# CKB SNP Clustering plots

Kuang Lin [kuang.lin@ndph.ox.ac.uk](mailto:kuang.lin@ndph.ox.ac.uk) 30 June 2016

Given batches of SNP callings, we performed (logistic) regressions to detect SNPs with potential batch and plate effects.

Some manual (visual) checking of the clustering plots is necessary, for deciding the thresholds for QC and the rejection of SNPs with data of poor quality and/or bad calling.

We need to be able to generate the clustering plots from the genotype calling data.

# Linux command alias:

alias h='head'

alias t='tail'

alias skh='tail -n +2 '

# Go to a directory on the NC2 computer (nc2.ndph.ox.ac.uk).

cd /kuser/kuangl/dev/snp\_clustering\_plot

################################################################################

# 1. Input files #

################################################################################

indir=/kuser/shared/data/GWAS\_backup/

################################################################################

# 1.1 SNP lists:

# SNPS with plate effects

h $indir/full\_data/plate-effect/variant\_plate\_effects\_v2.txt

# 4 columns

# batch plate probeset p-value

# 'p-value' is missing in the header

# 33621 entries, 30570 unique SNPs

# SNPs with batch effects

h $indir/full\_data/batch\_test/variant\_batch\_effects.txt

# 3 columns

# batch probeset P-val

# 6407 entries, 4048 unique SNPs

# SNPs with batch effects found in the non-relatives dataset

h $indir/full\_data/batch\_test/variant\_batch\_norel\_effects.txt

# 3 columns

# batch probeset P-val

# 4154 entries, 2876 unique SNPs

################################################################################

# 1.2 calling files:

# 7 batches of calling files at

$indir/plates1-53/

$indir/plates54-105/

$indir/plates106-156/

$indir/plates157-209/

$indir/plates210-261/

$indir/plates262-318/

$indir/plates319-367/

# in a batch, we need the four files:

AxiomGT1.calls.txt

AxiomGT1.confidences.txt

AxiomGT1.snp-posteriors.txt

AxiomGT1.summary.txt

# and the plate cel file lists (needed for plate highlighting)

$indir/plates1-53.txt

$indir/plates54-105.txt

$indir/plates106-156.txt

$indir/plates157-209.txt

$indir/plates210-261.txt

$indir/plates262-318.txt

$indir/plates319-367.txt

################################################################################

# 2. SNP cluster plots #

################################################################################

awk '{print $3}' $indir/full\_data/plate-effect/\

variant\_plate\_effects\_v2.txt > t.ls

awk '{print $2}' $indir/full\_data/batch\_test/\

variant\_batch\_effects.txt >> t.ls

awk '{print $2}' $indir/full\_data/batch\_test/\

variant\_batch\_norel\_effects.txt >> t.ls

sort t.ls | uniq | grep -v probeset > snp.ls

# The 34394 unique SNPs in the file ‘snp.ls’ are to be plotted.

################################################################################

# 2.1 SNPolisher SNP classification

Install the R package SNPolisher from http://www.affymetrix.com/estore/partners\_programs/programs/developer/\

tools/devnettools.affx

Note that the SNP classification process is not too slow. We applied it on ALL SNPs.

awk '{print $2'} $indir/full\_data/\*stage1.bim | \

sort | uniq > full\_snp.ls

The 687236 SNPs in the file ‘full\_snp.ls’ are to be classified.

mkdir b01 b02 b03 b04 b05 b06 b07

The R script SNP\_classify.R

1. calculates SNP clustering metrics for all SNPs in 'full\_snp.ls';

2. classifies them into 7 categories;

3. grabs the calls, confs, posterior and summary sub-tables for the listed SNPs using a perl script included in the SNPolisher package;

4. plots the clustering, if the last two sections were uncommented.

Let it run over night on NC2:

nohup SNP\_classify.R b01 plates1-53/ &

nohup SNP\_classify.R b02 plates54-105/ &

nohup SNP\_classify.R b03 plates106-156/ &

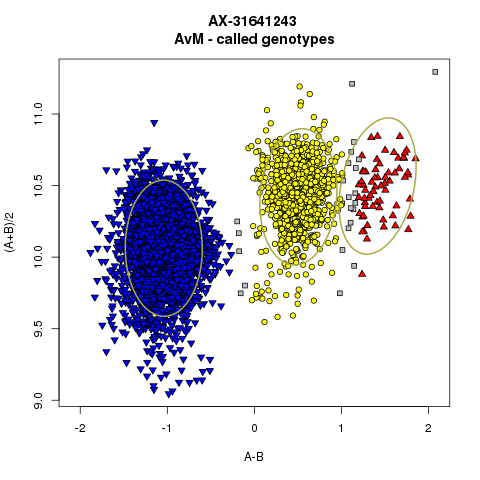
nohup SNP\_classify.R b04 plates157-209/ &

