Simulated SNP Set and Corresponding Signal Sets

Contents

In this document, we provide a detailed look at the different signal sets used in our simulations; we examine the specific indices of $\boldsymbol{X}$ that comprise each signal set, the breakdown of which signals contribute to the effect of $\boldsymbol{X}$ on each component of $\boldsymbol{Y}$, and the correlation structure of each signal set. We also generate a correlation heatmap for the simulated SNP set used in our simulation study.

```r
# load packages
library(Matrix)
library(sim1000G)
library(reshape2)
library(ggplot2)
library(viridis)

# Define correlation heatmap function
heatmap <- function(X, tag){
  ggplot(data = melt(round(cor(X), 2)),
         aes(x = Var1, y = Var2, fill = value)) +
    geom_tile() +
    scale_fill_gradient2(low = "darkblue",
                         high = "red",
                         mid = "white", midpoint = 0, limit = c(-1,1),
                         space = "Lab", name = "Pearson Cor.") +
    coord_fixed() +
    theme_classic() +
    labs(x = "SNP number", y = "SNP number",
         title = "Correlation Heatmap",
         subtitle = paste0(tag, " (", ncol(X), " SNPs)"))
}
```

Full SNP set

We begin by initializing a scenario where $\boldsymbol{X}$ represents SNP-set data, using the parameters seen below.

```r
# Scenario parameters
size_or_power <- "power"
alpha <- 0.05
num_permutations <- 1000
num_replicates <- 1000
x_type <- "snp"
```

```
error_distribution <- "normal"
signal_strength <- 1
signal_density <- "sparse"
signal_correlation <- "low"
error_correlation_strength <- 0.5
n <- 1000
p <- 567

# Root and source script directories
dir_main <- dirname(dirname(rstudioapi::getActiveDocumentContext()$path))
dir_src <- file.path(dir_main, 'source_scripts')

# Initialize scenario simulation environment
source(file.path(dir_src, 'initialize_simulation_scenario.R'))
```

```
## [#.......] Reading VCF file..
```

```
## Rows: 569 Columns: 104
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (101): ID, REF, ALT, FILTER, INFO, FORMAT, NA06984, NA06989, NA12347, NA...
## dbl   (3): #CHROM, POS, QUAL
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## [##......] Chromosome:   4  Mbp:  77.35628  Region Size:  347.154 kb  Num of individuals: 95
## [##......] Before filtering  Num of variants: 567 Num of individuals: 95
## [###.....] After filtering  Num of variants: 567 Num of individuals: 95
```

