R で GWAS と GS
チュートリアル

2018.5.9

岩田洋佳

本チュートリアルでは、R を用いて GWAS と GS のための予測モデル構築を行う方法について説明する。

（1） 例として利用するデータ

ここでは、R を用いて GWAS および GS モデル構築を行う方法を解説する。なお、Zhao ら（2011; Nature Communications 2:467）がイネ遺伝資源の GWAS に用いた表現型データと、McCouch ら（2016; Nature Communication 7:10532）が用いた High Density Rice Array (HDRA)を用いてジェノタイピングされた SNP 遺伝子型データ（いずれも、Rice Diversity http://www.ricediversity.org/data/ からダウンロードできる）を、例として用いる。後者のデータは、70 万 SNPs のデータのうち、他のマーカーとの冗長性が無く、遺伝子型データの欠測率が低く（< 1%）、かつ、マイナーアリル頻度（minor allele frequency: MAF）が高い（> 5%）、38,769 SNPs のデータを用いる。

（2） 利用する R のパッケージ

SNP データの読み込み・解析に vcfR と pcaMethods を、GWAS には rrBLUP と qqman パッケージを、GS の予測モデル構築には glmnet パッケージを用いる。いずれも R のパッケージインストーラで簡単にダウンロード・インストールが可能である。

```
> # the following libraries are necessary for this script
> require(vcfR)
 要求されたパッケージ vcfR をロード中です
 (省略)
> require(pcaMethods)
 要求されたパッケージ pcaMethods をロード中です
 (省略)
> require(rrBLUP)
 要求されたパッケージ rrBLUP をロード中です
> require(qqman)
 要求されたパッケージ qqman をロード中です
 (省略)
> require(glmnet)
 要求されたパッケージ glmnet をロード中です
 (省略)
```

（3） GWAS の実行

まず、vcf 形式のマーカー遺伝子型データ（McCouch ら 2016; Nature Communication 7:10532）を読み込む。vcf 形式のデータの読み込み、処理には、vcfR パッケージに含まれる関数を用いる。

```
> # read compressed vcf file (*.vcf.gz)
> vcf <- read.vcfR("CSHL_EVA_Release_HDRA/HDRA-G6-4-RDP1-RDP2-NIAS.AGCT_MAF005MIS001NR.vcf.gz")
Scanning file to determine attributes.
File attributes:
  meta lines: 16
  header line: 17
  variant count: 38769
  column count: 1577
Meta line 16 read in.
All meta lines processed.
gt matrix initialized.
Processed variant: 38769
All variants processed
9
```

次に、読み込んだデータから遺伝子型データ、染色体番号、物理位置の情報を取り出し、最初の 10 系統 10 SNPs の情報を表示する。

```
> # extract genotype data from vcf
> gt <- extract.gt(vcf)
>
> # get marker information (chromosone numbers and positions)
> chrom <- getCHROM(vcf)
> pos <- getPOS(vcf)
>
> # show the first 10 SNPs of the first 10 lines
> gt[1:10, 1:10]
           IRGC121316@c88dcbba.0 IRGC121578@950ba1f2.0 IRGC121491@0b7d031b.0
SNP-1.8562.  "0/0"               "0/0"                "0/0"
SNP-1.18594. "0/0"               "0/0"                "0/0"
SNP-1.24921. "0/0"               "0/0"                "0/0"
SNP-1.25253. "0/0"               "0/0"                "1/1"
SNP-1.29213. "0/0"               "0/0"                "0/0"
SNP-1.30477. "1/1"               "0/0"                "0/0"
SNP-1.31732. "0/0"               "0/0"                "1/1"
SNP-1.32666. "1/1"               "1/1"                "1/1"
SNP-1.40457. "0/0"               "0/0"                "0/0"
SNP-1.75851. "0/0"               "0/0"                "0/0"
（省略）
```

マーカー遺伝子型が、0/0, 0/1, 1/1 の形式で保存されている。各行が各 SNP、各列が各系統を表す。

読み込んだマーカー遺伝子型データを、数値データに変換する。変換は、0/0, 0/1, 1/1 をそれぞれ、-1, 0, 1 とする。ホモ接合の遺伝子型のどちらを-1 にして、どちらを 1 にするかは任意であ

るが、ここでは、0/0 を-1、1/1 を 1 とした。

```
> # create a matrix of gt scores
> gt.score <- matrix(NA, nrow(gt), ncol(gt))
> gt.score[gt == "0/0"] <- -1
> gt.score[gt == "0/1"] <- 0
> gt.score[gt == "1/1"] <- 1
>
> # name the rows and columns of matrix
> rownames(gt.score) <- rownames(gt)
> colnames(gt.score) <- colnames(gt)
>
> # transpose the matrix
> gt.score <- t(gt.score)
```

次に、表現型データ（Zhao ら 2011; Nature Communications 2:467）を読み込む。

```
> # read phenotype and line information
> line <- read.csv("Pheno/RiceDiversityLine4GWASGS.csv", row.names = 1)
> pheno <- read.csv("Pheno/RiceDiversityPheno4GWASGS.csv", row.names = 1)
>
> # show the first 6 rows of the datasets
> head(line)
               NSFTV.ID GSOR.ID       IRGC.ID Accession.Name Country.of.origin
 Latitude
NSFTV1@86f75d2b.0       1  301001 To be assigned      Agostano           Italy 41.871940
NSFTV3@5ef1be74.0       3  301003        117636  Ai-Chiao-Hong          China 27.902527
NSFTV4@81d03b86.0       4  301004        117601      NSF-TV 4           India 22.903081
NSFTV5@5533f406.0       5  301005        117641      NSF-TV 5           India 30.472664
NSFTV6@0d125c0e.0       6  301006        117603      ARC 7229           India 22.903081
NSFTV7@e37be9e5.0       7  301007 To be assigned         Arias       Indonesia -0.789275
（省略）
> head(pheno)
               Flowering.time.at.Arkansas Flowering.time.at.Faridpur
NSFTV1@86f75d2b.0                  75.08333                       64
NSFTV3@5ef1be74.0                  89.50000                       66
NSFTV4@81d03b86.0                  94.50000                       67
NSFTV5@5533f406.0                  87.50000                       70
NSFTV6@0d125c0e.0                  89.08333                       73
NSFTV7@e37be9e5.0                 105.00000                       NA
（省略）
```

表現型データがある系統のマーカー遺伝子型データだけを抜き出す。388 系統のデータが抽出される。

```
> # extract marker genotypes of lines with phenotypic data
> gt.score.wp <- gt.score[rownames(pheno),]
> dim(gt.score.wp)
[1]   388 38769
```

388 系統のもつ、集団構造（population structure）を主成分分析を用いて視覚化する。なお、通常の主成分分析では、欠測値があると実行できないが、ここでは、パッケージ pcaMethods に含まれる pca 関数を用いて確率的主成分分析（probabilistic principal component analysis）を行う。388 系統の第 1〜6 主成分得点について、分集団で色付けして表示する。分集団の情報は、先ほど読み込んだ系統データ（line）に含まれている。

```
> # perform probablistic principal component analysis (ppca) with the pca function
> # this method allows the existence of missing values in the data
> pca <- pca(gt.score.wp, "ppca", nPcs = 6)
> plot(scores(pca)[,1:2], col = line$Sub.population)
> plot(scores(pca)[,3:4], col = line$Sub.population)
> plot(scores(pca)[,5:6], col = line$Sub.population)
```

