For C distinct produced families, the additional family-based nominal share is

$$c_i^{\text{family}} = \frac{1}{2C} \sum_{e:G_e>0} \{\mathbf{1}(i_e = i) + \mathbf{1}(j_e = i)\}.$$

The produced and retained shares remain as defined above. Each reported relatedness is $c'Kc$ for its stated basis on the complete original named reference, with zero shares for unused reference parents. Missing kinship input leaves relatedness unavailable. Inherited constraints retain their declared basis; family-slot diversity retains the complete supplied reference and its parental-occurrence convention.

Normalization, numerical scoring and centered accumulation of the mixture moments are C-owned. The evaluator independently recomputes supported budget, bound, contribution, relatedness and diversity residuals from the input plan. Evaluation status describes those checks and is separate from the source optimizer's convergence or optimality status. Counts are not repaired during evaluation.

Future-oriented allocation across breeding cycles, as investigated through simulation by Hamazaki and Iwata (2024), requires a further declared mating and selection policy. Here G2 means crosses between selected G1 DH lines followed by new DH derivation and selection. Current family means, segregation variances and parental kinship alone do not identify the selected genomic state needed for this forecast. Selected genomic ancestry, selected genomic diversity and G2 merit therefore remain unavailable in the initial evaluator.

A.8 Reproductive roles and practical constraints

Parent-count limits, role-specific restrictions, equal parental use and contribution capacities are established mate-selection choices; examples are described by Gorjanc and Hickey (2018). The associated manual groups them under "Mating/Crossing constraints". Its parent-count sections are "NumberOfParents", "NumberOfMaleParents" and "NumberOfFemaleParents". Equal use is described in "EqualizeContributions", while "LimitContributionsMin" and "LimitContributionsMax" specify contribution bounds. These manual sections have no numbered formulas. The equations below specify the incidence bases, pooling conventions and C search implemented in SelectionTools.

The biological response key remains an unordered nonself parent pair $e = \{i, j\}$. An identified directed record $a = (i_{\text{seed}}, j_{\text{pollen}})$ additionally records its reproductive roles. Equivalent reciprocal records pool their produced and retained counts before selection response is evaluated. Different directions do not create independent selection opportunities within the same biological family under this direction-neutral score model. The equal-family planner chooses at most one eligible direction for each selected biological pair. Explicit eligibility and allowed-pair restrictions intersect.

For basis b equal to produced, retained or operations, w_a^b is the produced count, retained count or explicitly supplied operation count, respectively. With $T_b = \sum_a w_a^b$, the role-specific incidences are

$$S_i^b = \sum_{a:\text{seed}(a)=i} w_a^b, \qquad L_i^b = \sum_{a:\text{pollen}(a)=i} w_a^b, \qquad D_i^b = S_i^b + L_i^b.$$

The overall and role-normalized nominal shares have different denominators:

$$c_i^b = \frac{D_i^b}{2T_b}, \qquad h_i^{b,\text{seed}} = \frac{S_i^b}{T_b}, \qquad h_i^{b,\text{pollen}} = \frac{L_i^b}{T_b}.$$

Each role share sums to one. Pool shares sum these values over the pool without re-normalizing within that pool. A produced-use limit counts descendant slots, while operations count only explicitly recorded crossing events. Operation counts are not inferred from produced counts. A zero basis total leaves its shares and relatedness unavailable.

For family slots, each distinct produced biological family contributes one parental incidence at each end. Its overall denominator is twice the number of those families. Role-specific family slots require a single unambiguous direction per family. A pooled family with multiple directions instead needs identified produced, retained or operation accounting for a role-specific restriction.

For a selector of pool $\mathscr{P}$ and role t, $Q_i^{b,t}$ equals D_i^b, S_i^b or L_i^b for all, seed or pollen use. Distinct used parents, aggregate uses and aggregate shares are, respectively,

$$N_{\mathscr{P},t}^b = \sum_{i\in\mathscr{P}} \mathbf{1}(Q_i^{b,t} > 0), \qquad U_{\mathscr{P},t}^b = \sum_{i\in\mathscr{P}} Q_i^{b,t}, \qquad H_{\mathscr{P},t}^b = \sum_{i\in\mathscr{P}} v_i^{b,t},$$

where $v = c$ for all roles and $v = h$ for a specific role. Each limits row places inclusive lower and upper bounds on one of these quantities. Equal use requires the positive $Q_i^{b,t}$ values in the selected subset to be identical integers. It does not force every eligible parent to be used. Exact parent counts use equal lower and upper integer bounds.

Count accumulation and feasibility comparisons use checked 64-bit integer arithmetic. Diagnostic decimal strings retain exact achieved counts and residuals; numeric fields become unavailable when such integers exceed the exact binary64 reporting range. A share constraint compares an integer numerator q with the actual represented numeric bound on denominator d:

$$\lceil \ell d \rceil \le q \le \lfloor u d \rfloor.$$

The product boundaries use portable two-limb integer arithmetic, preserving the represented binary64 bounds rather than their rounded printed values. Integer equality has no floating-point tolerance. Relatedness calculations retain their separate numerical uncertainty.

Inherited and explicit requests remain separate intersecting restrictions. The direction map is joined to the actual current family rows and their counts, so saved counts do not override an edited plan. The common checker recomputes every supported request from these inputs. A failed check is reported without repairing the plan; unavailable or unresolved checks do not count as passes.

Role-aware exact cross selection extends the bounded family search with eligible direction choices and biological-pair exclusivity. A fixed-pair allocation with coupled produced-use, share, relatedness or diversity constraints requires the constrained integer path. Separable increments can bound its objective, but their unconstrained optimum is not a certificate for the constrained feasible set. A completed search, valid bounds and qualified numerical comparisons determine the optimum claim. Exhausted work limits retain an honestly qualified result rather than a capacity-violating fallback or a false infeasibility proof.

The family-based nominal PropSD sample is `family.slots`. Produced counts define `produced.founder.slots`, while retained counts define `retained.founder.slots`. All three retain the complete named reference matrix and literal integer parental multiplicities. Produced and retained founder-slot totals are $2T$ and $2R_+$, respectively, within the supported literal-count domain. Each strict computed loss limit intersects any inherited family-slot limit. These nominal founder samples do not identify selected-DH genomic diversity or ancestry.

