Planing crosses

Matthias Frisch

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(-2) # Only error messages
> data("v-tropmaize-vcf")
> data("v-tropmaize-phe")
> st.load.vcf.data(v.tropmaize.vcf)
> st.load.performance.data(v.tropmaize.phe)
```

preprocess the data

```
> 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  )
```

and load the phenotypic data.

```
> st.load.performance.data(v.tropmaize.phe)
```

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" )
```

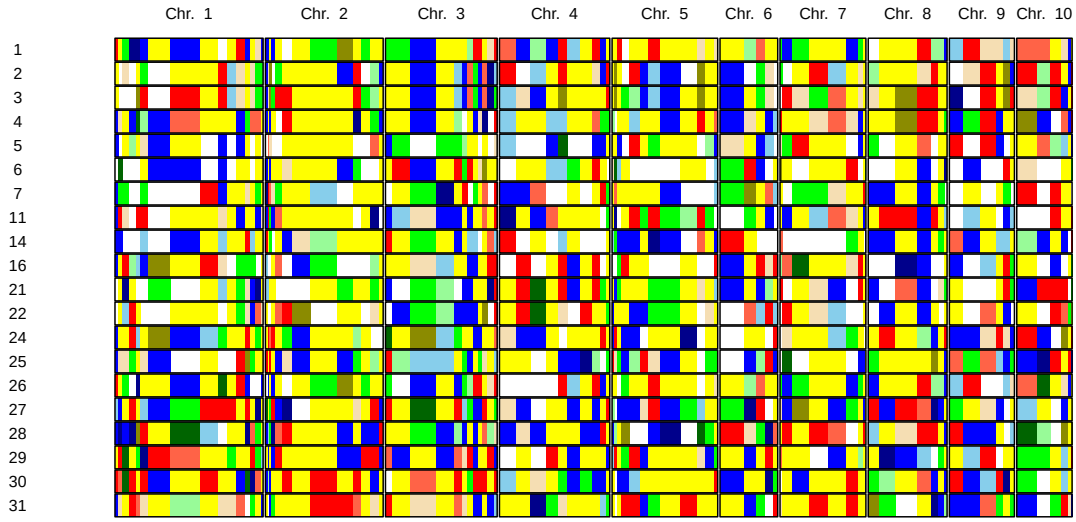
In the second step the variants of the haplotype blocks are determined. For inbred lines, the variants are determined with `st.recode.hil()`.

```
> 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[1:10,]
```

```
   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

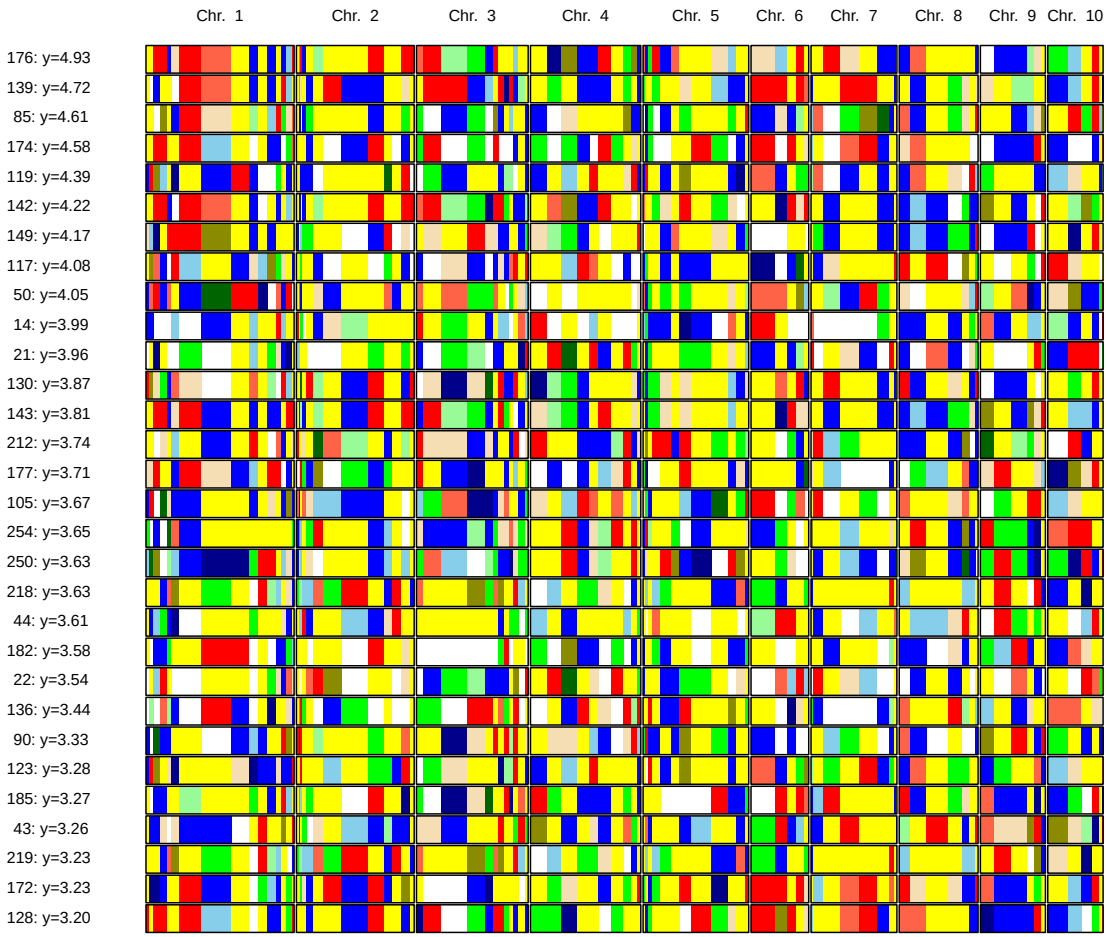
```
> st.copy.marker.data( "hblocks02", "preprocessed" )
> h2 <- st.def.hblocks( hap      = 5,
+                       hap.unit = 2,
+                       data.set = "hblocks02" )
> h2[1:10,]
```