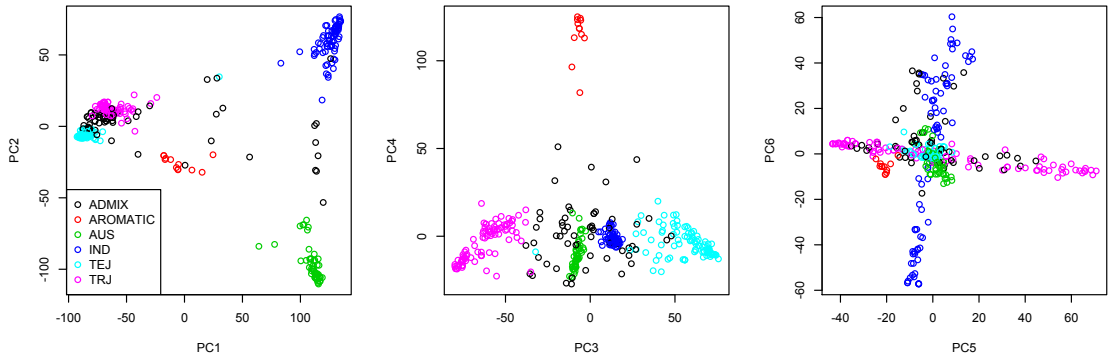

図 1. 確率的主成分分析の結果。388 系統の第 1 主成分（PC1）〜第 6 主成分までの得点を示す。

主成分分析の結果、第 1〜第 4 主成分で、分集団間の遺伝的変異がよく捉えられているのが分かる。第 5、第 6 主成分は、分集団（第 5 は温帯ジャポニカ TRJ、第 6 はインディカ IND）内の変異を捉えている。

次に、GWAS を行う準備を行う。GWAS には、rrBLUP パッケージの GWAS 関数を用いる。GWAS 関数に読み込ませるマーカー遺伝子型データと、ゲノム関係行列の準備を行う。

```
> # prepare genotype data for GWAS function
> g <- data.frame(colnames(gt.score.wp), as.numeric(chrom), pos, t(gt.score.wp))
> rownames(g) <- 1:nrow(g)
> colnames(g) <- c("marker", "chrom", "pos", rownames(gt.score.wp))
>
> # calculate the additive relational matrix of 388 lines
> amat <- A.mat(gt.score.wp, shrink = T)
[1] "Shrinkage intensity: 0"
> colnames(amat) <- rownames(amat) <- rownames(gt.score.wp)
> amat[1:6, 1:6]
               NSFTV1@86f75d2b.0 NSFTV3@5ef1be74.0 NSFTV4@81d03b86.0 NSFTV5@5533f406.0
NSFTV1@86f75d2b.0       1.39552387       -0.9535068       -0.8405541       0.01379468
NSFTV3@5ef1be74.0      -0.95350677        2.8383952        0.7739508      -0.29162221
NSFTV4@81d03b86.0      -0.84055408        0.7739508        2.7417211      -0.15323945
NSFTV5@5533f406.0       0.01379468       -0.2916222       -0.1532394       1.73549666
NSFTV6@0d125c0e.0      -0.79550134        0.7326205        2.3110755      -0.14310452
NSFTV7@e37be9e5.0       0.10480161       -0.6929374       -0.4992329       0.09055647
               NSFTV6@0d125c0e.0 NSFTV7@e37be9e5.0
NSFTV1@86f75d2b.0       -0.7955013       0.10480161
NSFTV3@5ef1be74.0        0.7326205      -0.69293738
NSFTV4@81d03b86.0        2.3110755      -0.49923292
NSFTV5@5533f406.0       -0.1431045       0.09055647
NSFTV6@0d125c0e.0        2.6871199      -0.46973007
NSFTV7@e37be9e5.0       -0.4697301       1.43917061
```

次に、表現型データの準備をする。ここでは、種子の縦横比を例に解析を行う。

```
> # select a trait (actually, you can select multiple traits here)
> y <- pheno$Seed.length.width.ratio
>
> # prepare phenotypic value for GWAS
> p <- data.frame(rownames(gt.score.wp), y)
> colnames(p) <- c("gid", "y")
```

早速、GWAS を行ってみる。マーカー数が多いので少し時間がかかる。計算が終わったら、最初の 6 SNP についての結果を表示してみる。結果は、マーカー名、染色体番号、物理位置、$-log10(p)$（検定結果のp値の対数の負値をとったもの）として表示される。$-log10(p)$は、大きいほど有意であることを示している。

```
> # GWAS with Q (population structure) + K (kinship relatedness)
> # minor allele frequencuy is set as 0.05
> gwa <- GWAS(p, g, K = amat, n.PC = 4, min.MAF = 0.05, plot = F)
[1] "GWAS for trait: y"
[1] "Variance components estimated. Testing markers."
>
> # show the result of GWAS for the first six SNPs
> head(gwa)
       marker chrom  pos         y
1  SNP-1.8562.     1  9563 0.133228972
2 SNP-1.18594.     1 19595 0.000000000
3 SNP-1.24921.     1 25922 0.118017484
4 SNP-1.25253.     1 26254 0.088053310
5 SNP-1.29213.     1 30214 0.170218610
6 SNP-1.30477.     1 31478 0.007463631
```

なお、2 番めの SNP で、$-log10(p)$が 0 となっている。これは一体何なんだろうか。実は、GWAS 関数では、minor allele frequency（MAF）が指定した値より低いものについては、計算を実施せずに、結果として$-log10(p)$に 0 の値を返す仕様になっている。このことを自分で MAF を計算して、確かめてみる。

```
> # calculate maf by ourselves
> p.al1 <- apply(gt.score.wp + 1, 2, mean, na.rm = T) / 2
> maf <- pmin(p.al1, 1 - p.al1)
> head(maf)
 SNP-1.8562. SNP-1.18594. SNP-1.24921. SNP-1.25253. SNP-1.29213. SNP-1.30477.
  0.13788660   0.04392765   0.12823834   0.28552972   0.13436693   0.23453608
>
> # check the relationship and replace -logp of 0 with NA
> gwa$y[maf < 0.05][1:50]
 [1] 0.0000000 0.6208820 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[10] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[19] 1.8415559 1.8806587 1.7430450 1.3983617 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[28] 0.0000000 0.0000000 0.5190228 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[37] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[46] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
```

表現型データに欠測があるために、完全に結果は一致しないが、MAF が 0.05 以下のマーカーの$-log10(p)$の値はほぼ 0 となっている。この$-log10(p)$の値が 0 の結果が残ったままでは、この後の偽発見率（false discovery rate: FDR）の計算などにも差し支える。そこで、それら結果を NA（欠測）で置き換えることにする。

```
> # replace -log P = 0 with NA
> gwa$y[gwa$y == 0] <- NA
```

得られた結果を、マンハッタンプロットを描いて視覚化する。マンハッタンプロットの描画には、qqman パッケージの manhattan 関数を用いる。なお、rrBLUP パッケージの GWAS 関数では、結果が$-log10(p)$のかたちで得られるが、manhattan 関数は、p値そのものを入力する必要があるため、値の変換（$p = 10^{log10(p)}$）を行う。

```
> # draw manhattan plot
> mht <- data.frame(SNP = gwa$marker, CHR = gwa$chrom, BP = gwa$pos, P = 10^(-gwa$y))
> mht <- na.omit(mht)
> manhattan(mht)
```