A.9 Joint family activation and produced counts

For allowed unordered biological families e, activation $y_e \in \{0,1\}$ and nonnegative integer produced count n_e satisfy

$$\sum_e y_e = C, \qquad \sum_e n_e = P, \qquad Ly_e \le n_e \le Uy_e, \qquad r_e = ry_e, \quad L \ge r > 0.$$

Consequently the retained budget is $R = Cr$. With reproductive roles, exactly one eligible directed alternative is chosen for each active biological family. Reciprocal alternatives retain the same biological mean and variance and do not create another selection opportunity. For selection intensity $a(n,r)$ defined above, the two supported criteria are

$$J_{\text{mean}} = \frac{1}{C}\sum_e y_e \mu_e, \qquad J_{G1} = \frac{1}{C}\sum_{e:y_e=1} \{\mu_e + \rho\sqrt{v_e}\, a(n_e,r)\}.$$

These are retained-count weights r/R, rather than produced-count weights n_e/P. The finite-family response follows the formulas and qualification in the preceding order-statistic subsection; the joint activation, resource constraints and search described here specify the SelectionTools implementation. The future-oriented progeny-allocation topic is also investigated by Hamazaki and Iwata (2024), whose simulation-based methodology does not define this G1 search.

The planner's parental contribution limits and $c'Kc \le \theta$ continue to use $c_i = d_i/(2C)$, where d_i is the number of active families containing parent i. They are independent of n_e. Additional practical constraints retain their explicitly declared family, produced or retained basis and the complete original relationship reference. Common retained counts make family and retained shares coincide here; their provenance remains separate. Operation counts are not inferred from descendant production.

The exact C search branches over biological family choices and eligible directions, with the original parental and practical constraints. A fixed-family subproblem allocates the produced budget. Its separable finite-normal path uses the discrete increments above; the Burrows path uses the corresponding bounded dynamic program. Coupled produced-use, share or diversity restrictions require their checked constrained search. The branch-and-bound principle follows Land and Doig (1960); the particular family graph, resource model, bounds and numerical qualification are the implementation described here. An exact fixed-family allocation does not establish an optimum over family choices by itself.

The family-search bound permits each candidate its greatest supported response independently, relaxing the shared production budget and practical restrictions. The finite-normal bound uses its qualified monotone count endpoint and estimated numerical error. A Burrows bound uses an enumerated count range; if no justified finite bound is available, the upper bound remains unknown. This relaxation can be loose, but it does not replace the actual-count objective or make a failed constraint feasible. Completed branch exclusion and estimated numerical comparisons determine the reported gap.

For larger searches, multiple starts and one-family, two-family and direction changes seek a better feasible plan, with a new bounded allocation for each proposed family set. Their result remains best found unless a global bound independently closes the gap. One call-owned C cache shares finite-normal intensities and direct marginals across all proposals. Its keys retain the actual counts, scoring method and numerical refinement. The allocation work ceiling is cumulative across the complete joint call; cached values reduce repeated integration but do not erase search work. An unfinished allocation leaves a family set unresolved instead of proving it infeasible. A feasible supplied incumbent remains available when a work limit stops the search.

The common evaluator recomputes final budget, bound and practical checks from the actual pairs, directions and counts. The printout distinguishes baseline, response and selected mean, as well as the objective bound and its numerical qualification. A finite-normal gap is based on estimated quadrature and rounding uncertainty and is not an interval certificate for the exact integrals or a validation of the finite-locus prediction.

A.10 Two-budget allocation of produced and retained descendants

For each allowed biological family, $y_e \in \{0,1\}$ indicates produced activation, and n_e, r_e are nonnegative integer produced and retained counts. The constraints are

$$\sum_e y_e = C, \qquad \sum_e n_e = P, \qquad \sum_e r_e = R > 0,$$

$$L_e y_e \le n_e \le U_e y_e, \qquad l_e y_e \le r_e \le u_e y_e, \qquad r_e \le n_e, \qquad L_e \ge 1.$$

An inactive family has zero produced and retained counts. An active family with $r_e = 0$ still consumes production and a family slot; its retained weight is zero. Such families require an explicit zero retained lower bound. The example above instead has $l_e = 1$ for every candidate. The optional bounds table joins these limits by unordered parent-pair keys, including candidates unused in the returned plan. Its specified cells replace common scalar limits. Directions share the biological bounds.

With the intensity $a(n,r)$ defined above and common accuracy ρ, the selected G1 mean and its two components are

$$M_{1,\text{base}} = \sum_e \frac{r_e}{R}\mu_e, \qquad \Delta_1 = \sum_{e:r_e>0} \frac{r_e}{R}\rho\sqrt{v_e}\,a(n_e,r_e),$$

$$J_{G1} = M_{1,\text{base}} + \Delta_1.$$

The mean criterion uses only $M_{1,\text{base}}$ in optimization. The usefulness formula combines the earlier family prediction with actual retained-individual weights. The finite-normal intensity is the selected order-statistic mean of Okoye et al. (2026), Equations (4)–(8); the Burrows alternative retains its preceding definition. The two-budget activation, weighting and constraints define the present implementation. No intensity or selected family mean is evaluated for $r_e = 0$. Its global utility contribution is exactly zero, while full retention has $a(n_e,n_e) = 0$.

One original-score anchor μ_0 permits utilities in mean units:

$$b_e(n,r) = \frac{r}{R}(\mu_e - \mu_0), \qquad d_e(n,r) = \frac{r}{R}\rho\sqrt{v_e}\,a(n,r), \qquad J_{G1} = \mu_0 + \sum_e \{b_e + d_e\}.$$

The C calculations retain baseline and response components separately and form their differences before comparison. They avoid unnormalized $r\mu_e$ intermediates and reconstruct the common anchor once, using the integer identity $\sum_e r_e = R$. This preserves the intended scale when large shared merits coexist with a small selection response.