nohup SNP\_classify.R b05 plates210-261/ &

nohup SNP\_classify.R b06 plates262-318/ &

nohup SNP\_classify.R b07 plates319-367/ &

It takes about 8~9 hours to classify all SNP callings of a batch, and get the 'metrics.txt' and 'Ps.performance.txt' files. Then it can grab sub-tables for 4000 SNPs in one hour, and plot 4000 SNPs in two hours.



################################################################################

# 2.2 SNP calling classes

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

skh $batch/Ps.performance.txt | awk '{print $16}' | sort | uniq -c

done

# b01 b02 b03 b04 b05 b06 b07

# PolyHighResolution 521250 519578 518984 513339 516908 516373 515777

# NoMinorHom 104752 107367 109141 110955 102307 101447 102348

# MonoHighResolution 70838 66297 65634 67220 69333 76164 74700

# Hemizygous 1227 1227 1227 1227 1227 1227 1227

# OTV 3437 4393 4279 3947 4218 4004 3901

# Other 77710 80636 79786 82277 85380 79715 81393

# CallRateBelowThreshold 2723 2439 2886 2972 2564 3007 2591

# PolyHighResolution 66.2 +- 0.34 %

# NoMinorHom 13.5 +- 0.48 %

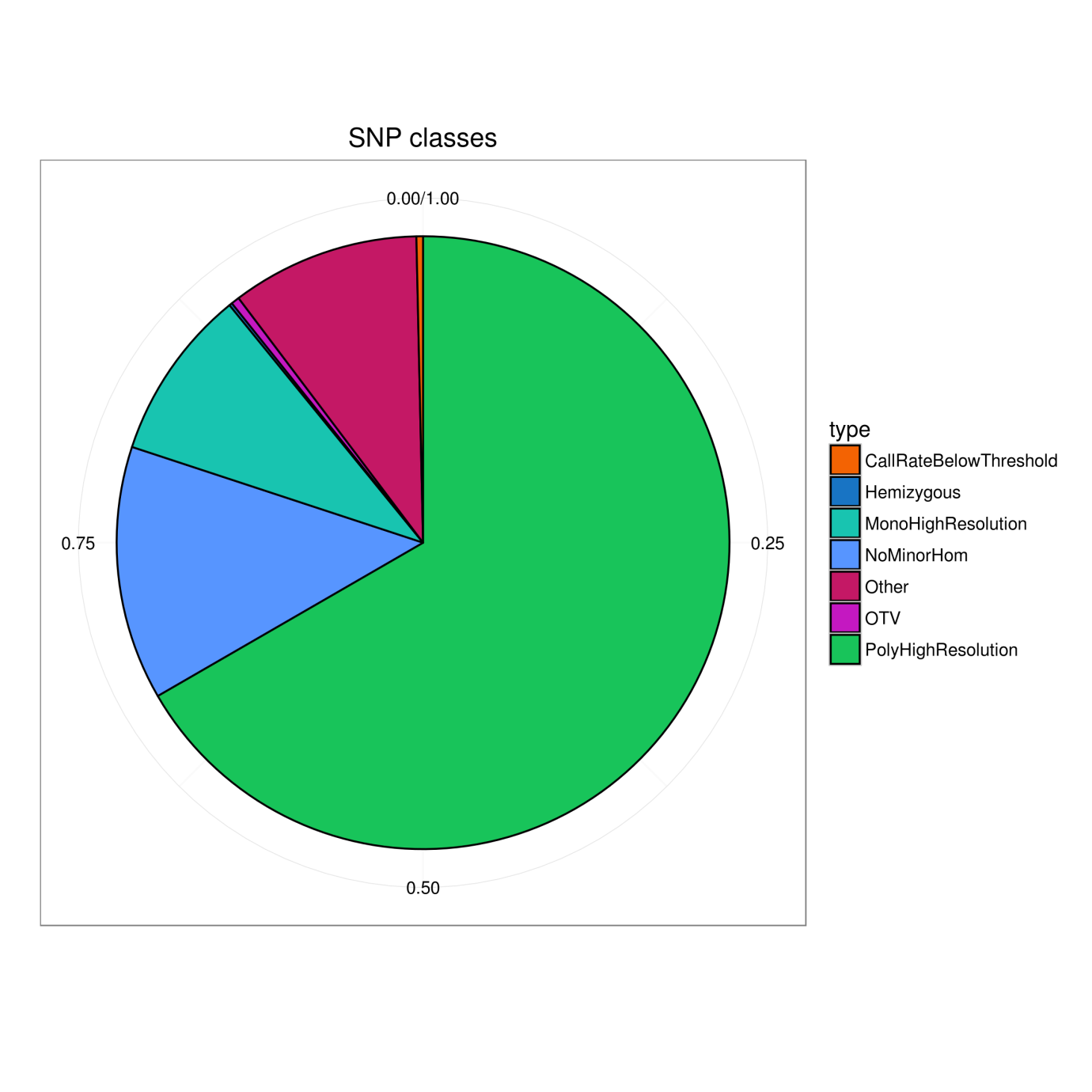
# MonoHighResolution 9.0 +- 0.53 %

# Hemizygous 0.2 +- 0 %

# OTV 0.5 +- 0.04 %

# Other 10.4 +- 0.31 %

# CallRateBelowThreshold 0.4 +- 0.03 %



The pie plot was created using