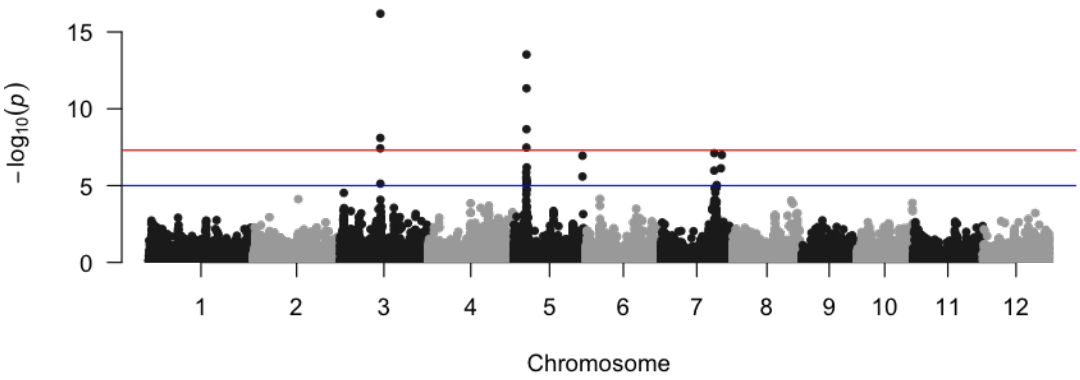

図 2. 種子の縦横比を例に行った GWAS の結果。第 3、第 5 に高度に有意なアソシエーションが見られる。

縦軸は、$-log10(p)$の値であり、これが大きいほど表現型と SNP の遺伝子型のアソシエーションが有意である。なお、qqman パッケージの qq 関数を用いることで、QQ プロットを描くことができる。

```
> # draw qq plot
> qq(mht$P)
```

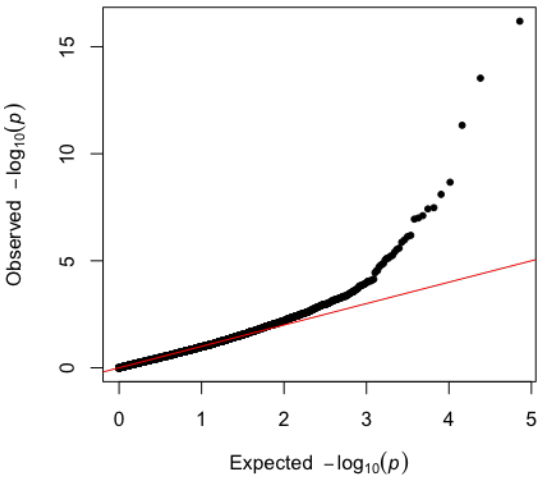

図 3. GWAS の$-log10(p)$値の QQ プロット。横軸が、全ての SNP についてアソシエーションが無い場合に期待される$-log10(p)$の値、縦軸が、実際に観察された$-log10(p)$の値を示す。

QQ プロットは、全ての SNP について、表現型と SNP 間のアソシエーションが無い場合に期待される$-log10(p)$の値と、実際に観察された$-log10(p)$の値を対散布したものである。描かれる点が大きく$y = x$の直線から逸脱する場合には、集団構造の存在などが原因で偽陽性が生じている可能性が示唆される。この場合は、$-log10(p)$が大きな SNP について逸脱が見られるが、これは、これらの SNP については、表現型とのアソシエーションが存在するため、と考えるのが妥当である。

GWAS では、多数の SNP についてそれぞれ仮説検定を行うため、有意水準（有意と判断するp値あるいは$-log10(p)$値）を適切に調整することが重要である。例えば、よく仮説検定で用いる p < 0.01 という基準を用いた場合、3 万 8 千個の SNP について検定を行えば、（すべての SNP が独立だとすると）380 個の SNP が有意と判断されてしまうことになる。したがって、より厳しい基準が必要となる。ここでは、まず、偽発見率（false discovery rate: FDR）をもとに、基準を決める。偽発見率を 0.05（5%）として p.adjust 関数を用いて計算を行う。

```
> # calculate false discovery rate
> fdr <- p.adjust(mht$P, method = "BH")
>
> # select markers under the threshold
> fdr.thresh = 0.05
> sig <- mht[fdr < fdr.thresh, ]
> dim(sig)
[1] 29  4
> sig
               SNP CHR      BP          P
9566   SNP-3.1595841.   3  1596846 2.977973e-05
11357 SNP-3.16686770.   3 16688126 3.771125e-08
11361 SNP-3.16718013.   3 16719368 7.916753e-09
11362 SNP-3.16732086.   3 16733441 6.455920e-17
11363 SNP-3.16735968.   3 16737323 7.528046e-06
17088  SNP-5.5249928.   5  5249951 1.353555e-06
17089  SNP-5.5265891.   5  5265914 2.959608e-06
17091  SNP-5.5298269.   5  5298292 3.624749e-05
17095  SNP-5.5321834.   5  5321857 3.948053e-06
17096  SNP-5.5327890.   5  5327913 3.319451e-08
17097  SNP-5.5338182.   5  5338205 7.660721e-06
17099  SNP-5.5362461.   5  5362484 2.138316e-09
17102  SNP-5.5371749.   5  5371772 2.951114e-14
17103  SNP-5.5372932.   5  5372955 4.681594e-12
17104  SNP-5.5375826.   5  5375849 1.650437e-05
17107  SNP-5.5388340.   5  5388363 1.952519e-05
17111  SNP-5.5442800.   5  5442823 6.492041e-07
17112  SNP-5.5458356.   5  5458379 6.090497e-06
17117  SNP-5.5516131.   5  5516194 5.670845e-06
19184 SNP-5.28470501.   5 28533147 2.570930e-06
19185 SNP-5.28485355.   5 28548001 1.131917e-07
24581 SNP-7.22056570.   7 22057564 1.052957e-06
24586 SNP-7.22112016.   7 22113010 7.803667e-08
24611 SNP-7.22251179.   7 22252173 1.483201e-05
24636 SNP-7.22528645.   7 22529639 2.874789e-05
24700 SNP-7.23047484.   7 23048478 1.337573e-05
24709 SNP-7.23103184.   7 23104178 9.121408e-06
24892 SNP-7.24844031.   7 24845026 7.454728e-07
```

FDR < 0.05 を基準とすると、29 SNP が有意と判断される。特に、第 3 染色体の 16733441（4 行目）と、第 5 染色体の 5371772（13 行目）に高度に有意なアソシエーションが検出されていることが分かる。

有意と判断された SNP に色付けをして、マンハッタンプロットを描く。

```
> # draw manhattan plot with coloring significant SNPs
> manhattan(mht, highlight = sig$SNP)
```

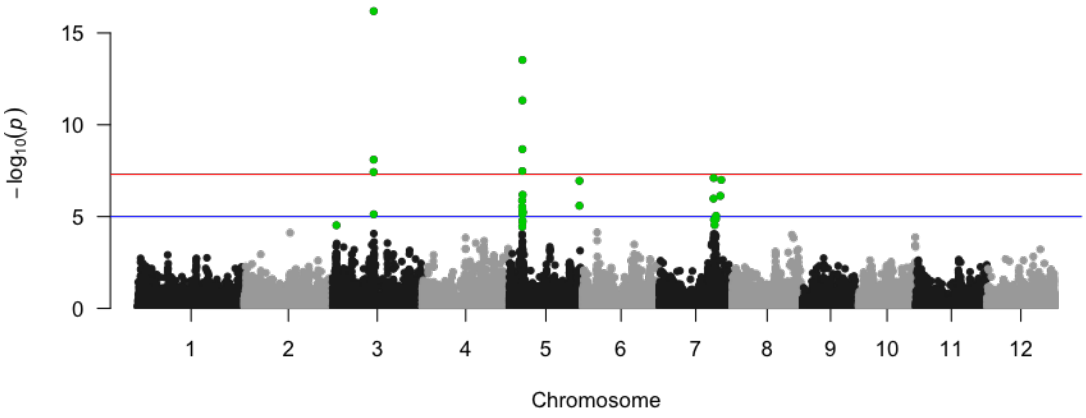

図 4. 偽陽性率が 5%以下の SNP に色付けしたマンハッタンプロット。

有意なアソシエーションが検出された第 3, 5, 7 染色体をクローズアップする。

```
> # close up the 3rd, 5th, and 7th chromosomes
> manhattan(subset(mht, CHR %in% c(3,5,7)), highlight = sig$SNP)
```