For fixed families and separable restrictions, the recurrence follows the dynamic programming principle of Bellman (1954):

$$D_e(p,s) = \max_{(n,r)\in\mathscr{A}_e} \{D_{e-1}(p-n,s-r) + b_e(n,r) + d_e(n,r)\},$$

where $\mathscr{A}_e$ is the family-specific admissible integer domain, $D_0(0,0) = 0$, and impossible states are excluded. Traceback recovers both count vectors. The mean criterion omits d_e from its comparisons. Coupled produced or retained practical constraints need their original-input checks; a relaxed dynamic-programming solution is not automatically feasible.

The planner's parental limits and original relatedness ceiling still use family-slot shares $c_i^F = d_i/(2C)$. Actual retained shares instead use

$$c_i^R = \frac{1}{2R}\sum_{e:i\in e} r_e, \qquad q_R = (c^R)'Kc^R.$$

The complete original reference is common to q_R and the family-slot value q_F. Their difference reflects nominal representation under the declared counts; it does not describe marker-level selection within a family. Retained-basis practical limits, equal use, parent counts and diversity checks are recomputed with the actual r_e, while family-slot restrictions retain their original incidence basis.

A bound based on the unweighted average of family scores does not bound the variable-weight criterion. For the finite-normal calculation, a positive-retention family upper uses $a(U_e, \max(1,l_e))$: intensity increases with production at fixed retention and decreases with retention at fixed production. The maximum of such individual selected-mean uppers is a valid weak population upper because the positive retained weights sum to one. Numerical uncertainty remains included. A stronger retained-capacity relaxation is possible only on distinct biological families. Burrows bounds instead require the admissible count grid or a separately justified bound; finite-normal monotonicity is not silently transferred to that approximation.

The search shares a call-owned count-pair cache across all family sets. One cumulative work allowance includes tested count alternatives, dynamic-programming transitions and unsuccessful proposals. Cached intensities reduce repeated integration but do not remove executed search work from that allowance. A

stopped family-set allocation stays unresolved unless a valid bound excludes it; failure to find an incumbent does not prove infeasibility.

For small family sets, the implementation can enumerate precomputed local utilities with the last family's counts forced by both remaining budgets. Conservative work and workspace estimates precede that path. Other supported separable cases use the two-budget dynamic program. Coupled constraints use their original-input callback; an unconstrained relaxed solution remains a bound until those checks pass. A bounded larger path alternates production improvements at fixed retention, retention improvements at fixed production, and two-family count blocks preserving both totals. Family and direction exchanges use the same bounded inner solver. These searches preserve a checked supplied incumbent but do not certify the global family-choice problem through local improvement alone.

The variable-retention normal kernel also uses equivalent integrals in the normal variate for counts up to 256, away from its exact boundary cases. Here $b_{a,b}(p)$ is the beta density, $Q(z) = 1-\Phi(z)$ and $H(z) = \phi(z) - zQ(z)$. A change of variable $p = \Phi(z)$ in the preceding beta-expectation identity gives, for $1 \le r < n$,

$$a(n,r) = \frac{n}{r}\int_{-\infty}^{\infty} \phi(z)^2 b_{r,n-r}\{\Phi(z)\}\,dz.$$

The positive direct increment, for $1 \le r \le n$, is equivalently

$$a(n+1,r) - a(n,r) = \frac{1}{r}\int_{-\infty}^{\infty} H(z)\phi(z) b_{r,n-r+1}\{Q(z)\}\,dz.$$

The latter follows by substituting $q = Q(z)$ in the threshold stop-loss beta expectation. These changes of variable give equivalent representations of the normal order-statistic objective and direct increment defined above.

The C quadrature integrates the two intervals $[-12,0]$ and $[0,12]$ using log densities and the stable normal stop-loss calculation. For integer beta shapes, the density is at most one less than their sum, by its binomial probability representation. At a cutoff $T > 0$, the omitted tails are bounded respectively by

$$\frac{n(n-1)}{r}\frac{\phi(T)^2}{T}, \qquad \frac{n}{r}\left\{\phi(T) + \frac{\phi(T)^2}{T}\right\}.$$

For $T = 12$ and $n \le 256$, the implementation adds a conservative absolute allowance of 10^{-27}. This bounds truncation, while adaptive quadrature and arithmetic errors remain estimated. Larger counts or an unqualified density integration use the preceding beta-quantile calculation. The existing fixed-retention route keeps its original integration path.

Cache entries retain count pair, method, direct-increment type, refinement and numerical uncertainty. Successful cache hits avoid separate timing calls; their counts remain recorded, and total inner time still includes cache work. Small-family enumeration can reject a candidate whose coarse outward binary64 upper is strictly below a checked incumbent's lower. Near ties, cancellation and uncertain cases retain the full compensated component comparison, and final winning counts are rescored. This shortcut does not remove any charged count-search work or turn estimated normal errors into rigorous interval certificates.

A.11 Exact DH moments and selection-aware founder contributions

For marker j, $x_{pj} \in \{0,1\}$ identifies the effect allele in homozygous founder p, and u_j is the complete stored dosage coefficient. Thus the founder prediction and family contrast are

$$g_p = u_0 + 2\sum_j u_j x_{pj}, \qquad \mu_{ab} = \frac{g_a + g_b}{2}, \qquad d_j = u_j(x_{aj} - x_{bj}).$$

The factor two comes from the difference between homozygous dosages zero and two. The precise stored effects define these moments; rounded exported parental predictions do not replace them.

$S_j = 1$ denotes origin from parent a and $S_j = -1$ origin from b. For an immediate F1-derived DH, $E(S) = 0$. Its origin correlation matrix Ω has diagonal one, zero between chromosomes, and

$$\Omega_{jk} = \exp\{-2|p_j - p_k|/100\}$$

within a chromosome when map positions p are in cM. With fixed nonnegative marker weights w_j summing to one,

$$G = \mu_{ab} + d'S, \qquad V = d'\Omega d, \qquad C_a = \frac{1 + w'S}{2}, \qquad C_b = 1 - C_a,$$

$$\mathrm{Var}(C_a) = \frac{1}{4} w'\Omega w, \qquad h_{ab} = \mathrm{Cov}(C_a, G) = \frac{1}{2} w'\Omega d.$$

These unselected identities are exact under the declared map and fixed additive effects. The covariance contractions specialize Allier et al. (2019a), Equations (2) and (4a,b), and Allier et al. (2019b), Equations (4b) and (5). The DH origin correlation follows the preceding Osthushenrich et al. (2017) formulas. Origins remain labelled where the observed parental alleles coincide.