SNP\_class\_pie.R

################################################################################

# 2.3 calling class square plots

For each SNP, in addition to its clustering plots, we also want a square plot to show the calling classes across the 7 batches.

mkdir class\_png

# To combine SNPs classifications across all 7 batches into the file 't.in'

awk '{print $1,$16}' b01/Ps.performance.txt > t.out

grab -f snp.ls t.out > t.in

for batch in b02 b03 b04 b05 b06 b07 ; do

awk '{print $1,$16}' $batch/Ps.performance.txt > t.out

grab -f snp.ls t.out > t2.in

paste t.in t2.in > t.out

# check SNP ids match

awk '{if ($1 != $(NF-1) ) print "WHAT?",$0}' t.out

awk '{print $2}' t2.in > t3.in

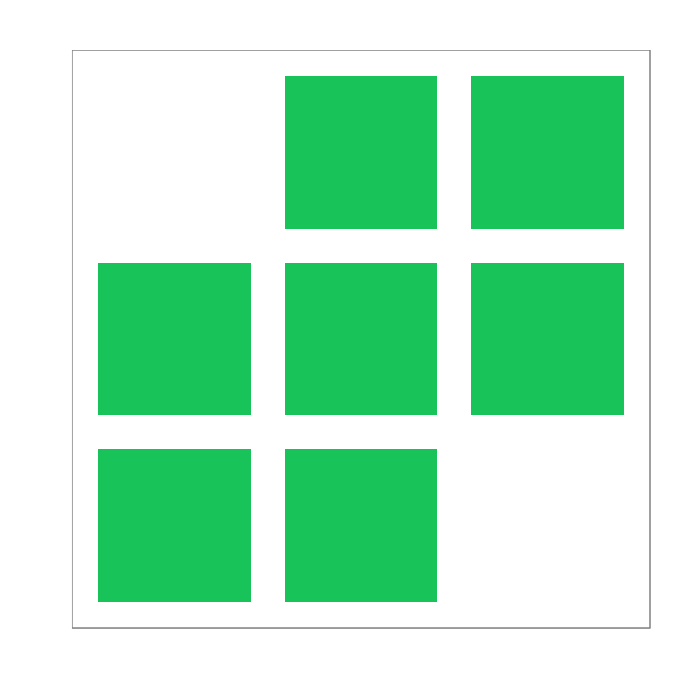
paste t.in t3.in > t2.in

mv t2.in t.in

done

nohup SNP\_class\_squares.R &

The script uses the same colouring scheme as in the pie plot. It generates more than 200 SNP class plots per minute, 300,000 per day. It is considered fast enough.



################################################################################

# 2.4 re-visit the plotting problem

We need SNP clustering plots. But the R script SNP\_classify.R, which uses the SNPolisher library, is too slow (about 1 plot per min). And we need to modify the plots so it's easier to check them visually.