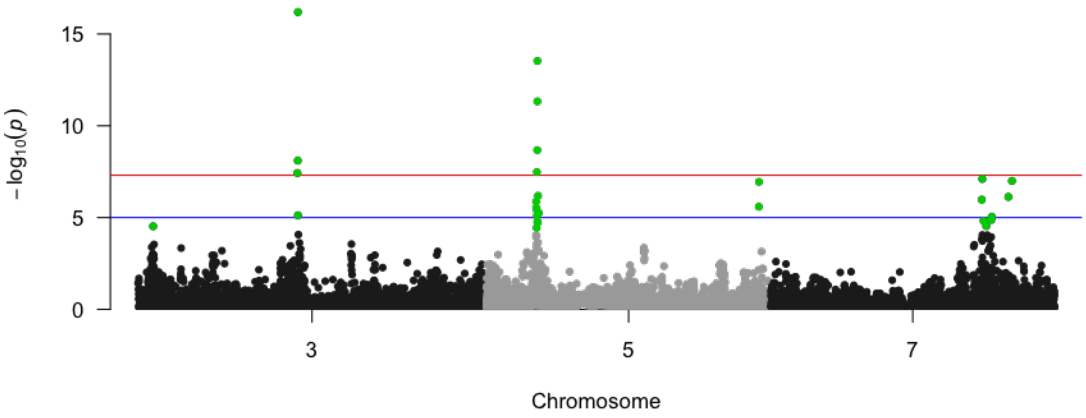

図 5. 第 3、第 5、第 7 染色体についてのマンハッタンプロット。

では、有意なアソシエーションを検出する基準を変更してみよう。ここでは、Bonferroni 補正を行ったp値を基準として、有意なアソシエーションを検出してみる。

```
> # calculate false discovery rate
> p.bonferroni <- p.adjust(mht$P, method = "bonferroni")
>
> # select markers under the threshold
> p.thresh = 0.05
> sig <- mht[p.bonferroni < p.thresh, ]
> dim(sig)
[1] 14  4
> sig
               SNP CHR      BP          P
11357 SNP-3.16686770.   3 16688126 3.771125e-08
11361 SNP-3.16718013.   3 16719368 7.916753e-09
11362 SNP-3.16732086.   3 16733441 6.455920e-17
17088  SNP-5.5249928.   5  5249951 1.353555e-06
17096  SNP-5.5327890.   5  5327913 3.319451e-08
17099  SNP-5.5362461.   5  5362484 2.138316e-09
17102  SNP-5.5371749.   5  5371772 2.951114e-14
17103  SNP-5.5372932.   5  5372955 4.681594e-12
17111  SNP-5.5442800.   5  5442823 6.492041e-07
19185 SNP-5.28485355.   5 28548001 1.131917e-07
24581 SNP-7.22056570.   7 22057564 1.052957e-06
24586 SNP-7.22112016.   7 22113010 7.803667e-08
24892 SNP-7.24844031.   7 24845026 7.454728e-07
24914 SNP-7.25213656.   7 25214651 1.004278e-07
```

ボンフェローニ補正のほうが、相対的に保守的な検定結果となる。再度、マンハッタンプロットも描いておく。

```
> # draw manhattan plot with coloring significant SNPs
> manhattan(mht, highlight = sig$SNP)
```

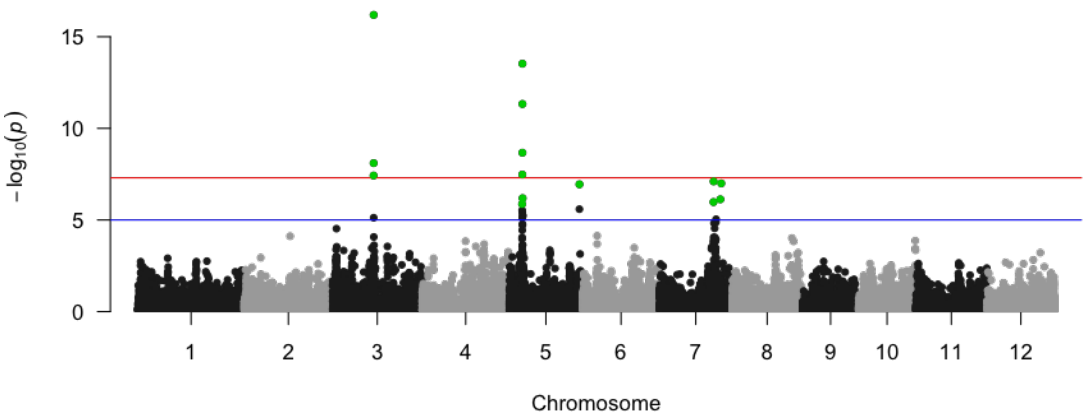

図 6. ボンフェローニ補正を行った結果。有意水準 0.05 で有意なマーカーに色付けしてある。なお、第 3 染色体の高度に有意なアソシエーションは GS3 によるもの、第 5 染色体の高度に有

意なアソシエーションは GW5 である。

種子の縦横比とは異なる形質でも GWAS を行ってみる。ここでは、個体あたりの稔性のある頴花数で GWAS を行ってみる。

```
> # select a trait
> # Number of fertile florets per plant
> y <- pheno$Panicle.number.per.plant * pheno$Florets.per.panicle *
 pheno$Panicle.fertility
> # prepare phenotypic value for GWAS
> p <- data.frame(rownames(gt.score.wp), y)
> colnames(p) <- c("gid", "y")
>
> # GWAS with Q (population structure) + K (kinship relatedness)
> # minor allele frequencuy is set as 0.05
> gwa <- GWAS(p, g, K = amat, n.PC = 4, min.MAF = 0.05, plot = F)
[1] "GWAS for trait: y"
[1] "Variance components estimated. Testing markers."
```

得られた結果をもとに、マンハッタンプロットを描く。

```
> # draw manhattan plot
> mht <- data.frame(SNP = gwa$marker, CHR = gwa$chrom, BP = gwa$pos, P = 10^(-gwa$y))
> mht <- na.omit(mht)
> manhattan(mht)
```

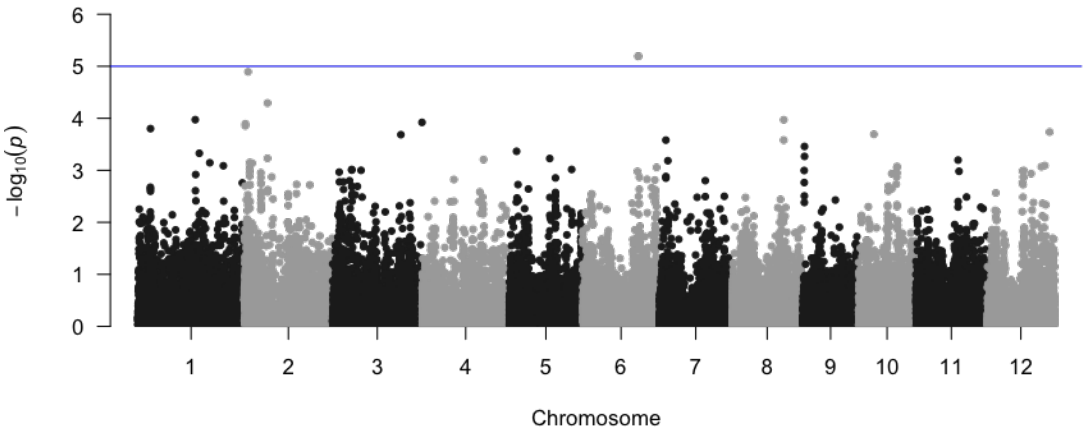

図 7. 個体あたりの稔性のある頴花数で行った GWAS の結果。目立って有意なマーカーが見られない。

儀発見率を計算して、FDR < 0.05 で有意なマーカーを抽出してみる。

```
> # calculate false discovery rate
> fdr <- p.adjust(mht$P, method = "BH")
> fdr.thresh = 0.05
> sig <- mht[fdr < fdr.thresh, ]
> sig
[1] SNP CHR BP  P
 <0 行> (または長さ 0 の row.names)
```