The implementation privately sorts markers within chromosomes. Exponential forward and backward recurrences calculate Ωw and the origin variance in linear time. Prepared founder sums $T_p = \sum_j (\Omega w)_j u_j x_{pj}$ give $h_{ab} = (T_a - T_b)/2$; a compensated direct contrast sum is used when that subtraction loses material precision. The trait variance uses a positive linear-time recurrence. Within a chromosome, with adjacent decay $q_j = \exp(-2\Delta_j)$ in Morgans,

$$F_1 = d_1, \qquad F_j = d_j + q_j F_{j-1}, \qquad V_{\mathrm{chr}} = F_m^2 + \sum_{j<m} (1 - q_{j+1}^2) F_j^2.$$

Innovation weights use `expm1` to preserve small distances. Chromosome resets and equal positions retain the same origin law. A fixed founder anchor and direct contrasts protect genetic deviations and covariances from a large common intercept. Family moments are cached independently of produced counts, retained counts and the selection method.

For positive variance, the declared ranking score is

$$I = \rho \frac{G - \mu}{\sqrt{V}} + \sqrt{1 - \rho^2}\, \varepsilon, \qquad \varepsilon \sim N(0,1),$$

with independent noise. A joint-normal regression approximation gives

$$c_{a,e}^* = \frac{1}{2} + \rho\, a(n_e, r_e) \frac{h_e}{\sqrt{V_e}}, \qquad c_{b,e}^* = 1 - c_{a,e}^*, \quad r_e > 0.$$

The contribution regression follows Allier et al. (2019a), Equations (5) and (6a–c), and Allier et al. (2019b), Equation (7). The latter is written for accuracy one. Its use with the declared independent-noise accuracy factor and the finite-family intensity $a(n,r)$ is the present adaptation. The normal intensity is the order-statistic mean cited from Okoye et al. (2026), Equations (4)–(8); the Burrows option keeps its preceding definition. The approximation concerns the conditional distribution of bounded origin variables, even when the normal integral is evaluated accurately.

Full retention, accuracy zero or constant genetic values give exactly one-half expected parental origin under neutral ranking. A zero retained count has no conditional selected summary and contributes zero population weight. Finite raw estimates outside $[0,1]$ remain visible. Their qualified contribution and proxy are unavailable with a stated approximation reason; values within that interval can still be biased.

A.12 Founder proxy, constraints and actual genomic metrics

With $R = \sum_e r_e > 0$, the analytic aggregate contribution is

$$c_p^* = \sum_{e: r_e > 0} \frac{r_e}{R} \{ \mathbf{1}(p = a_e) c_{a,e}^* + \mathbf{1}(p = b_e) c_{b,e}^* \}, \qquad Q_{\mathrm{proxy}} = (c^*)' K c^*.$$

This is the separately named `ucpc.founder.proxy` on the complete original supplied reference. Unused reference parents have zero contribution. Allier et al. (2019b), Equations (8)–(10), motivate the selected-population objective and diversity constraint; the current retained-count weighting and supplied-K convention are explicit here. An arbitrary supplied genomic matrix does not make $1 - Q_{\mathrm{proxy}}$ a gene-diversity value.

For a finite realized origin vector c_{real}, the quadratic expectation identity is

$$E(c'_{\text{real}} K c_{\text{real}}) = E(c_{\text{real}})' K E(c_{\text{real}}) + \text{tr}\{K \,\text{Var}(c_{\text{real}})\}.$$

The mean-contribution proxy omits the covariance term. It also averages origins over markers before using the supplied founder matrix. The discussion "Predicting the Next-Generation Diversity" in Allier et al. (2019b) identifies the consequences of replacing locus-specific ancestry by genome-average contributions.

Without `selected.founder.theta`, these predictions are report-only and leave the optimization problem unchanged. An active cap imposes $Q_{\text{proxy}} \leq \theta_{\text{selected}}$ in addition to the original family-slot and practical restrictions. It couples the actual produced and retained counts. Qualified analytic passes accept a candidate; an unavailable approximation or unresolved comparison cannot prove infeasibility. The same actual-count callback checks fixed-retention and variable-retention solves. Numerical intensity errors propagate through the contribution and quadratic calculations, with estimated arithmetic uncertainty. They do not bound the regression approximation bias. Fractional contributions do not become literal PropSD multiplicities.

For the scenario allele metric, source reference dosages give fixed frequencies p_{0j} and $D = \sum_j w_j p_{0j}(1 - p_{0j})$. With $X = Z - 2p_0$, the weighted construction is

$$A_w = \frac{X \,\text{diag}(w) X'}{2D}, \qquad K_{\text{genomic}} = A_w/2.$$

This is the weighted extension of the first method in VanRaden (2008, p. 4416; unnumbered formula). For one retained DH panel with allele frequencies p_{sj}, the all-pairs mean including diagonals is

$$Q_s = \frac{\sum_j w_j (p_{sj} - p_{0j})^2}{D}, \qquad H_s = 2\sum_j w_j p_{sj}(1 - p_{sj}).$$

The metric name is `weighted.vanraden.A.over.2.fixed.reference`. It uses the fixed source reference and sampling with replacement over retained pairs; it is not re-centered on the selected lines. A zero reference denominator leaves Q_s unavailable. Population gene diversity H_s remains distinct from individual DH heterozygosity, which is zero. Nonlinear Q_s and H_s are calculated inside each replicate and then averaged, preserving their finite-panel estimands.

A.13 Actual-genotype scenarios, variance and inheritance

Each chromosome starts at either founder with equal probability and switches parental origin over adjacent distance Δ Morgans with probability $(1 - \exp(-2\Delta))/2$. The scenario algorithm streams candidate genetic deviations and rank priorities into a top-r heap, then regenerates the selected origin paths. It uses theoretical family variance in the ranking law, with neutral independent priorities for ties. Actual origin labels remain available at identical alleles. Genotypes are generated only for the requested final assessment.