```
   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

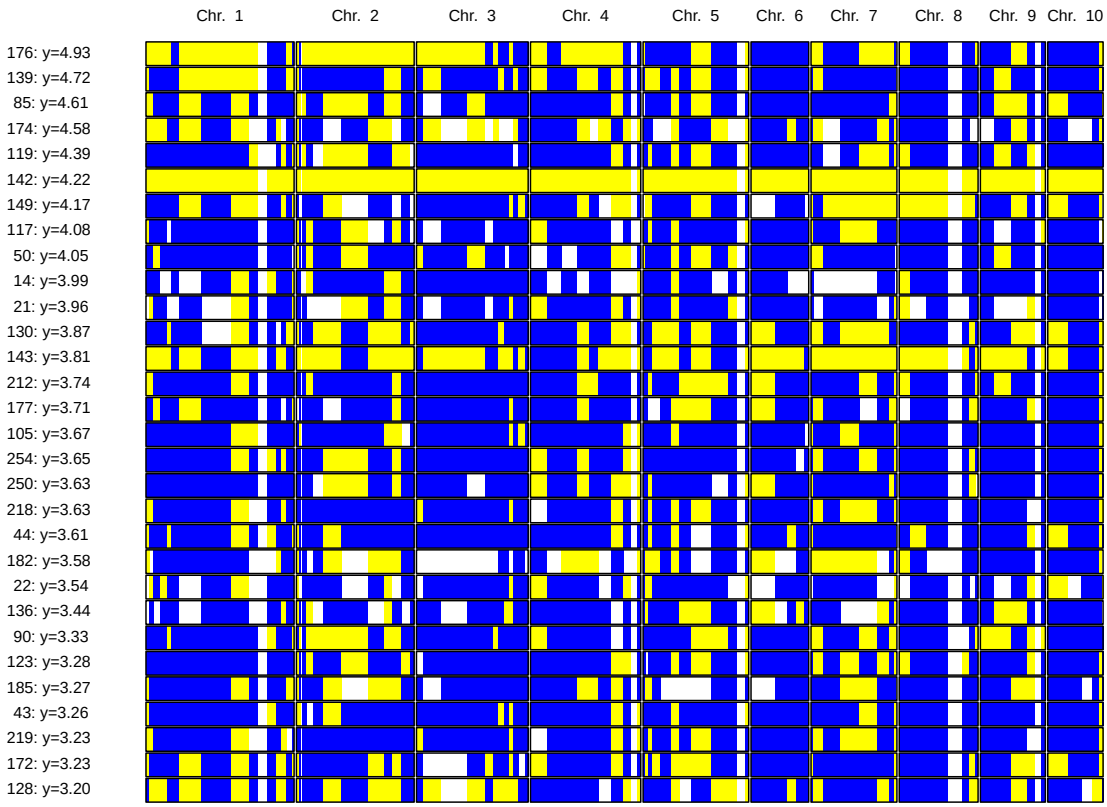
```
> st.copy.marker.data( "hblocks03", "hblocks01" )
```