FDR < 0.05 を基準とすると、有意なマーカーは無い。これは、解析した形質の遺伝率や原因遺伝子の数、効果の大きさに依存していると考えられる。種子の縦横比の場合は、遺伝率が高く、かつ、効果の大きな遺伝子（GS3、GW5 など）が存在するために有意なアソシエーションが検出された。個体あたりの稔性のある頴花数は、種子の縦横比に比べると遺伝率が低く、かつ、効果の大きな遺伝子が少ないか存在しない（効果の大きくない遺伝子が複数存在する）と考えられる。

では、ここから、ゲノミックセレクション（genomic selection: GS)で用いられるゲノミック予測（genomic prediction: GP）を行ってみよう。

GP に用いられるモデル化手法では、入力変数（独立変数、説明変数とも呼ぶ）の欠測に対応できないものがほとんどである。そこで、欠測データを予め補完しておく。通常、マーカー遺伝子型データの欠測の補完には、Beagle などの専用のソフトウエアが用いられるが、ここでは、簡単に、マーカー遺伝子型スコア（-1, 0, 1 としてスコア化された遺伝子型データ）の平均値で置き換えることとする。もともと欠測率が 1%以下の SNP だけを用いているので、この簡単な補完方法でも問題はないと考えられる。

```
> # impute missing data in gt.score
> gt.score.imp <- gt.score
> for(i in 1:ncol(gt.score)) {
+     gt.score.imp[,i][is.na(gt.score[,i])] <- mean(gt.score[,i], na.rm = T)
+ }
```

次に、補完されたデータを、表現型データのある 388 系統と、表現型データが無い残りの 1180 系統に分けておく。最終的には、前者のデータから予測モデルを作成し、マーカー遺伝子型データをもとに後者の表現型を予測する。後述するように、こうした予測を行うことによって、表現型データの無い遺伝資源についても、有望な系統のスクリーニングを行うことが可能となる。

```
> # separate gt.score.imp into datasets with and without phenotypic data
> gt.score.imp.wp <- gt.score.imp[rownames(pheno),]
> gt.score.imp.wop <- gt.score.imp[!(rownames(gt.score.imp) %in% rownames(pheno)), ]
> dim(gt.score.imp.wop)
[1]  1180 38769
```

改めて主成分分析を行い、表現型データのある 388 系統と、表現型データが無い残りの 1180 系統間の遺伝的関係を調べておく。

```
> pca <- pca(gt.score.imp.wp, "ppca", nPcs = 6)
> pca.wop <- predict(pca, gt.score.imp.wop)
> plot(scores(pca)[,1:2], col = line$Sub.population)
> points(pca.wop$scores[,1:2], col = "gray")
> legend("bottomright", levels(line$Sub.population), col =
 1:nlevels(line$Sub.population), pch = 1)
> plot(scores(pca)[,3:4], col = line$Sub.population)
> points(pca.wop$scores[,3:4], col = "gray")
> plot(scores(pca)[,5:6], col = line$Sub.population)
> points(pca.wop$scores[,5:6], col = "gray")
```

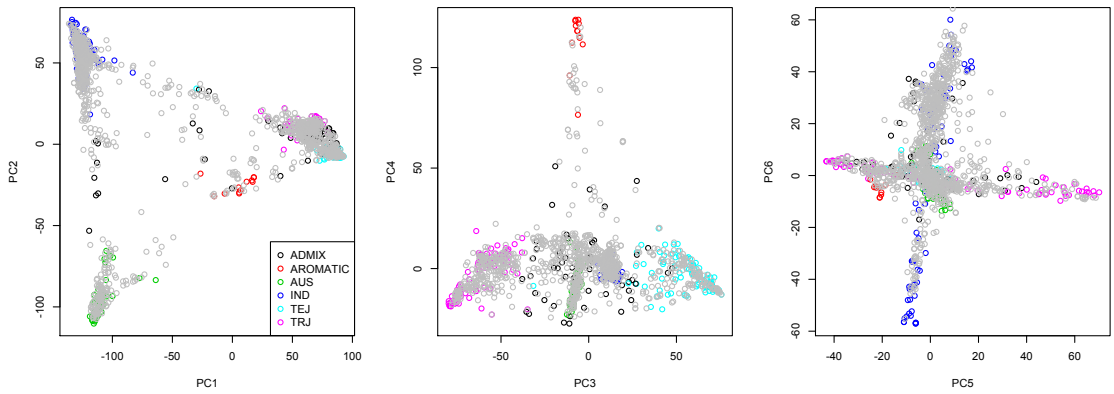

図 8. 主成分分析の結果。表現型データの無い系統は灰色のプロットで示してある。

主成分分析の結果をみると、表現型データのある 388 系統の分布と、表現型データが無い 1180 系統間の分布は互いに重なりあっており、前者と後者が同じような遺伝的背景をもった集団であることが分かる。

なお、GS は、GWAS では有意なマーカーが検出されないような形質において、特に有効である。ここでは、対象とする形質を、個体あたりの稔性のある頴花数として解析を行う。

```
> # prepare y again
> y <- pheno$Panicle.number.per.plant * pheno$Florets.per.panicle *
 pheno$Panicle.fertility # Fertile florets per plant
```

では、マーカー遺伝子型データを入力として、表現型値を予測する GP モデルを構築してみよう。GS のモデル化には、様々な手法が用いられるがここでは、リッジ回帰と LASSO という 2 つの正則化回帰分析を行ってみる。glmnet パッケージを用いれば、リッジ回帰、LASSO、ElasticNet などの正則化回帰モデルを構築できる。

まずは、リッジ回帰を行うための準備を行う。glmnet では、alpha というパラメータがあり、これを 0 に設定するとリッジ回帰、1 に設定すると LASSO、0〜1 の中間的な値に設定すると ElasticNet（「伸び縮みする網」という意味。網目の細かいリッジと粗い LASSO の間を臨機応変に設定できるので、このような名前がついたのだと思われる）を用いてモデル構築が行われる。

```
> # develop a prediction model with ridge regression
> alpha = 0  # 0 corresponds to ridge regression
```

次に、表現型データに欠測がある系統を除く。その際、yだけでなく、マーカー遺伝子型データについても同じ系統を除く必要がある。

```
> # select only samples with non-missing phenotype data
> selector <- !is.na(y)
> y <- y[selector]
> X <- gt.score.imp.wp[selector,]
```

さらに、多型の無いマーカーを除く。多型の有無のチェックには、分散の計算を用いている。分散が 0 でなければ多型があると判断できる。

```
> # select only polymorphic markers
> poly <- apply(X, 2, var) > 0
> X <- X[, poly]
>
> # extract also the chromosome id info of the selected markers
> X.chrom <- chrom[poly]
```

表現型データの無い 1180 系統のゲノムワイドマーカーについても、同じマーカーのセットに揃えておく。

```
> # select the same markers also for lines without phenotypic data
> X.wop <- gt.score.imp.wop[, poly]
```

glmnet パッケージの cv.glmnet 関数を用いてリッジ回帰のモデルを構築する。cv.glmnet 関数は、課すペナルティの大きさを調整するパラメータを、交差検証（cross-validation）によって最適化してくれる。関数名の最初の cv は交差検証を意味している。

```
> model.ridge <- cv.glmnet(X, y, alpha = alpha, standardize = F)
```

得られたモデルをもとに、マーカー遺伝子型データから表現型データを計算する。ただし、ここでは、モデル構築に用いた系統のマーカー遺伝子型データを入力にするため、モデルに基づく「予測」（prediction）ではなく、モデルの「あてはめ」（fitting）である。関数名 predict に惑わされないよう注意する。

```
> # predict y based on X (but actually this is not prediction)
> y.fit <- predict(model.ridge, newx = X, s = "lambda.min")
```