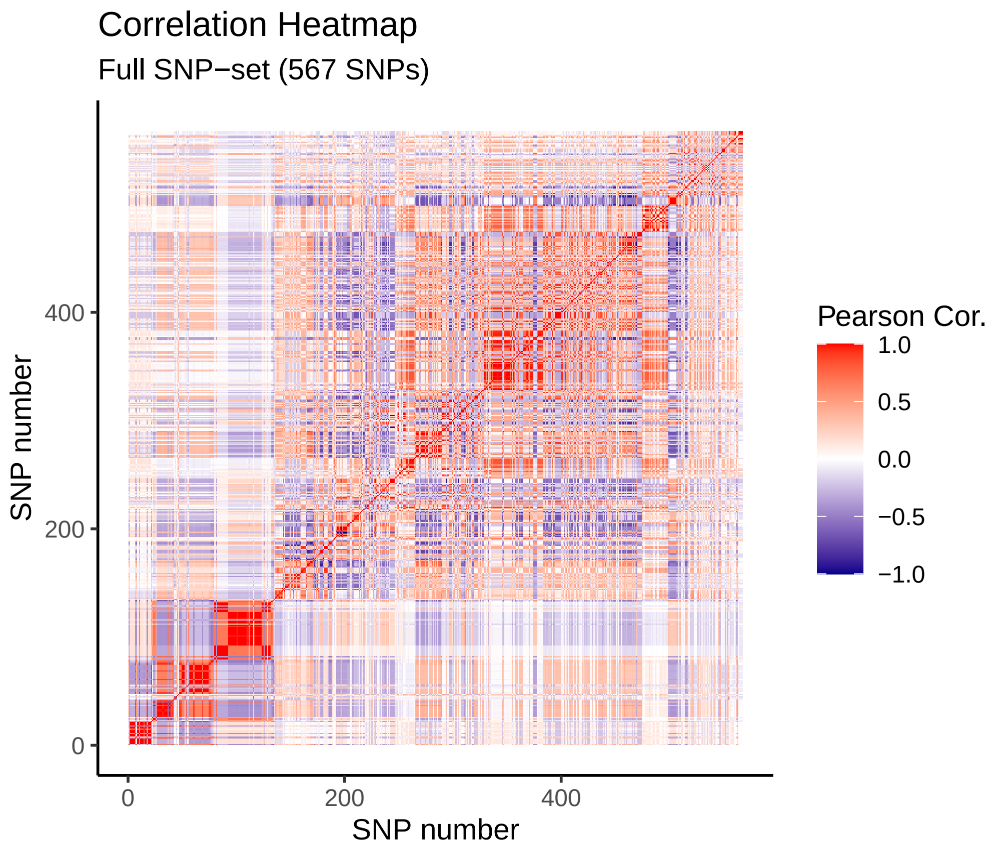
Details on the reference file are seen above. The full SNP-set contains $p = 567$ SNPs, with the reference data comprised of 95 observations. We simulate a sample of size $n = 1000$:

```
X <- simulateDataX()
```

```
## [#####...] Creating SIM object
## [#####...] Haplodata object created
## Downloading genetic map for chromosome  4
##  -> Downloading genetic map from: https://github.com/adimitromanolakis/geneticMap-GRCh37/raw/master/
##  -> Saving genetic map to:  C:\Users\Brian\AppData\Local\Temp\Rtmp0IvCCc/genetic_map_GRCh37_chr4.txt
##       -> Genetic map has 211115 entries
##    user  system elapsed
##    0.78    0.00    0.78
##    user  system elapsed
##    0.77    0.00    0.76
##    user  system elapsed
##    0.75    0.02    0.77
##    user  system elapsed
##    0.75    0.01    0.77
```

We produce a sample correlation heatmap for the simulated data:

```r
heatmap(X, "Full SNP-set")
```



Signal Sets with Weakly Correlated Components

We consider the two signal sets comprised of SNPs among which the pairwise correlation was typically weak.

Sparse Signal Set (28 signals)

The scenario initialized for the full SNP set in the previous section corresponds to H_1, with 28 signal variables present among the 567 SNPs (referred to as the sparse setting). We check the indices of $\boldsymbol{X}$ that correspond to the signal variables:

```r
all_signals <- sort(unique(c(
  signal_indices_shared, signal_indices_y1, signal_indices_y2_only,
  signal_indices_y3, signal_indices_y4)))

all_signals
```

```
##  [1] 170 172 173 174 175 176 177 214 215 216 217 218 219 220 513 514 515 517 519
## [20] 520 521 540 541 542 543 544 546 547
```

We check the total number of variables in the signal set:

```r
length(all_signals)
```

```
## [1] 28
```

We check both which and how many signal variables affect all four components of $\boldsymbol{Y}$:

```
# X variables affecting all Y components:
signal_indices_shared
```

```
##  [1] 170 172 175 176 215 216 219 220 515 517 521 540 543 544
```

```
# Number of X variables affecting all Y components:
length(signal_indices_shared)
```

```
## [1] 14
```

We check the signal variables affecting Y_1:

```
# X variables affecting Y_1:
sort(signal_indices_y1)
```

```
##  [1] 170 172 173 175 176 177 215 216 217 219 220 513 515 517 519 521 540 541 543
## [20] 544 546
```

```
# Number of X variables affecting Y_1:
length(signal_indices_y1)
```

```
## [1] 21
```

We check the signal variables affecting Y_2.

```
# X variables affecting Y_2:
sort(unique(c(signal_indices_shared, signal_indices_y2_only)))
```

```
##  [1] 170 172 173 174 175 176 177 214 215 216 217 218 219 220 513 514 515 517 519
## [20] 520 521 540 541 542 543 544 546 547
```

```
# Number of X variables affecting Y_2:
length(
  sort(unique(c(signal_indices_shared, signal_indices_y2_only))))
```

```
## [1] 28
```