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

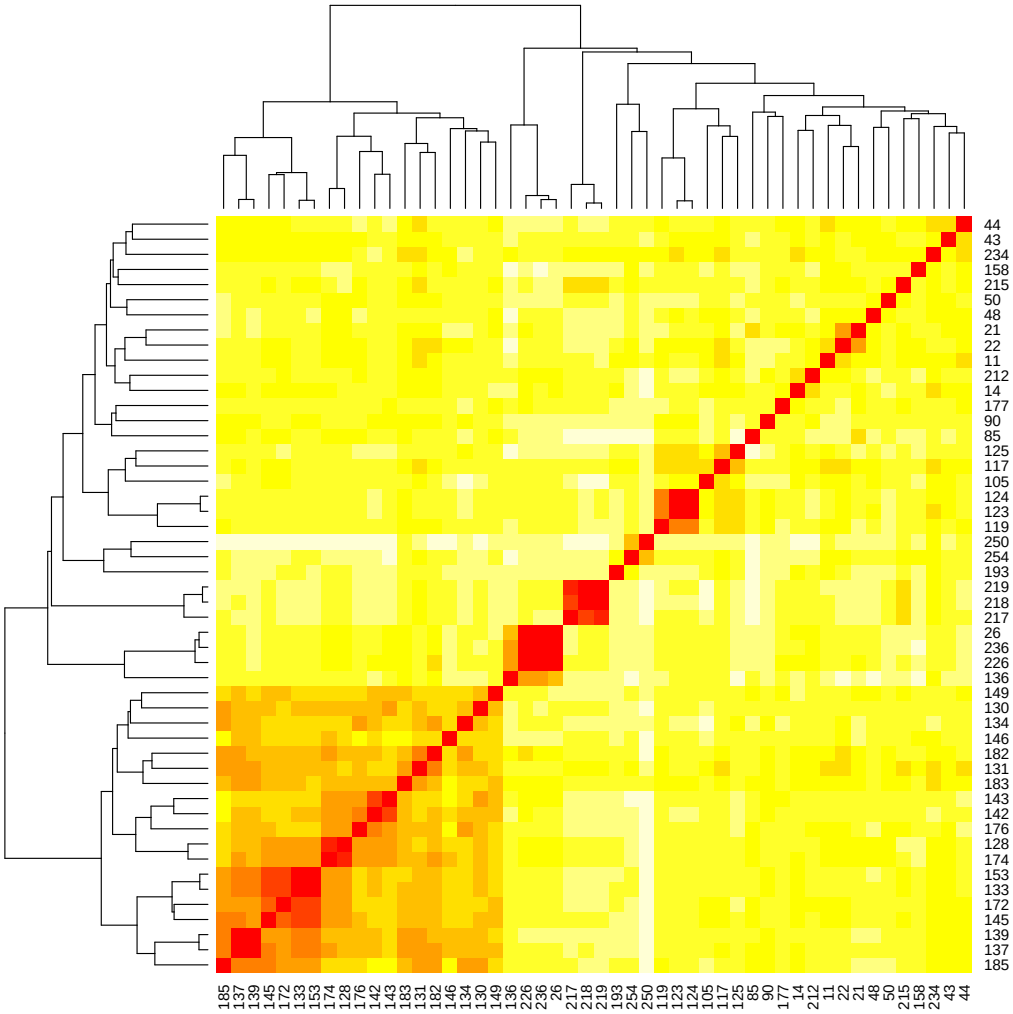
```
> best <- st.select.phen( n=50, data.set="gd01" )
```