###################

# To double-check the orders of chips are the same between the call and summary files

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

echo $batch

head -n 1 $batch/calls.txt > t.ls

transpos\_file t.ls > t1.ls

head -n 1 $batch/summary.txt > t.ls

transpos\_file t.ls > t2.ls

diff t1.ls t2.ls | wc

done

# No inconsistency was found.

###################

# To double-check the summary files have the order of A and B of the same SNP

# e.g. 'AX-100002645-A' followed by 'AX-100002645-B'

# then 'AX-100002667-A' followed by 'AX-100002667-B'

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

echo $batch

awk '{print $1}' $batch/summary.txt | tail -n +2 > t.ls

awk -F"-" '{print $2}' t.ls | uniq -c | awk '{print $1}' | uniq -c

# all 2

awk -F"-" '{print $3}' t.ls | sort | uniq -c

# 'A' or 'B', nothing else

awk -F"-" '{print $3}' t.ls | uniq -c | awk '{print $1}' | uniq -c

# all 1, 'A' and 'B' are never consecutive

done

#######################################

# 2.4.1 To get the subsets from the posterior files

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

get\_posterior.py $batch > ${batch}.posterior

done

# very quick

#######################################

# 2.4.2 To generate avm files

The avm (A\_vs\_M) files should have the a and b signals, in the form of A <- (log2(a) + log2(b))/2 and M <- log2(a) - log2(b). The SNP callings are also included into an avm file. Originally they are '0', '1', '2' or '-1' for missing. Here we change '-1' to '3' for missing.

This script uses the’ calls.txt’ and ‘summary.txt’ files in the batch directories. It reads the SNP ids from the file 'snp.ls'.

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

nohup get\_avm.py $batch &

done

A job produces 500 SNP avm files per min. We finished making ALL 687236 x 7 avm files for ALL SNPS in 17 hours.

However, DO NOT DO IT (again).

Having 4.8 million extra files excessively deteriorates the performance of NC2. Make sure the 'snp.ls' is not too long (>100k SNP ids), unless a parallelized hard-drive cluster which can easily handle millions of files becomes available.

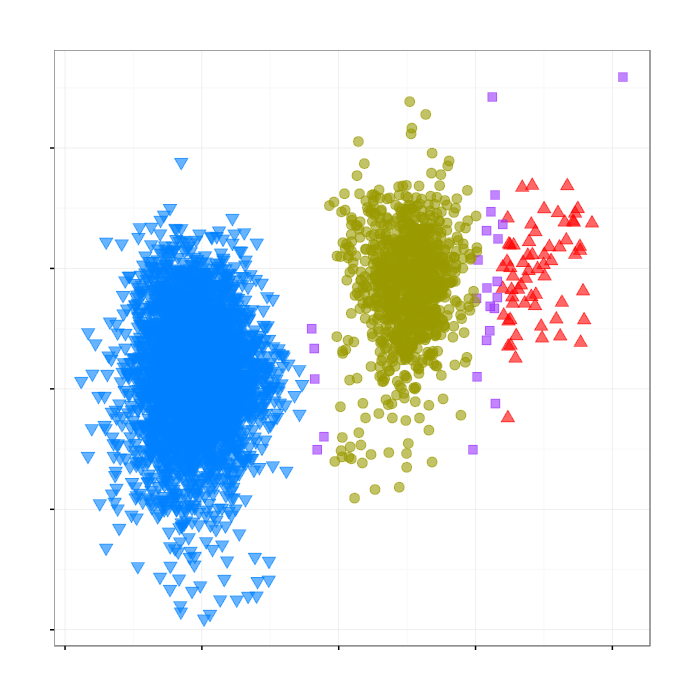
#######################################

# 2.4.3 to plot the A\_vs\_M files

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

nohup SNP\_cluster\_plot.R $batch &

done

The script normally produces 70 pictures per minute per job on smaller SNP lists. It is about 60 times faster than the script using the SNPolisher functions. 