We check the signal variables affecting Y_3:

```
# X variables affecting Y_3:
sort(signal_indices_y3)
```

```
##  [1] 170 172 174 175 176 214 215 216 218 219 220 514 515 517 520 521 540 542 543
## [20] 544 547
```

```
# Number of X variables affecting Y_3:
length(signal_indices_y3)
```

```
## [1] 21
```

We check the signal variables affecting Y_4:

```
# X variables affecting Y_4:
sort(signal_indices_y4)
```

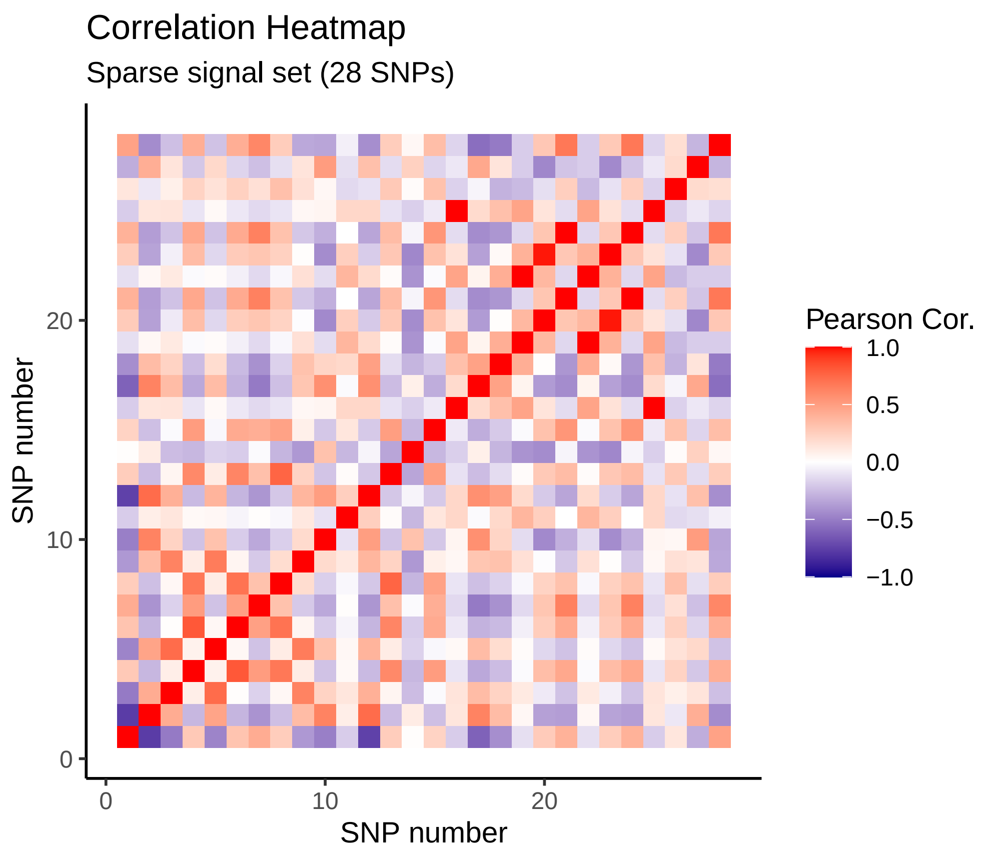
```
##  [1] 170 172 173 174 175 176 177 214 215 216 217 218 219 220 513 514 515 517 519
## [20] 520 521 540 541 542 543 544 546 547
```

```
# Number of X variables affecting Y_4:
length(signal_indices_y4)
```

```
## [1] 28
```

A correlation heatmap for the SNPs in the signal set is seen below.

```
heatmap(X[ , all_signals], "Sparse signal set")
```



Dense Signal Set (122 signals)

We now initialize a scenario for the dense setting, which involves 122 signal variables among the 567 SNPs:

```
signal_density <- "dense"
source(file.path(dir_src, 'initialize_simulation_scenario.R'))
```

```
## [#.......] Reading VCF file..

## Rows: 569 Columns: 104
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (101): ID, REF, ALT, FILTER, INFO, FORMAT, NA06984, NA06989, NA12347, NA...
## dbl   (3): #CHROM, POS, QUAL
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## [##......] Chromosome:   4  Mbp:  77.35628  Region Size:  347.154 kb  Num of individuals: 95
## [##......] Before filtering  Num of variants: 567 Num of individuals: 95
## [###.....] After filtering  Num of variants: 567 Num of individuals: 95
```