The private bank records its protocol, core model, seed and replicate. Primitive candidate keys use the unordered biological family, candidate, chromosome and separate origin, noise and tie domains. Produced and retained counts, accuracy and display order are excluded from those primitive keys. Larger produced counts reuse candidate prefixes, and changed retention or accuracy reuses the same genetic paths and priorities. Streaming regeneration avoids a full produced-genotype tensor. Caller RNGs, population identities and marker-data state remain independent of this private assessment.

For replicate s with R selected genetic values, $\overline{G}_s$ is their mean and

$$V_s = \frac{1}{R}\sum_{i=1}^{R} (G_{si} - \overline{G}_s)^2.$$

The expected finite-panel variance is $E(V_s)$. Expected empirical sample variance uses $RV_s/(R-1)$ for $R > 1$. The variance of a randomly chosen selected individual is

$$V_{\text{marginal}} = E(V_s) + \text{Var}(\overline{G}_s).$$

Its estimate combines the mean panel variance with the unbiased sample variance of replicate panel means, requiring multiple replicates. At $R = 1$, panel variance is zero, empirical sample variance is undefined, and marginal variance may remain positive. The same distinctions apply within each family. Centered accumulation preserves small genetic variation in the presence of a large shared intercept.

Ordinary metric SE uses the sample standard deviation of independent whole-replicate values divided by the square root of the completed replicate count. One replicate has no such SE. The marginal-variance SE is explicitly unavailable in the present implementation. A work-limited bank reports completed whole replicates and its partial status; no partial current replicate contributes to the published estimates. Analytic proxy feasibility and scenario completion retain separate states.

The normalized model stores an immutable versioned snapshot of candidate alleles, effects, map, weights and fixed reference information. Canonical linkage-group keys use the smallest marker identifier in each group. Evaluation and allocation inherit that core and its original cap; live dataset changes do not rebind it. Current actual counts and the current selection method determine a new analytic assessment. Scenario execution requires an explicit current request, so inherited historical scenario results do not become predictions for changed counts. This model describes the selected G1 DH panel; a further crossing and DH cycle requires the separate future-population policy.

A.14 Actual selected DH lines and a second crossing cycle

The current design p contains the active founder pairs and their produced and retained counts (n_{1e}, r_{1e}). Its fixed totals are C_1, P_1 and $R_1 = \sum_e r_{1e}$. Actual F1-derived DH candidates follow the preceding additive Haldane model and the declared independent-noise index. The selected logical lines retain their original family and candidate identities, including distinct lines with identical observed genotypes.

For even $R_1 \geq 2$, a uniform permutation of all selected lines is split into adjacent pairs. This gives a uniform disjoint matching and uses every line once. The next cycle has $C_2 = R_1/2$ pairs, with uniform produced count n_2, retained count r_2 and accuracy ρ_2; hence $P_2 = C_2 n_2$ and $R_2 = C_2 r_2$. The policy permits two distinct selected lines from the same original family. It does not choose the future mates adaptively or reuse a selected parent.

For actual homozygous G1 allele vectors x and y, the new F1 has

$$\mu_{xy} = \frac{G(x) + G(y)}{2}, \qquad d_j = u_j(x_j - y_j), \qquad V_{xy} = d'\Omega d.$$

Here u_j, G and Ω have the preceding dosage and Haldane definitions. A new G2 DH has genetic value $\mu_{xy} + d'S$ from an actual origin path S. The already homozygous DH parents themselves have zero segregation; variation arises after their new cross. Conditional variance uses the positive linear-time recurrence stated above, not a kinship or mean-ancestry replacement. The reported pair diagnostics $E(V_{xy})$ and $E(\sqrt{V_{xy}})$ are distinct. In general, $E(\sqrt{V_{xy}}) \neq \sqrt{E(V_{xy})}$, and neither diagnostic replaces the actual two-cycle calculation.

The target is the expected selected G2 mean under this declared policy,

$$J_2(p) = E_p\left(\overline{G}_{2,\text{selected}}\right).$$

Each realized matching has average unselected G2 family mean equal to its selected G1 mean: every G1 line appears once and every G2 pair has the same retained weight. With

$$B_1(p) = \frac{\sum_e r_{1e}\mu_e}{R_1}, \qquad \Delta_{1s} = \overline{G}_{1s} - B_1(p), \qquad \Delta_{2s} = \overline{G}_{2s} - \overline{G}_{1s},$$

the component identity is

$$J_2(p) = B_1(p) + E_p(\Delta_1) + E_p(\Delta_2).$$

The two responses in this identity arise from actual sampled genotypes. The earlier normal-model G1 usefulness and its analytic response retain their separate definitions. Neutral G2 selection or full retention gives $E_p(\Delta_2) = 0$; a finite realized bank can have a nonzero response.

Niehoff et al. (2024), Equation (2), decomposes gametic variance among selected offspring. Their Equation (8) is an approximate two-generation selection criterion; it is not the actual matched-DH calculation above. Equation (9) and Appendix 1 give mean offspring gametic sampling variance as $(\sigma^2_{DH} - \sigma^2_{FS})/4$.

The factor four converts their diploid-DH variance to a gametic scale. The directly computed V_{xy} above already describes diploid DH values and is not divided by four. The actual mating and two DH generations define the present adaptation. Future-oriented progeny allocation is also the topic of Hamazaki and Iwata (2024), Sections 1 and 2.1.

A.15 Finite-bank search and paired future assessment

A fixed training bank with S_T complete replicates defines

$$\widehat{J}_{2,T}(p) = \frac{1}{S_T} \sum_{s=1}^{S_T} \overline{G}_{2s}(p).$$

Every competing candidate uses this same complete bank. An interrupted partial training bank supplies no comparable candidate score. Existing G1 solvers supply checked seeds and feasibility repairs; their separable G1 bounds are not bounds for J_2. The joint search preserves current production and retention totals through count transfers, joint count blocks and family exchanges, subject to the actual current constraints. A best-found status describes the finite training search. Exhaustive small-domain completion can establish an optimum of that bank, while the gap to the unknown expectation optimum remains unavailable without a separate valid bound.