################################################################################

# 2.5 To combine the square plot and the clustering plots.

IFS=$'\n' snps=($(cat snp.ls))

for snp in ${snps[@]} ; do

echo $snp

convert class\_png/$snp.png b01/$snp.png b02/$snp.png +append r1.png

convert b03/$snp.png b04/$snp.png b05/$snp.png +append r2.png

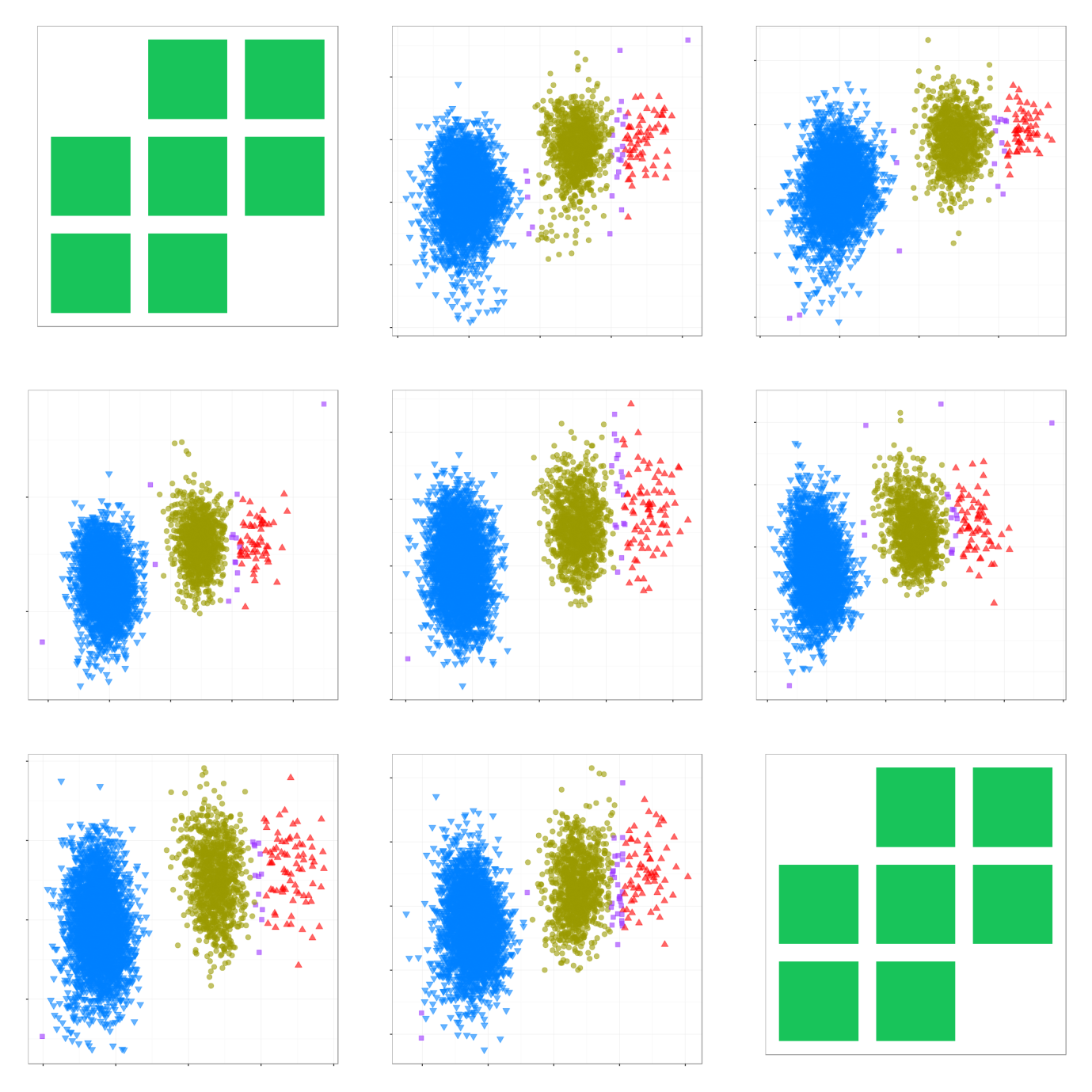
convert b06/$snp.png b07/$snp.png class\_png/$snp.png +append r3.png

convert r1.png r2.png r3.png -append ${snp}\_comb.png

done

mkdir to\_exam\_png.bak/

mv \*comb.png to\_exam\_png.bak/



The tgz of the directory 'to\_exam\_png.bak/' is also available as ‘/kuser/kuangl/backup/snp\_clust\_plot\_manual\_exam\_png.tgz' on NC2.