As before, we simulate data for the scenario:

```
X <- simulateDataX()
```

```
## [#####...] Creating SIM object
## [#####...] Haplodata object created
##    user  system elapsed
##    0.76    0.00    0.77
##    user  system elapsed
##    0.78    0.00    0.78
##    user  system elapsed
##    0.76    0.00    0.77
##    user  system elapsed
##    0.77    0.00    0.77
```

We check the indices of $\boldsymbol{X}$ that correspond to the signal variables:

```
all_signals <- sort(unique(c(
  signal_indices_shared, signal_indices_y1, signal_indices_y2_only,
  signal_indices_y3, signal_indices_y4)))

all_signals
```

```
##   [1] 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149
##  [19] 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
##  [37] 168 169 170 171 172 173 174 175 176 177 178 213 214 215 216 217 218 219
##  [55] 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237
##  [73] 238 239 240 241 242 243 505 506 507 508 509 510 511 512 513 514 515 516
##  [91] 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534
## [109] 535 536 537 538 539 540 541 542 543 544 545 546 547 549
```

We check the total number of variables in the signal set:

```
length(all_signals)
```

```
## [1] 122
```

We check both which and how many signal variables affect all four components of $\boldsymbol{Y}$:

```
# X variables affecting all Y components:
signal_indices_shared
```

```
##  [1] 132 133 136 137 140 141 144 145 148 149 152 153 156 157 160 161 164 165 168
## [20] 169 172 173 176 177 214 215 218 219 222 223 226 227 230 231 234 235 238 239
## [39] 242 243 507 508 511 512 515 516 519 520 523 524 527 528 531 532 535 536 539
## [58] 540 543 544 547
```

```
# Number of X variables affecting all Y components:
length(signal_indices_shared)
```

```
## [1] 61
```

We check the signal variables affecting Y_1:

```
# X variables affecting Y_1:
sort(signal_indices_y1)
```

```
##  [1] 132 133 134 136 137 138 140 141 142 144 145 146 148 149 150 152 153 154 156
## [20] 157 158 160 161 162 164 165 166 168 169 170 172 173 174 176 177 178 214 215
## [39] 216 218 219 220 222 223 224 226 227 228 230 231 232 234 235 236 238 239 240
## [58] 242 243 505 507 508 509 511 512 513 515 516 517 519 520 521 523 524 525 527
## [77] 528 529 531 532 533 535 536 537 539 540 541 543 544 545 547 549
```

```
# Number of X variables affecting Y_1:
length(signal_indices_y1)
```

```
## [1] 92
```

We check the signal variables affecting Y_2.

```
# X variables affecting Y_2:
sort(unique(c(signal_indices_shared, signal_indices_y2_only)))
```

```
##   [1] 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149
##  [19] 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
##  [37] 168 169 170 171 172 173 174 175 176 177 178 213 214 215 216 217 218 219
##  [55] 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237
##  [73] 238 239 240 241 242 243 505 506 507 508 509 510 511 512 513 514 515 516
##  [91] 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534
## [109] 535 536 537 538 539 540 541 542 543 544 545 546 547 549
```

```
# Number of X variables affecting Y_2:
length(
  sort(unique(c(signal_indices_shared, signal_indices_y2_only))))
```

```
## [1] 122
```

We check the signal variables affecting Y_3:

```
# X variables affecting Y_3:
sort(signal_indices_y3)
```

```
##  [1] 132 133 135 136 137 139 140 141 143 144 145 147 148 149 151 152 153 155 156
## [20] 157 159 160 161 163 164 165 167 168 169 171 172 173 175 176 177 213 214 215
## [39] 217 218 219 221 222 223 225 226 227 229 230 231 233 234 235 237 238 239 241
## [58] 242 243 506 507 508 510 511 512 514 515 516 518 519 520 522 523 524 526 527
## [77] 528 530 531 532 534 535 536 538 539 540 542 543 544 546 547
```

```
# Number of X variables affecting Y_3:
length(signal_indices_y3)
```

```
## [1] 91
```

We check the signal variables affecting Y_4:

```
# X variables affecting Y_4:
sort(signal_indices_y4)
```