では、あてはめられたyと観察値のyについて比較をしてみよう。

```
> # compare fitted y with observed y
> plot(y, y.fit)
> abline(0, 1)
> cor(y, y.fit)
          1
[1,] 0.9335584
```

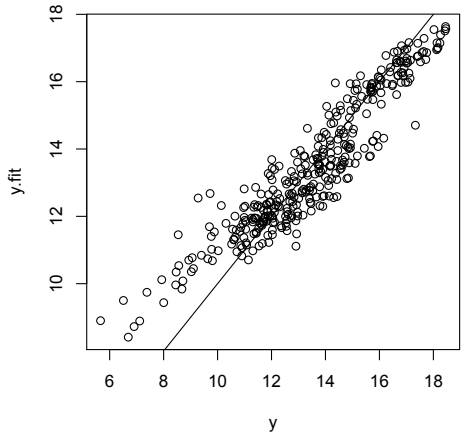

図 9. yの観察値（横軸）とあてはめ値（縦軸）の比較。相関も高く、関係は強い。

なお、入力される説明変数の数が多い場合には、あてはまりの良いモデルを構築するのは容易である。したがって、あてはまりの良さは、モデルの予測能力を反映しない。モデルの予測能力を評価するためには、モデル構築に用いなかったデータに対する「予測」を行い、その精度を評価する必要がある。そこで、本当はモデル構築に用いることができるデータの一部を、モデル構築から除いておき、除いておいたデータを予測して、その予測値を観察された値と比較する。除いておいたデータは、「本当はモデル構築に用いることができるデータ」であり、表現型データ（y）があるデータである。したがって、比較が可能である。

n-fold 交差検証とよばれる方法では、無作為にデータをnグループ分割し、そのうち 1 グループをモデル構築から除いておき、残りの$n-1$クラスで予測モデルを構築する。そして、除いておいた 1 グループについて、予測を行い、実測値との比較を行う。では、無作為に 1〜nの ID を系統にふってみよう。ここでは、$n=5$とする。無作為に ID を並べ替えるには sample 関数を用いる。

```
> # prepare ids (1 to n.fold) for all lines and sort them
> n.fold = 5
> id <- 1:length(y) %% n.fold + 1
> id
  [1] 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3
 [38] 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
 [75] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2
[112] 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4
[149] 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1
[186] 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3
[223] 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
[260] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2
[297] 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4
[334] 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2
> cv.id <- sample(id)
> cv.id
  [1] 2 1 4 2 2 3 4 4 4 3 1 3 2 5 2 1 1 2 4 3 2 2 2 3 3 3 2 5 2 5 4 1 3 2 2 4 1
 [38] 5 5 5 4 5 5 5 5 3 2 2 5 4 1 3 5 2 5 3 4 5 4 4 2 2 4 1 2 1 3 2 3 4 2 2 5 5
 [75] 4 5 3 3 3 4 3 5 1 4 1 4 1 2 1 5 5 1 3 4 4 4 1 5 1 3 5 3 5 1 2 4 1 2 5 1 5
[112] 5 2 1 5 1 4 4 2 2 5 1 3 4 5 1 2 4 5 1 4 4 3 4 4 4 4 1 2 5 4 2 4 1 4 4 3 3
[149] 3 2 4 3 3 2 3 2 5 5 3 4 1 2 1 4 4 3 5 3 2 1 4 1 3 5 1 2 1 4 2 1 5 1 4 4 2
[186] 3 4 1 4 3 5 3 5 4 3 4 4 1 1 1 3 3 3 5 4 5 1 3 2 1 3 3 1 2 1 4 2 2 1 2 4 3
[223] 4 3 4 5 3 2 3 3 4 4 3 2 5 4 4 2 2 5 3 3 5 3 1 2 3 2 1 1 4 5 3 1 2 1 2 5 5
[260] 5 2 2 3 1 2 5 5 5 1 3 5 1 1 2 1 3 4 5 1 1 1 5 5 5 3 5 4 5 4 5 4 1 5 3 5 1
[297] 3 2 2 2 2 5 2 5 1 3 1 1 3 3 1 2 1 3 1 2 3 3 4 2 1 2 1 5 3 5 2 2 5 5 2 4 3
[334] 1 1 4 2 2 4 5 5 5 4 3 3 1 2 4 5 3 3 4 1 4 1 2
```

では、ID が 1 番の系統を除いて、モデル構築用データを作成する。

```
> # closs validate
> y.train <- y[cv.id != 1]
> X.train <- X[cv.id != 1, ]
```

データを分割したことにより、多型がなくなるマーカーが出現する。そこでこれらを取り除く。

```
> # remove monomorphic markers
> poly <- apply(X.train, 2, var) > 0
> X.train <- X.train[, poly]
```

訓練用データでモデルを構築し、1 分割目の予測を行なう。

```
> # build a model
> model <- cv.glmnet(X.train, y.train, alpha = alpha, standardize = F)

> # predict y of the 1st fold
> y.pred <- predict(model, newx = X[cv.id == 1, poly], s = "lambda.min")
```

1 分割目について、観察値と予測値を比較する。

```
> # compare prediction with observation
> plot(y[cv.id == 1], y.pred)
> abline(0, 1)
> cor(y[cv.id == 1], y.pred)
```

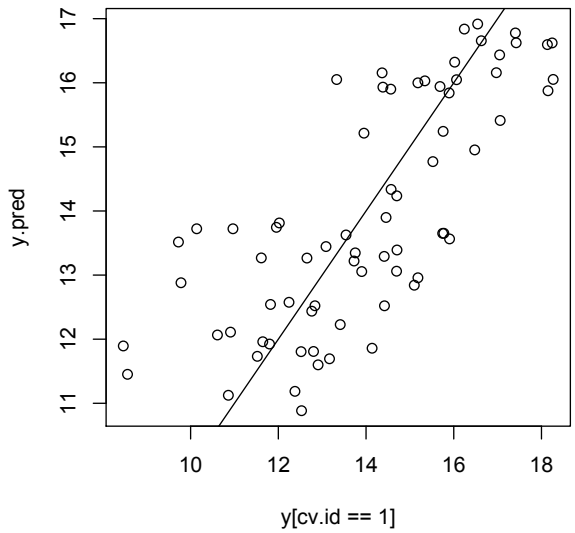

図 10. 1 分割目のyの観察値（横軸）と予測値（縦軸）の比較

実際には、各分割について、同様の計算を繰り返し、それらをまとめて精度評価する。そこで、以上の操作を、for 文を用いて 1〜5 の分割について繰り返す。

```
> # repeat this analysis for #1 - #5 folds
> y.pred <- rep(NA, length(y))
> for(i in 1:n.fold) {
+     print(i)
+     y.train <- y[id != i]
+     X.train <- X[id != i, ]
+
+     poly <- apply(X.train, 2, var) > 0
+     X.train <- X.train[, poly]
+
+     model <- cv.glmnet(X.train, y.train, alpha = alpha, standardize = F)
+     y.pred[id == i] <- predict(model, newx = X[id == i, poly], s = "lambda.min")
+ }
```

予測値と観察値を比較。相関係数も計算する。

```
> # comparison between prediction and observation
> plot(y, y.pred)
> abline(0, 1)
> cor(y, y.pred)
[1] 0.7431501
```