################################################################################

# 3. plate effect plotting #

################################################################################

We modified the script get\_avm.py, so the problematic plate will be highlighted.

for batch in b01 b02 b03 b04 b05 b06 b07 ; do

nohup get\_highlight\_avm.py $batch &

done

33620 avm files are to be made. All are done in 45 minutes. It creates files like 'b01/AX-64101281\_NOR14120204.avm'.

ls b0?/\*avm > t.ls

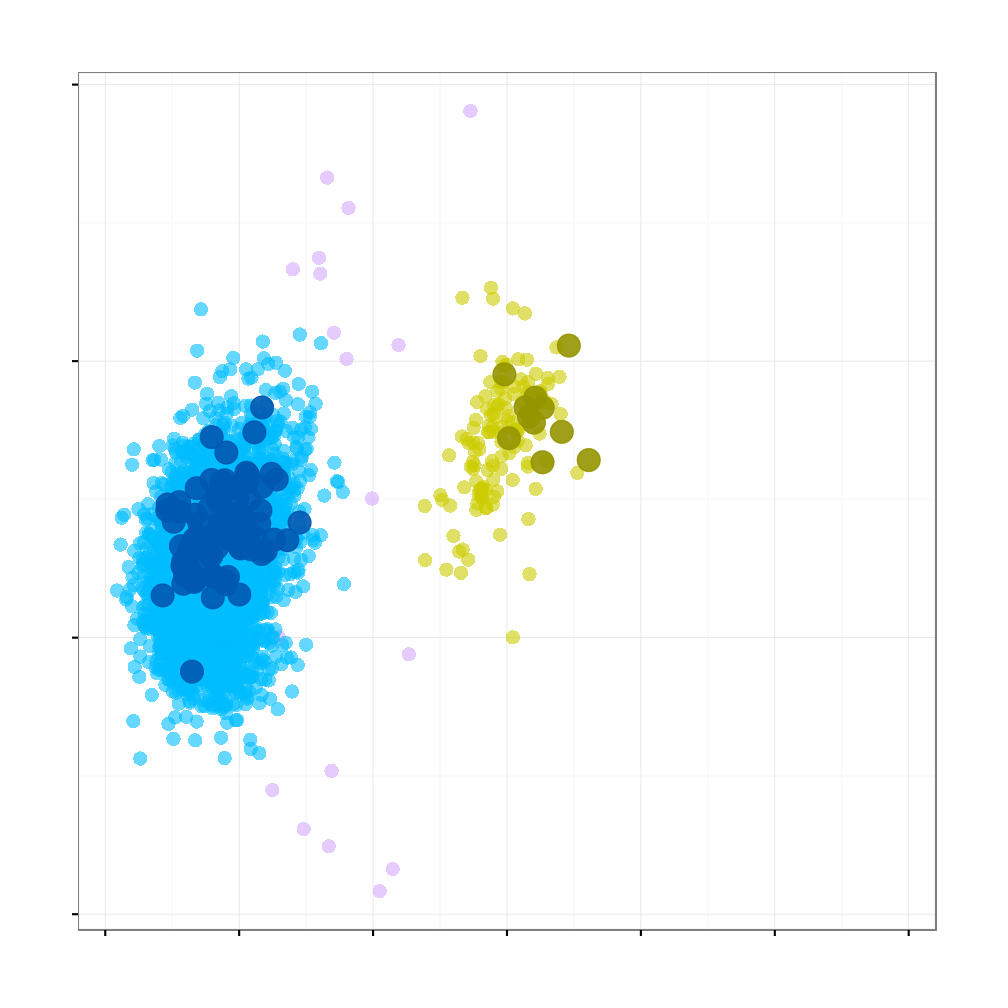
split -l 1682 t.ls

We simplified the script SNP\_cluster\_plot.R to highlight the callings from specific (problematic) plates.

for ifile in x?? ; do

nohup SNP\_highlight\_cluster\_plot.R $ifile &

done



It is much faster. No posterior eclipses are to be plotted this time. About 100 png files are created per minute per job. We finished all plotting in 22 minutes.

################################################################################

# 4. Summary #

################################################################################

To obtain some clustering plots

1. Creat the list file ‘snp.ls’
2. SNP\_classify.R (not needed if classification was done for all SNPs)
3. get\_posterior.py
4. get\_avm.py
5. SNP\_cluster\_plot.R
6. SNP\_class\_squares.R (optional)
7. Combine the plots (optional)

Or, to obtain clustering plots with plates highlighted

1. Creat the list file ‘snp.ls’
2. SNP\_classify.R (not needed if classification was done for all SNPs)
3. get\_highlight\_avm.py
4. SNP\_highlight\_cluster\_plot.R

The scripts are available at K:\kadoorie\GWAS\_data\phase12\_snp\_clustering\_plot.

################################################################################

# The END #

################################################################################