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=rev(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" )
> crosses[1:10,]
```

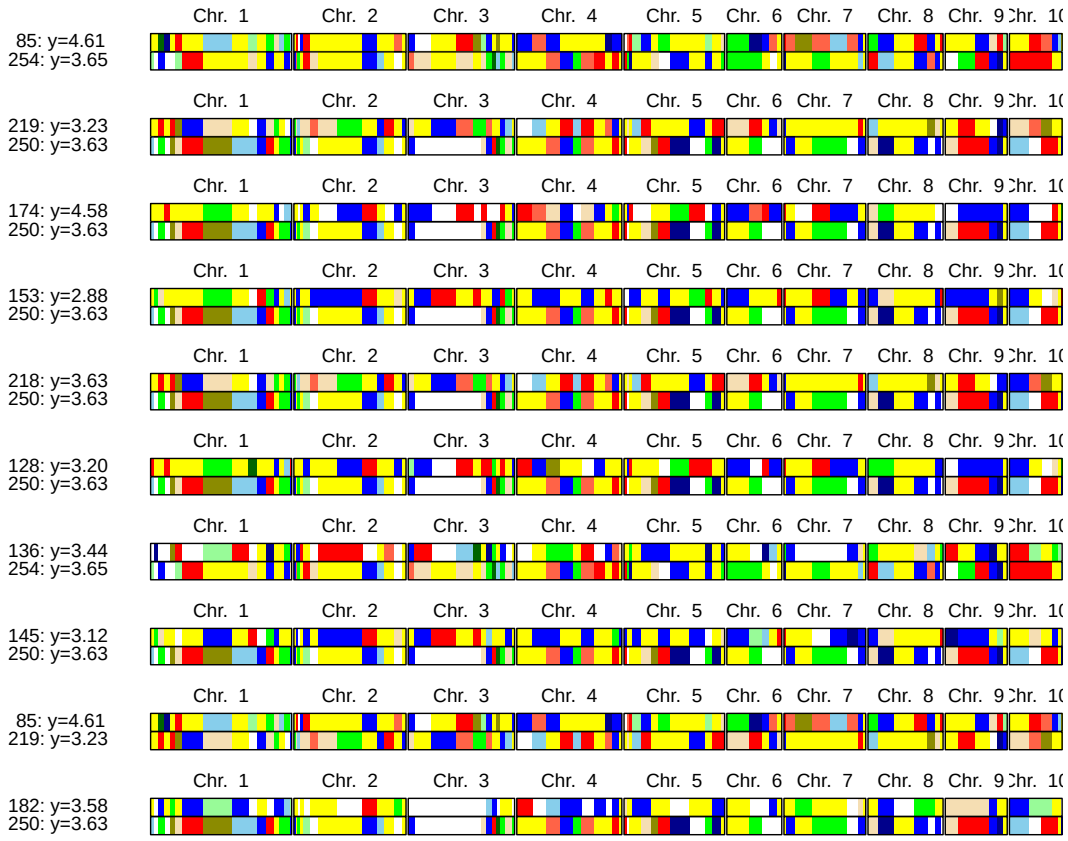
```
> crosses[1:10,]
   P1No P2No P1Name P2Name       gd mu mi ma va es
1    10   50     85    254 0.611269  0  0  0  0  0
2    45   49    219    250 0.607450  0  0  0  0  0
3    34   49    174    250 0.606780  0  0  0  0  0
4    31   49    153    250 0.606128  0  0  0  0  0
5    44   49    218    250 0.606128  0  0  0  0  0
6    18   49    128    250 0.604990  0  0  0  0  0
7    23   50    136    254 0.603865  0  0  0  0  0
8    28   49    145    250 0.603353  0  0  0  0  0
9    10   45     85    219 0.602531  0  0  0  0  0
10   37   49    182    250 0.602506  0  0  0  0  0
```

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,
+                 data.set="gd01")
+     dev.off()
+ }
> st.set.info.level(0)
```



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` give lower and upper marker-effect based limits 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 supplied and 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).

```
> 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. 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")
> format(crosses,digits=2,nsmall=2)
```

```
  P1No P2No P1Name P2Name    gd   mu    mi   ma    va   es
1  121  150    142    176 0.180 3.65 -1.16 6.59 0.023 3.80
2  122  150    143    176 0.199 3.60 -1.30 6.73 0.029 3.78
3  121  122    142    143 0.097 3.58 -0.86 6.15 0.018 3.72
4  109  150    130    176 0.211 3.52 -1.47 6.84 0.021 3.67
```

```
5   109  121    130    142 0.236 3.50 -1.40 6.68 0.021 3.65
6   121  148    142    174 0.207 3.49 -1.25 6.56 0.024 3.65
7   148  150    174    176 0.211 3.51 -1.33 6.84 0.017 3.65
8   116  121    137    142 0.263 3.48 -1.43 6.76 0.027 3.65
9   118  121    139    142 0.267 3.47 -1.48 6.78 0.027 3.64
10  116  150    137    176 0.250 3.50 -1.58 6.91 0.017 3.64
11   68  150     85    176 0.319 3.39 -1.96 7.04 0.054 3.63
12  118  150    139    176 0.250 3.49 -1.60 6.93 0.017 3.63
13  122  148    143    174 0.203 3.44 -1.31 6.56 0.029 3.62
14  116  122    137    143 0.283 3.43 -1.50 6.87 0.032 3.62
15  109  122    130    143 0.219 3.46 -1.43 6.69 0.024 3.62
16  107  150    128    176 0.220 3.47 -1.47 6.77 0.020 3.61
17  107  121    128    142 0.207 3.44 -1.33 6.46 0.026 3.61
18  118  122    139    143 0.288 3.42 -1.56 6.89 0.032 3.61
19   98  150    119    176 0.360 3.36 -2.02 7.20 0.054 3.60
20   68  121     85    142 0.318 3.37 -1.78 6.82 0.049 3.60
```

References

Burrows RM (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

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

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