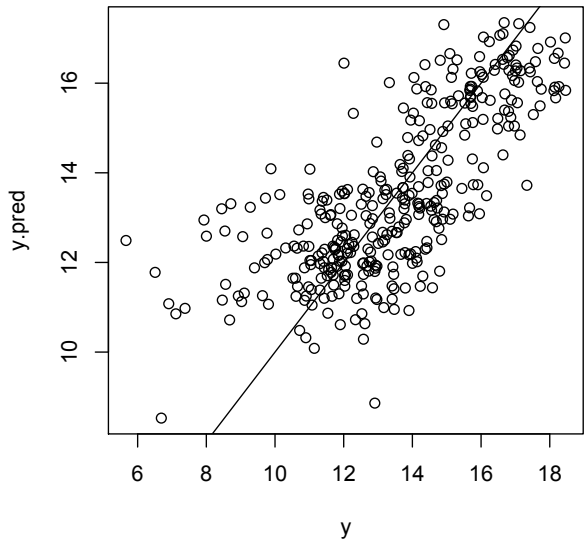

図 10. 1〜5 分割を合わせた場合のyの観察値（横軸）と予測値（縦軸）の比較

マーカー効果を図示してみる。全てのデータを用いてモデルを構築し、それをもとにマーカー効果を図示する。マーカー効果は、coef 関数で取り出せる。

```
> # estimate and draw marker effects
> model.ridge <- cv.glmnet(X, y, alpha = alpha, standardize = F)  # use all data
> beta <- coef(model.ridge, s = "lambda.min")[-1]
> plot(abs(beta), col = X.chrom, main = "Ridge", ylab = "Coefficients", xlab = "Position
 (bp)")
```

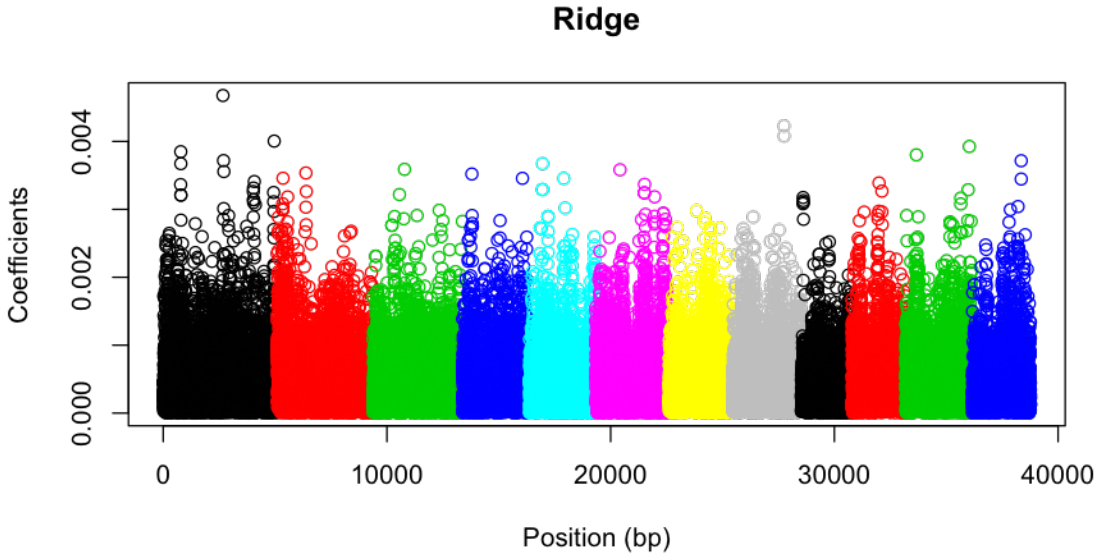

図 11. リッジ回帰において推定された各 SNP の効果の大きさ（絶対値）。

推定効果をプロットしてみると、特に目立って効果の大きな SNPs があるわけではなく、多くの SNP がある程度の遺伝効果を持っていることが分かる。これは、GWAS の際にも示唆されたように、特に効果の大きな遺伝子が存在しないことによると推察される。こうした形質で、観察値と予測値の相関が 0.74 程度であることは重要である。なぜなら、例えば、少数の遺伝子に支配される形質は従来のマーカー利用選抜（marker assisted selection: MAS）で選抜でき、GS の利用は必ずしも必要ではない。しかし、GS では、特に目立った効果を示す遺伝子が存在しない（あるいは、見つからない）形質でも、ゲノムワイドマーカー多型をもとに選抜可能である。収量や品質などの経済的に重要な農業関連形質の多くは、多数の遺伝子に支配される所謂「複雑形質（complex traits）であり、GS 利用の意義が大きい。

最後に、表現型データがある 388 系統から得られた予測モデルを、表現型データの無い 1180 系統に適用して、後者の表現型値を予測する。

```
> # prediction of phenotypes of lines without phenotypic data
> y.pred.wop <- predict(model.ridge, newx = X.wop, s = "lambda.min")
> conf <- hist(c(y.pred,y.pred.wop), col = "#ff00ff40", border = "#ff00ff")
> hist(y.pred.wop, col = "#0000ff40", border = "#0000ff", breaks = conf$breaks, add = T)
> sum(y.pred.wop>16)
[1] 77
```

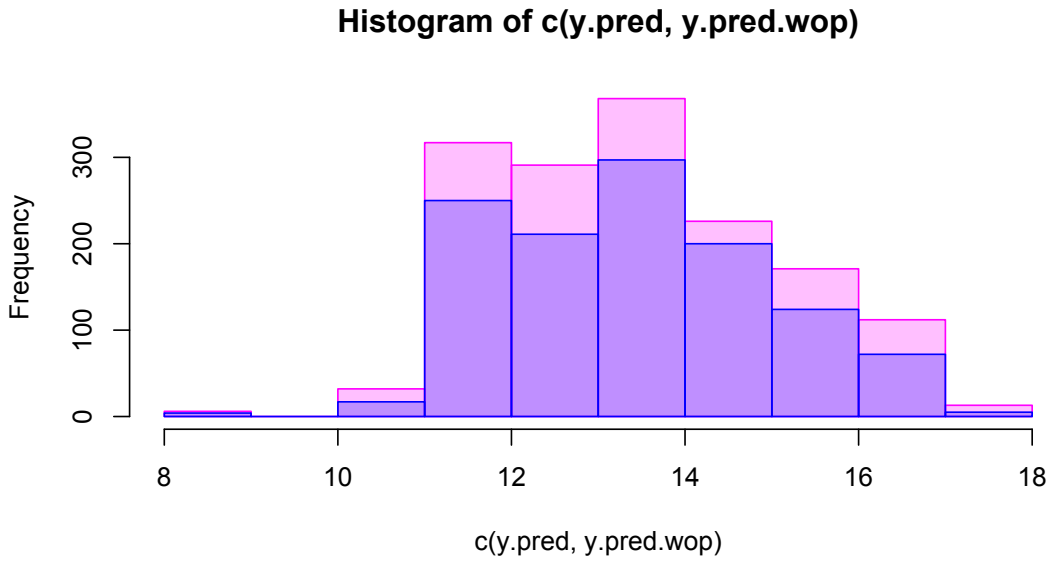

図 12. モデル作成に用いられた表現型データをもつ 388 系統の予測値と、表現型データの無い 1180 系統の予測値のヒストグラム。個体あたりの稔性穎花数が多いほうが良い場合に、例えば、この値が 16 を超える系統が 1180 系統中 77 系統近くあることが分かる。

最近、多数の遺伝資源についてゲノムデータが解析・公開されるようになってきている。例えば、イネでは遺伝資源 3,000 系統について全ゲノム配列のリシークエンスが行われ、そのデータが既に公開されている（The 3,000 Rice Genome Project 2014, Gigasicence 3:7; Sun et al. 2017, Nucl. Aci. Res. 45:597-605）。また、今回例として用いているデータも 1,500 系統以上について 70 万 SNPs の遺伝子型データが解析され、公開されている。こうしたデータを活用することにより、表現型データのスクリーニングが難しいために、死蔵されることが多かった遺伝資源を簡易にスクリーニングできるようになる。したがって、GS を用いることで、有用な遺伝資源を迅速に発見し、育種に効率的に活用する道が開ける（Yu et al. 2016, Nature Plants 2: 16150; Tanaka and Iwata 2017, Theor. Appl. Genet. 131:93-105）。