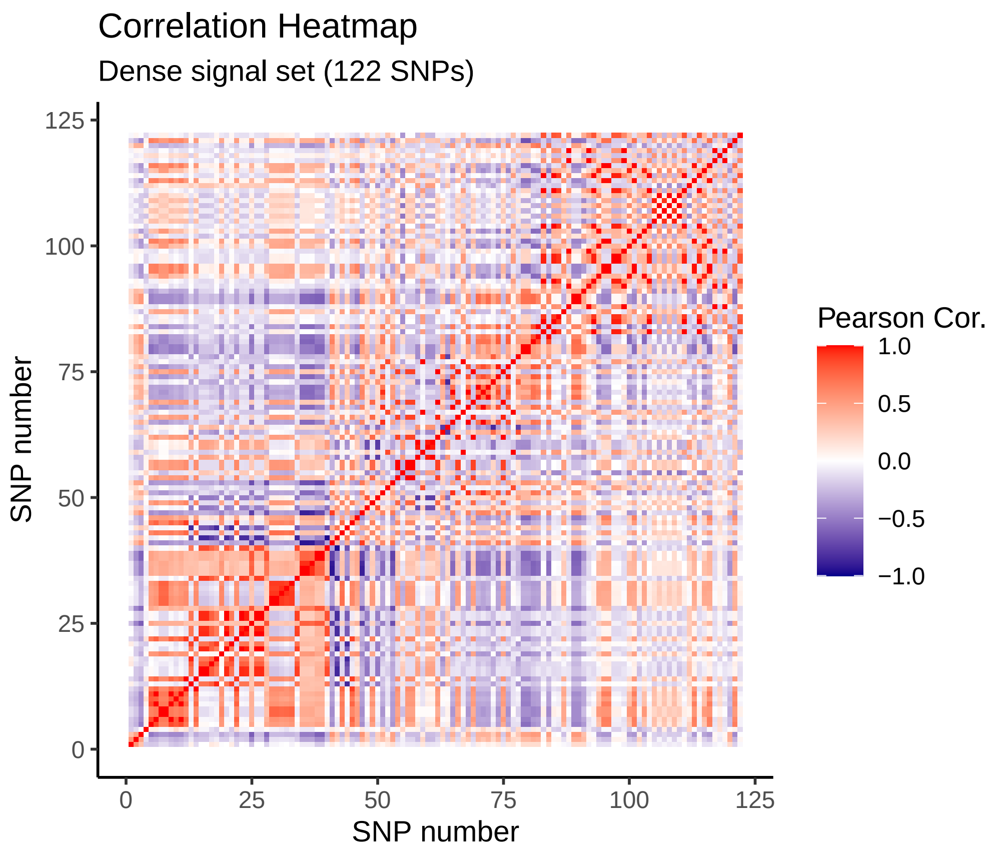
```
##   [1] 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149
##  [19] 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
##  [37] 168 169 170 171 172 173 174 175 176 177 178 213 214 215 216 217 218 219
##  [55] 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237
##  [73] 238 239 240 241 242 243 505 506 507 508 509 510 511 512 513 514 515 516
##  [91] 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534
## [109] 535 536 537 538 539 540 541 542 543 544 545 546 547 549
```

```
# Number of X variables affecting Y_4:
length(signal_indices_y4)
```

```
## [1] 122
```

A correlation heatmap for the SNPs in the signal set is seen below.

```
heatmap(X[ , all_signals], "Dense signal set")
```



Signal Sets with Strongly Correlated Components

We consider the two signal sets comprised of SNPs among which the pairwise correlation was typically strong. The variables for each of these signal sets comprise a contiguous region at the beginning of the SNP set.

