Association

# Model (Current Scripts)

## Association

Covariates

* Pre-computed first 10 PC
* Sex
* Birth year
* Genotype measurement wells

Parameters

* White British



## Get Phenotypes and Covariates

|  |  |  |
| --- | --- | --- |
| **Description** | **Field ID** | **Column Number** |
| Standing Height | 50-1.0 | 82 |
| 40 PC | 22009 | 10004 – 10043 |
| Ethnicity | 21000 | 9786 |
| Sex | 22003 | 9997 |
| Birth Year | 34 | 25 |
| Genotype measurement batch | 22000 | 9996 |
| Genotype measurement plate | 22007 | 10002 |
| Genotype measurement well | 22008 | 10003 |

\*6/1/2021 data columns in phenotype file are shifted to the right by 1



* IDs column



* Get column number based on Field ID



* Extract desired fields using column number



* Keep only white British from ethnicity data field
  + White British == 1001



* Concatenate covariates



* Remove rows with missing values



## Combine Results



## Scatter Plot and Correlation

* Plot the negative log10 p-values of Neale Lab and plink2 results to examine for errors
* Plot betas of Neale Lab and plink2 results
* Obtain correlation between Neale and plink2 p-value results





## Manhattan Plot

Code adapted from:

<https://piperwrites95180714.wordpress.com/2018/04/04/genome-wide-association-study-manhattan-plot-tutorial/>

<https://stackoverflow.com/questions/37463184/how-to-create-a-manhattan-plot-with-matplotlib-in-python>

* Manhattan plots show the neg log 10 p-values on a genomic scale. The skyscrapers represent clusters of SNPs in strong LD with each other. X axis is ordered by chromosome and chromosome position.

#!/usr/bin/env python3

# import packages

import matplotlib

matplotlib.use("Agg")

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import sys

import os

def manhattan(pheno="phenotype", sex="both\_sex", y\_max = 0):

    #set variables

    file\_name = "{1}\_all.{0}.glm.linear".format(pheno, sex)

    plot\_name = "manhattan\_{0}\_{1}.png".format(pheno, sex)

    plot\_title = "Manhattan Plot of {0} : {1}".format(pheno.capitalize(), sex.capitalize)

    # DATAFRAME

    # load file to dataframe

    results\_df = pd.read\_csv(file\_name, sep="\t", usecols =['#CHROM','POS','P','ID'], dtype= {'#CHROM':np.int64,'POS':np.int64,'P':np.float64,'ID':str})

    # drop rows with any column having null/missing data ; should already be removed when combine results

    results\_df = results\_df.dropna()

    # sort by column then position; reset index

    results\_df = results\_df.sort\_values(['#CHROM', 'POS'])

    results\_df.reset\_index(inplace=True, drop=True)

    # change P column to float type

    results\_df['P'] = pd.to\_numeric(results\_df['P'])

    # create column with negative log p value

    print(results\_df[results\_df['P'] == 0])

    results\_df = results\_df[(results\_df['P'] != 0)]

    results\_df['NEG\_LOG\_P'] = -np.log10(results\_df['P'])

    # make chromosome column into type category

    results\_df['#CHROM'] = results\_df['#CHROM'].astype('category')

    # index; used for x axis; assume uniform SNP distrubtion across chromosome

    results\_df['index'] = range(len(results\_df))

    # group by chromosome

    grouped\_df = results\_df.groupby(('#CHROM'))

    #PLOT

    fig = plt.figure(figsize=(18,14))

    # axes of figure - 1row,1col,1idx

    ax = fig.add\_subplot(111)

    colors = ['#466EA6','#7251B8']

    x\_labels = []

    x\_labels\_pos = []

    # create subplots for each chromosome (name = #CHROM)

    for num, (name, group) in enumerate(grouped\_df):

        ##### plot, x is index and y is neg log p ######

        group.plot(kind='scatter', x='index', y='NEG\_LOG\_P',color=colors[num % len(colors)], ax=ax, s=5, marker = '.')

        # name of chr

        x\_labels.append(name)

        # tick marks; middle of group

        x\_labels\_pos.append((group['index'].iloc[-1] - (group['index'].iloc[-1] - group['index'].iloc[0])/2))

    #line

    ax.plot([0,len(results\_df)],[5,5])

    # figure labels

    ax.set\_xticks(x\_labels\_pos)

    ax.set\_xticklabels(x\_labels)

    ax.set\_xlim([0, len(results\_df)])

    ax.set\_xlabel('Chromosome')

    ax.set\_title(plot\_title)

    #y-axis, keep as max of both\_sex

    if sex == "both\_sex":

        y\_max = results\_df['NEG\_LOG\_P'].max()

        print(y\_max)

        ax.set\_ylim([0, y\_max])

        plt.tight\_layout()

        plt.savefig(plot\_name)

        return y\_max

    else:

        ax.set\_ylim([0, y\_max])

        print(results\_df['NEG\_LOG\_P'].max())

        # save as png

        plt.tight\_layout()

        plt.savefig(plot\_name)

# set working directory

os.chdir("/scratch1/08005/cz5959/GWAS\_Results")

# sys.argv[1] should be phenotype name

y\_max = manhattan(str(sys.argv[1]),"both\_sex")

manhattan(str(sys.argv[1]),"female", y\_max)

manhattan(str(sys.argv[1]),"male", y\_max)