Held-out evaluation uses a separate role domain. For two actual designs p and q, the common evaluation bank gives paired differences

$$D_s = \overline{G}_{2s}(p) - \overline{G}_{2s}(q), \qquad \widehat{D} = \frac{1}{S_E} \sum_{s=1}^{S_E} D_s,$$

$$\mathrm{MCSE}(\widehat{D}) = \sqrt{\frac{\sum_s (D_s - \widehat{D})^2}{S_E (S_E - 1)}}, \qquad S_E > 1.$$

The same construction applies to G1 means and the additional G2 response. Adding unrelated design-level SEs does not preserve this pairing. Held-out observations do not select a training incumbent, stopping rule or later frontier warm start.

The private two-cycle bank separates core, role, stage, seed and replicate. G1 primitive keys use intrinsic original-family and candidate identities. Selected identities are sorted before the descending Fisher–Yates permutation of Durstenfeld (1964), Algorithm 235. Unbiased integer rejection supplies each permutation choice from the private bank. Realized unordered pairs are oriented by logical identity and sorted into matching slots. G2 path, index-noise and tie primitives use slot and candidate keys, with independent slots within a replicate. The same actual pair can occupy another slot in another design, so a cached pair response also depends on slot and path identity. This common bank preserves each design's marginal law without claiming identical trajectories for every pair across designs. It is separate from the preceding single-cycle scenario protocol.

Signed trait sums use compensated accumulation following Neumaier (1974). For an accumulated main term s, residual c and new summand a, the update has the form

$$t = s + a, \qquad s' = t, \qquad c' = c + \begin{cases} (s - t) + a, & |s| \geq |a|, \\ (a - t) + s, & |s| < |a|, \end{cases}$$

with the final sum represented by $s + c$. The implementation applies this protection to signed anchor, family-contrast and future-trait boundaries. Centered components preserve small responses beside a large common mean; scale-aware variance accumulation protects the corresponding second moments. These safeguards do not provide arbitrary precision. An unrepresentable required quantity has an explicit numerical-range status.

Bounded reusable caches retain intrinsic candidate paths, selected mosaics, pair moments and slot-specific results. One cumulative cell and work budget covers all search candidates, frontier points and final assessment. Reserved final-assessment work remains inside that budget. Incomplete final banks report only

jointly completed whole replicates and their completion status. Resource limits do not establish biological infeasibility, and a partial result does not become a full-bank certificate.

Actual G1 and G2 genomic metrics use their realized finite panels, the unchanged fixed reference and the preceding Q_s, H_s and variance definitions. Nonlinear metrics are averaged after calculation inside each replicate. Monte Carlo SE describes simulation precision under fixed fitted effects. The accuracy and shrinkage qualifications in Niehoff et al. (2025), Sections 4.1 and 4.4, remain relevant; this assessment adds no phenotypic information or effect-estimation uncertainty.

A.16 Labelled gain–diversity frontier

Each declared frontier limit replaces one named current G1 constraint: family-slot θ or the qualified selected-founder UCPC proxy cap. Other constraints and total resources remain fixed. Default independent starts retain the recorded standalone start and seed at every point; optional warm starts carry actual previous counts and pairs with an accepted or fallback status. Core data and private banks can be shared, while the work budget covers the whole frontier call.

Every point retains its actual search, feasibility and assessment status. A complete declared schedule agrees with its corresponding standalone problem under the same recorded inputs and sufficient matching resources. A capped schedule remains explicitly incomplete. The horizontal metric describes a G1 constraint, whereas actual G2 genomic Q and H are additional estimates. The proxy cap is not a hard bound on actual G2 genomic coancestry. Sample-based or bounded-search points need not be monotone and do not establish a smooth global tradeoff curve.

B Cross selection from external diploid inputs

We now select five crosses from externally supplied marker data and marker effects. The examples use the first ten fully homozygous diploid parents and first 100 markers for additive yield, a two-trait additive index, and the same index with dominance. The input file `crossing-example-inputs.rds` is prepared separately and located in the working directory; no input data are included.

The RDS object is a list with `markers`, `additive`, `dominance`, and `weights`. The numeric marker matrix has parents in rows, markers in columns, unique row and column names, and at least ten rows and 100 columns. Dosages are 0 or 2 with no missing values. The additive and dominance components are named lists with `YLD` and `MAT` vectors named by marker. The numeric weights are named `YLD` and `MAT`. Effects and weights are finite and use compatible trait units and allele orientation; larger index values are preferred.

B.1 Family means and variances

All 45 unordered nonself crosses are evaluated with an independent-marker approximation that omits linkage covariance. For dosage z, a marker contributes $az + d\,1_{\{z=1\}}$, with dominance expressed as the heterozygote deviation from the additive midpoint and no fitted intercept. Three selfing cycles after F_1 give F_4 progeny and heterozygosity $h = 1/8$ at loci where the parents differ. At these loci, the mean dominance contribution is hd and the variance is $(1-h)a^2 + h(1-h)d^2$. Equal parental loci have no segregation variance. Trait effects are weighted before the index moments are calculated, retaining covariance induced by effects at the same marker.