Sparse Signal Set (28 signals)

We initialize the simulation scenario and simulate the data:

```
signal_correlation <- "high"
signal_density <- "sparse"
source(file.path(dir_src, 'initialize_simulation_scenario.R'))
```

```
## [#.......] Reading VCF file..

## Rows: 569 Columns: 104
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (101): ID, REF, ALT, FILTER, INFO, FORMAT, NA06984, NA06989, NA12347, NA...
## dbl   (3): #CHROM, POS, QUAL
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## [##......] Chromosome:   4  Mbp:  77.35628  Region Size:  347.154 kb  Num of individuals: 95
## [##......] Before filtering  Num of variants: 567 Num of individuals: 95
## [###.....] After filtering  Num of variants: 567 Num of individuals: 95
```

```r
X <- simulateDataX()
```

```
## [#####...] Creating SIM object
## [#####...] Haplodata object created
##    user  system elapsed
##    0.75    0.00    0.76
##    user  system elapsed
##    0.77    0.00    0.76
##    user  system elapsed
##    0.76    0.00    0.77
##    user  system elapsed
##    0.76    0.00    0.79
```

We check the indices of $\boldsymbol{X}$ that correspond to the signal variables:

```r
all_signals <- sort(unique(c(
  signal_indices_shared, signal_indices_y1, signal_indices_y2_only,
  signal_indices_y3, signal_indices_y4)))

all_signals
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28
```

We check the total number of variables in the signal set:

```r
length(all_signals)
```

```
## [1] 28
```

We check both which and how many signal variables affect all four components of $\boldsymbol{Y}$:

```r
# X variables affecting all Y components:
signal_indices_shared
```

```
##  [1]  1  2  5  6  9 10 13 14 17 18 21 22 25 26
```

```r
# Number of X variables affecting all Y components:
length(signal_indices_shared)
```

```
## [1] 14
```

We check the signal variables affecting Y_1:

```r
# X variables affecting Y_1:
sort(signal_indices_y1)
```

```
##  [1]  1  2  3  5  6  7  9 10 11 13 14 15 17 18 19 21 22 23 25 26 27
```

```r
# Number of X variables affecting Y_1:
length(signal_indices_y1)
```

```
## [1] 21
```

We check the signal variables affecting Y_2.

```r
# X variables affecting Y_2:
sort(unique(c(signal_indices_shared, signal_indices_y2_only)))
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28
```

```r
# Number of X variables affecting Y_2:
length(
  sort(unique(c(signal_indices_shared, signal_indices_y2_only))))
```

```
## [1] 28
```

We check the signal variables affecting Y_3:

```r
# X variables affecting Y_3:
sort(signal_indices_y3)
```

```
##  [1]  1  2  4  5  6  8  9 10 12 13 14 16 17 18 20 21 22 24 25 26 28
```

```r
# Number of X variables affecting Y_3:
length(signal_indices_y3)
```

```
## [1] 21
```

We check the signal variables affecting Y_4:

```r
# X variables affecting Y_4:
sort(signal_indices_y4)
```

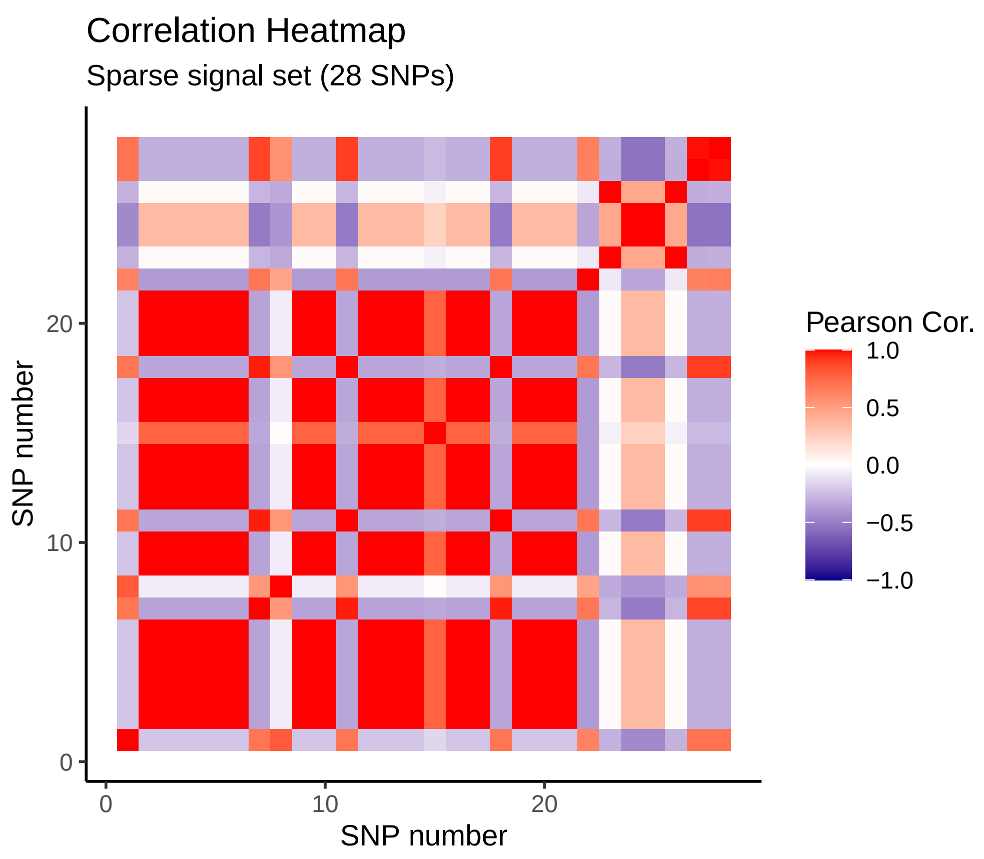
```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28
```

```r
# Number of X variables affecting Y_4:
length(signal_indices_y4)
```

```
## [1] 28
```