最後に、LASSO を適用する場合には、alpha を 1 にして cv.glment での解析を行なう。なお、alpha を 1 にする以外は、上で行った計算と全く同じである。

```
> alpha = 1  # do the same thing with alpha = 1
>
> y.pred <- rep(NA, length(y))
> for(i in 1:n.fold) {
+     print(i)
+     y.train <- y[id != i]
+     X.train <- X[id != i, ]
+
+     poly <- apply(X.train, 2, var) > 0
+     X.train <- X.train[, poly]
+
+     model <- cv.glmnet(X.train, y.train, alpha = alpha, standardize = F)
+     y.pred[id == i] <- predict(model, newx = X[id == i, poly], s = "lambda.min")
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
> plot(y, y.pred)
> abline(0,1)
> cor(y, y.pred)
[1] 0.719464
```

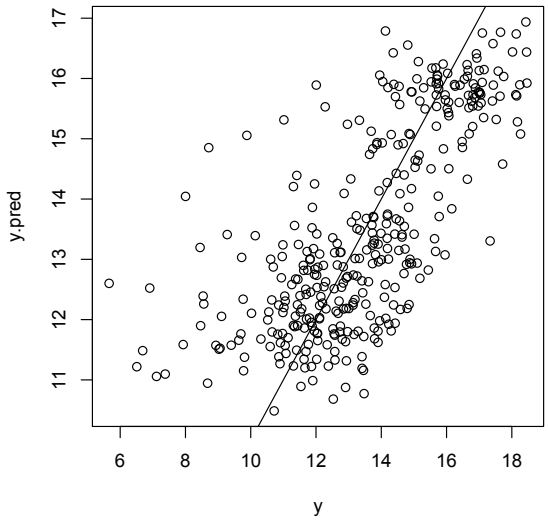

図 13. LASSO を GP に用いた場合の 5 分割交差検証の結果

推定された効果（の絶対値）を図示する。

```
> # estimate and draw marker effects
> model.lasso <- cv.glmnet(X, y, alpha = alpha, standardize = F)
> beta <- coef(model.lasso, s = "lambda.min")[-1]
> plot(abs(beta), col =X.chrom, main = "LASSO", ylab = "Coefficients", xlab = "Position
 (bp)")
```

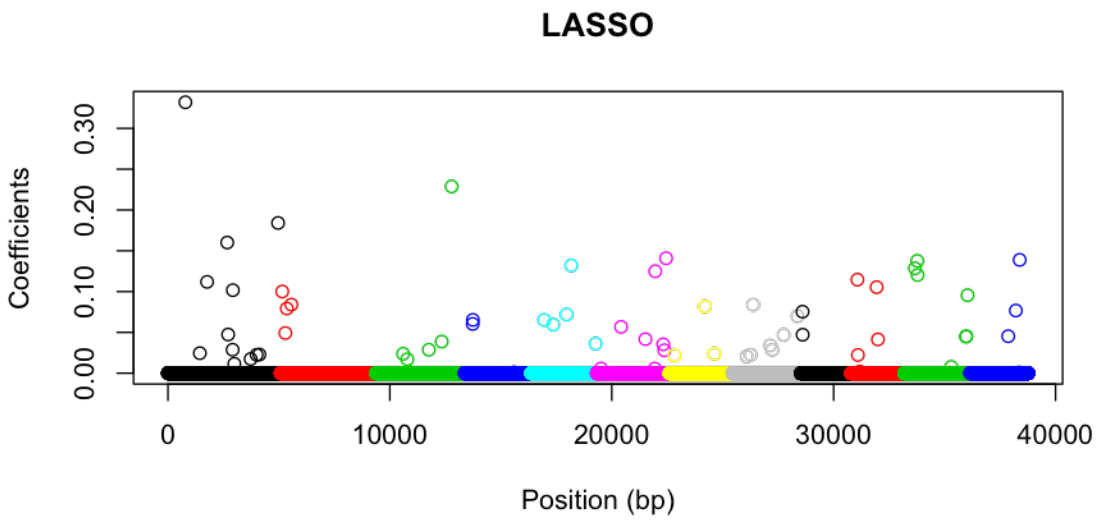

図 14. LASSO において推定された各 SNP の効果の大きさ（絶対値）。

リッジ回帰に比べ、効果をもつ SNPs の数が少ないことが分かる。また、効果をもつ場合は、リッジ回帰に比べ、効果の大きさが大きいことも分かる。しかし、LASSO を用いた場合でも、複数の SNPs に重み付けされており、この形質に関わる遺伝子の数が少なくないことが示唆される。

最後に、表現型データが無い 1180 系統の予測値を計算する。

```
> # prediction of phenotypes of lines without phenotypic data
> y.pred.wop <- predict(model.lasso, newx = X.wop, s = "lambda.min")
> conf <- hist(c(y.pred,y.pred.wop), col = "#ff00ff40", border = "#ff00ff")
> hist(y.pred.wop, col = "#0000ff40", border = "#0000ff", breaks = conf$breaks, add = T)
> sum(y.pred.wop>16)
[1] 54
```

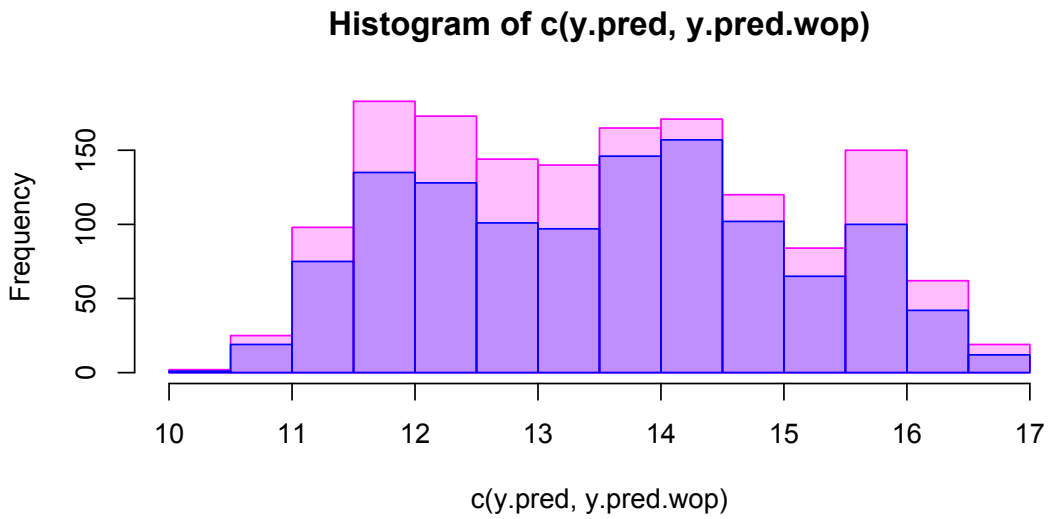

図 15. モデル作成に用いられた表現型データをもつ 388 系統の予測値と、表現型データの無い 1180 系統の予測値のヒストグラム。個体あたりの稔性頴花数が多いほうが良い場合に、例えば、この値が 16 を超える系統が 1180 系統中 54 系統近くあることが分かる。

用いたモデル化手法とモデルが異なるため、リッジ回帰の場合と少し異なる結果が得られる。実際に適用する場合には、複数の手法の中から、それぞれの形質に合った手法を経験的に選んで用いるのが良い。どの形質にも万能な手法は無い（データ科学の世界では、これを「No free lunch 定理」とよぶ）。実際の GS にはここでは紹介しなかった手法が多数用いられる。その中でも特に混合モデルと Bayes 回帰が重要であるが、時間の都合上解説はしない。

以上で R を用いた GWAS と GS 予測モデル構築の方法に関する説明は終わりである。ここでは種子の縦横比と個体あたりの稔性のある頴花数の 2 形質を用いて解析を行ったが、他の形質についても解析を行ってみるといろいろな発見があるだろう。