The expected superior progeny score is $\mu + i\sigma$, where $i = \phi(\Phi^{-1}(1-5/10))/(5/10)$. This normal selection approximation represents retaining five of ten progeny, without a finite-population intensity correction.

```
> input <- readRDS("crossing-example-inputs.rds")
> cross.example.prepare <- function(input) {
+     X <- input$markers
+     stopifnot(is.matrix(X), is.numeric(X), nrow(X) >= 10,
+               ncol(X) >= 100)
+     X <- X[seq_len(10), seq_len(100), drop = FALSE]
+     stopifnot(!anyNA(X), all(X %in% c(0, 2)),
```

```
+               !is.null(rownames(X)), !is.null(colnames(X)),
+               !anyDuplicated(rownames(X)), !anyDuplicated(colnames(X)))
+     loci <- colnames(X)
+     effects <- function(x) {
+         stopifnot(is.numeric(x), !is.null(names(x)),
+                   !anyDuplicated(names(x)), all(loci %in% names(x)))
+         out <- unname(x[loci])
+         stopifnot(all(is.finite(out)))
+         out
+     }
+     a.yield <- effects(input$additive$YLD)
+     stopifnot(is.numeric(input$weights),
+               !anyDuplicated(names(input$weights)),
+               all(c("YLD", "MAT") %in% names(input$weights)))
+     weights <- input$weights[c("YLD", "MAT")]
+     stopifnot(all(is.finite(weights)))
+     a.index <- weights[1] * a.yield +
+                weights[2] * effects(input$additive$MAT)
+     d.index <- weights[1] * effects(input$dominance$YLD) +
+                weights[2] * effects(input$dominance$MAT)
+     centered <- sweep(X, 2, colMeans(X), "-")
+     p <- colMeans(X) / 2
+     scale <- 2 * sum(p * (1 - p))
+     stopifnot(is.finite(scale), scale > 0)
+     G <- tcrossprod(centered) / scale
+     dimnames(G) <- list(rownames(X), rownames(X))
+     pairs <- t(combn(rownames(X), 2))
+     h <- 2^(-3)
+     intensity <- dnorm(qnorm(1 - 5 / 10)) / (5 / 10)
+     score <- function(a, d = rep(0, ncol(X))) {
+         mu <- va <- numeric(nrow(pairs))
+         for (j in seq_len(nrow(pairs))) {
+             x1 <- X[pairs[j, 1], ]
+             x2 <- X[pairs[j, 2], ]
+             segregating <- x1 != x2
+             mu[j] <- sum((x1 + x2) * a / 2) +
+                      h * sum(d[segregating])
+             va[j] <- sum(((1 - h) * a^2 +
+                           h * (1 - h) * d^2)[segregating])
+         }
+         data.frame(P1Name = pairs[, 1], P2Name = pairs[, 2],
+                    mu = mu, va = va,
+                    es = mu + sqrt(va) * intensity)
+     }
+     list(markers = X, G = G, K = G / 2,
+          effects = list(yield = a.yield, index = a.index,
+                         dominance = a.index),
+          scores = list(yield = score(a.yield), index = score(a.index),
+                        dominance = score(a.index, d.index)),
+          accuracy = 0.5, loss.limit = 0.05,
+          n.crosses = 5L, n.progeny = 10L, n.selected = 5L)
+ }
> example <- cross.example.prepare(input)
```

B.2 Three crossing plans

Each plan contains five distinct nonself crosses, ten progeny per family and five retained per family: 50 produced and 25 retained. The planner selects the families with these fixed counts. Its accuracy of 0.5 adjusts the supplied within-family selection differential once, giving $\mu + 0.5i\sigma$.

Allele frequencies from the ten-parent panel define the marker relationship matrix and `K=G/2`. The relatedness bound and maximum contribution of 0.5 are nonrestrictive here. The diversity constraint requires PropSD strictly below 0.05. Its sample comprises the ten parental occurrences in the selected families, in-

cluding repeated occurrences of a parent used in several families. The full ten-parent matrix remains the reference. This covariance-based expected SD loss differs from realized variation among selected descendants. Appendix A gives its definition.

The first plan uses additive yield effects.

```
> cross.example.plan <- function(example, scenario) {
+     merit <- setNames(drop(example$markers %*% example$effects[[scenario]]),
+                       rownames(example$markers))
+     SelectionTools::st.cross.plan(
+         merit = merit, K = example$K,
+         n.crosses = example$n.crosses, n.progeny = example$n.progeny,
+         theta = max(diag(example$K)), upper = 0.5,
+         scores = example$scores[[scenario]],
+         criterion = "usefulness", accuracy = example$accuracy,
+         diversity = list(G = example$G, max = example$loss.limit,
+                          sample = "family.slots"),
+         control = list(method = "exact", max.nodes = 3000000,
+                        starts = 4, maxit = 2500, seed = 1L)
+     )
+ }
> yield.plan <- cross.example.plan(example, "yield")
> yield.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: optimal
Families: 5; progeny per family: 10; total progeny: 50
Expected mean: 4.5042
Expected usefulness: 7.2786; accuracy: 0.5000
Group relatedness: 0.1163; requested limit: 1.2632
PropSD (family slots): 0.0447; strict upper limit: 0.0500
Average within-cross relatedness: -0.2379
Objective bound: 7.2786; optimality gap: <0.0001
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  g1_2      g1_3           10  4.6854  -0.2789      7.4090
       2  g1_2      g1_9           10  6.5700  -0.0684      9.3653
       3  g1_3      g1_9           10  6.9320  -0.2000      9.5395
       4  g1_4      g1_8           10  0.2173  -0.1737      2.8720
       5  g1_8      g1_9           10  4.1162  -0.4684      7.2073
```

The second plan uses the weighted additive yield and maturity index.

```
> index.plan <- cross.example.plan(example, "index")
> index.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: optimal
Families: 5; progeny per family: 10; total progeny: 50
Expected mean: 3.7261
Expected usefulness: 6.2713; accuracy: 0.5000
Group relatedness: 0.1295; requested limit: 1.2632
PropSD (family slots): 0.0498; strict upper limit: 0.0500
Average within-cross relatedness: -0.2000
Objective bound: 6.2713; optimality gap: <0.0001
Maximum constraint residual: 0.0000

Chosen crosses
```

```
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  g1_2      g1_4           10  3.3504  -0.0895      5.8141
       2  g1_2      g1_9           10  4.7400  -0.0684      7.2237
       3  g1_3      g1_9           10  4.5557  -0.2000      7.0925
       4  g1_4      g1_8           10  2.2974  -0.1737      4.7305
       5  g1_8      g1_9           10  3.6870  -0.4684      6.4955
```