A correlation heatmap for the SNPs in the signal set is seen below.

```r
heatmap(X[ , all_signals], "Sparse signal set")
```



Dense Signal Set (122 signals)

We initialize the simulation scenario and simulate the data:

```
signal_density <- "dense"
source(file.path(dir_src, 'initialize_simulation_scenario.R'))
```

```
## [#.......] Reading VCF file..

## Rows: 569 Columns: 104
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (101): ID, REF, ALT, FILTER, INFO, FORMAT, NA06984, NA06989, NA12347, NA...
## dbl   (3): #CHROM, POS, QUAL
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## [##......] Chromosome:   4  Mbp:  77.35628  Region Size:  347.154 kb  Num of individuals: 95
## [##......] Before filtering  Num of variants: 567 Num of individuals: 95
## [###.....] After filtering  Num of variants: 567 Num of individuals: 95
```

```
X <- simulateDataX()
```

```
## [#####...] Creating SIM object
## [#####...] Haplodata object created
##    user  system elapsed
##    0.75    0.00    0.77
##    user  system elapsed
##    0.76    0.02    0.78
##    user  system elapsed
##    0.77    0.00    0.77
##    user  system elapsed
##    0.76    0.00    0.77
```

We check the indices of $\boldsymbol{X}$ that correspond to the signal variables:

```
all_signals <- sort(unique(c(
  signal_indices_shared, signal_indices_y1, signal_indices_y2_only,
  signal_indices_y3, signal_indices_y4)))

all_signals
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 123
```

We check the total number of variables in the signal set:

```
length(all_signals)
```

```
## [1] 122
```

We check both which and how many signal variables affect all four components of $\boldsymbol{Y}$:

```
# X variables affecting all Y components:
signal_indices_shared
```

```
##  [1]   1   2   5   6   9  10  13  14  17  18  21  22  25  26  29  30  33  34  37
## [20]  38  41  42  45  46  49  50  53  54  57  58  61  62  65  66  69  70  73  74
## [39]  77  78  81  82  85  86  89  90  93  94  97  98 101 102 105 106 109 110 113
## [58] 114 117 118 121
```

```
# Number of X variables affecting all Y components:
length(signal_indices_shared)
```

```
## [1] 61
```

We check the signal variables affecting Y_1:

```
# X variables affecting Y_1:
sort(signal_indices_y1)
```

```
##  [1]   1   2   3   5   6   7   9  10  11  13  14  15  17  18  19  21  22  23  25
## [20]  26  27  29  30  31  33  34  35  37  38  39  41  42  43  45  46  47  49  50
## [39]  51  53  54  55  57  58  59  61  62  63  65  66  67  69  70  71  73  74  75
## [58]  77  78  79  81  82  83  85  86  87  89  90  91  93  94  95  97  98  99 101
## [77] 102 103 105 106 107 109 110 111 113 114 115 117 118 119 121 123
```

```
# Number of X variables affecting Y_1:
length(signal_indices_y1)
```

```
## [1] 92
```

We check the signal variables affecting Y_2.

```
# X variables affecting Y_2:
sort(unique(c(signal_indices_shared, signal_indices_y2_only)))
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 123
```

```
# Number of X variables affecting Y_2:
length(
  sort(unique(c(signal_indices_shared, signal_indices_y2_only))))
```

```
## [1] 122
```