The third plan adds the weighted dominance effects to this index.

```
> dominance.plan <- cross.example.plan(example, "dominance")
> dominance.plan
```

```
Optimum cross selection (usefulness)
Criterion: maximum predicted usefulness
Status: optimal
Families: 5; progeny per family: 10; total progeny: 50
Expected mean: 4.1876
Expected usefulness: 6.8450; accuracy: 0.5000
Group relatedness: 0.1295; requested limit: 1.2632
PropSD (family slots): 0.0498; strict upper limit: 0.0500
Average within-cross relatedness: -0.2000
Objective bound: 6.8450; optimality gap: <0.0001
Maximum constraint residual: 0.0000

Chosen crosses
  Family  Parent 1  Parent 2  Progeny    Mean  Kinship  Usefulness
       1  g1_2      g1_4           10  3.8697  -0.0895      6.4536
       2  g1_2      g1_9           10  5.8993  -0.0684      8.4953
       3  g1_3      g1_9           10  4.6654  -0.2000      7.2996
       4  g1_4      g1_8           10  2.3941  -0.1737      4.9261
       5  g1_8      g1_9           10  4.1095  -0.4684      7.0503
```

The displayed results are conditional on the external inputs used in this example. An `optimal` status certifies the optimum for the supplied scores and constraints. Another panel can require more search or yield no feasible plan. Yield and index objectives retain their respective effect scales. The normal selection approximation and supplied marker effects remain assumptions even at an optimum.

References

Allier A, Moreau L, Charcosset A, Teyssèdre S, Lehermeier C (2019a) Usefulness Criterion and Post-selection Parental Contributions in Multi-parental Crosses: Application to Polygenic Trait Introgression. G3: Genes, Genomes, Genetics 9:1469–1479. doi:10.1534/g3.119.400129

Allier A, Lehermeier C, Charcosset A, Moreau L, Teyssèdre S (2019b) Improving Short- and Long-Term Genetic Gain by Accounting for Within-Family Variance in Optimal Cross-Selection. Frontiers in Genetics 10:1006. doi:10.3389/fgene.2019.01006

Bellman R (1954) The theory of dynamic programming. Bulletin of the American Mathematical Society 60:503–515.

Boyd S, Vandenberghe L (2004) Convex Optimization. Cambridge University Press, Cambridge.

Burrows PM (1972) Expected selection differentials for directional selection. Biometrics 28:1091–1100

Crossa J, de los Campos G, Pérez P, Gianola D, Burgueño J, et al. (2010) Prediction of genetic values of quantitative traits in plant breeding using pedigree and molecular markers. Genetics 186:713–724

Durstenfeld R (1964) Algorithm 235: Random permutation. Communications of the ACM 7:420. doi:10.1145/364520.364540

Endelman JB (2025) Genomic prediction of heterosis, inbreeding control, and mate allocation in outbred diploid and tetraploid populations. Genetics 229:iyae193.

Fernández-González J, Metwally SM, Isidro y Sánchez J (2026) MateR: a novel genomic mating framework. Genetics 232:iyag013. doi:10.1093/genetics/iyag013

Gorjanc G, Hickey JM (2018) AlphaMate: a program for optimizing selection, maintenance of diversity and mate allocation in breeding programs. Bioinformatics 34:3408–3411. doi:10.1093/bioinformatics/bty375

Haldane JBS (1919) The combination of linkage values, and the calculation of distances between the loci of linked factors. Journal of Genetics 8:299–309.

Hamazaki K, Iwata H (2024) AI-assisted selection of mating pairs through simulation-based optimized progeny allocation strategies in plant breeding. Frontiers in Plant Science 15:1361894. doi:10.3389/fpls.2024.1361894

Kosambi DD (1943) The estimation of map distances from recombination values. Annals of Eugenics 12:172–175.

Land AH, Doig AG (1960) An automatic method of solving discrete programming problems. Econometrica 28:497–520.

Magnus JR (1978) The moments of products of quadratic forms in normal variables. Statistica Neerlandica 32:201–210.

Mehrotra S (1992) On the implementation of a primal-dual interior point method. SIAM Journal on Optimization 2:575–601.

Meuwissen THE (1997) Maximizing the response of selection with a predefined rate of inbreeding. Journal of Animal Science 75:934–940.

Neumaier A (1974) Rundungsfehleranalyse einiger Verfahren zur Summation endlicher Summen. ZAMM 54:39–51. doi:10.1002/zamm.19740540106

Niehoff TAM, ten Napel J, Bijma P, Pook T, Wientjes YCJ, Hegedűs B, Calus MPL (2024) Improving selection decisions with mating information by accounting for Mendelian sampling variances looking two generations ahead. Genetics Selection Evolution 56:41. doi:10.1186/s12711-024-00899-2

Niehoff TAM, ten Napel J, Calus MPL (2025) Practical Considerations When Using Mendelian Sampling Variances for Selection Decisions in Genomic Selection Programs. Journal of Animal Breeding and Genetics 142:419–437. doi:10.1111/jbg.12913

NIST Digital Library of Mathematical Functions. Chapter 5, Gamma function, Equations 5.2.1 and 5.11.8. National Institute of Standards and Technology. https://dlmf.nist.gov/5 (accessed 12 November 2026).

Okoye UJ, Frisch M, Herzog E (2026) Optimization of family sizes in sets of crosses with a greedy allocation strategy based on automatic differentiation. Frontiers in Plant Science 17:1727383. doi:10.3389/fpls.2026.1727383

Osthushenrich T, Frisch M, Herzog E (2017) Genomic selection of crossing partners on basis of the expected mean and variance of their derived lines. Plos One 12(12):e0188839

Reif JC, Melchinger AE, Frisch M (2005) Genetical and Mathematical Properties of Similarity and Dissimilarity Coefficients Applied in Plant Breeding and Seed Bank Management. Crop Science 45:1-7

Schnell FW, Utz HF (1975) F_1-Leistung und Elternwahl in der Züchtung von Selbstbefruchtern. Bericht über die Arbeitstagung der Vereinigung österreichischer Pflanzenzüchter, Gumpenstein, pp. 243–248.

VanRaden PM (2008) Efficient methods to compute genomic predictions. Journal of Dairy Science 91:4414–4423.

Wellmann R (2019) Optimum contribution selection for animal breeding and conservation: the R package optiSel. BMC Bioinformatics 20:25

Zhong S, Jannink JL (2007) Using quantitative trait loci results to discriminate among crosses on the basis of their progeny mean and variance. Genetics 177:567–576