We check the signal variables affecting Y_3:

```
# X variables affecting Y_3:
sort(signal_indices_y3)
```

```
##  [1]   1   2   4   5   6   8   9  10  12  13  14  16  17  18  20  21  22  24  25
## [20]  26  28  29  30  32  33  34  36  37  38  40  41  42  44  45  46  48  49  50
## [39]  52  53  54  56  57  58  60  61  62  64  65  66  68  69  70  72  73  74  76
## [58]  77  78  80  81  82  84  85  86  88  89  90  92  93  94  96  97  98 100 101
## [77] 102 104 105 106 108 109 110 112 113 114 116 117 118 120 121
```

```
# Number of X variables affecting Y_3:
length(signal_indices_y3)
```

```
## [1] 91
```

We check the signal variables affecting Y_4:

```
# X variables affecting Y_4:
sort(signal_indices_y4)
```

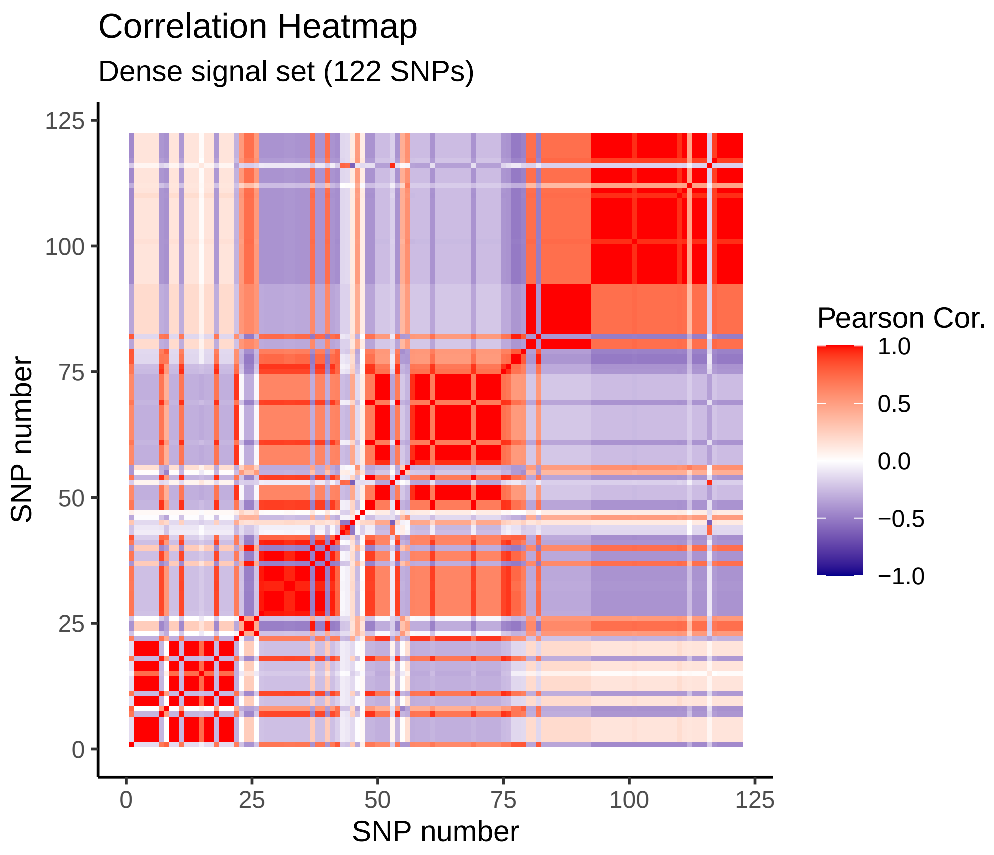
```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 123
```

```
# Number of X variables affecting Y_4:
length(signal_indices_y4)
```

```
## [1] 122
```

A correlation heatmap for the SNPs in the signal set is seen below.

```
heatmap(X[ , all_signals], "Dense signal set")
```


Correlation Heatmap
Dense signal set (122 SNPs)
SNP number
SNP number
0
25
50
75
100
125
Pearson Cor.
1.0
0.5
0.0
−0.5
−1.0