## QQ Plot

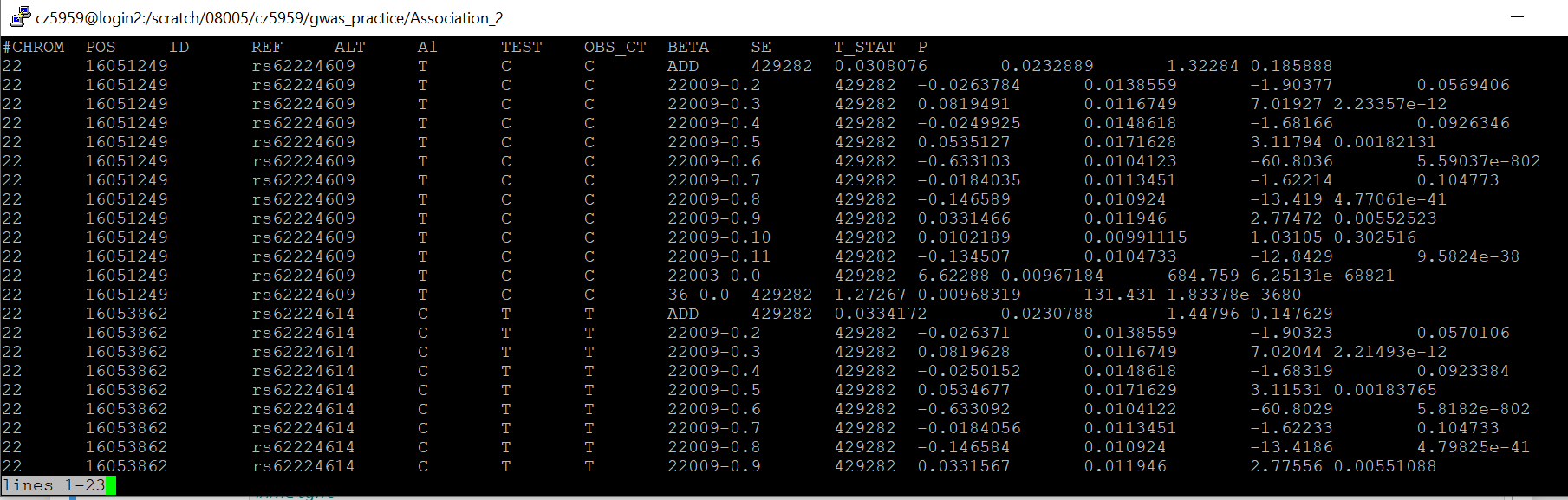
* A QQ plot shows the deviation of observed p-values from the null hypothesis. In the null hypothesis, the SNPs are not associated with the trait, and follow a uniform distribution. The observed p values are ordered and plotted against the expected.
* If the observed deviates towards y-axis too early, it means that many significant p values are more significant than expected under the null, which is rare.



# Log

### 5/24/2021

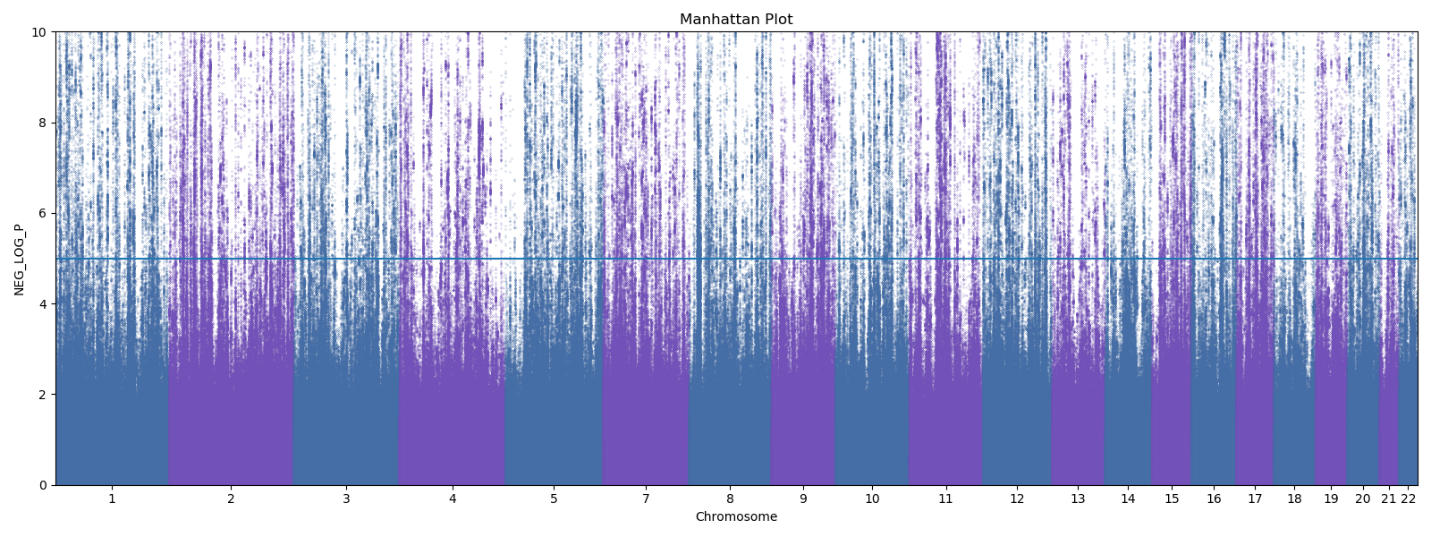
* Why do I have multiple p-values for the same SNP in my gwas results?



* + Plink2 default output file contains a line for each genotype column and each non-intercept covariate column
  + If not using information from covariate column, use ‘hide-covar’ modifier for –glm

### 5/25/2021

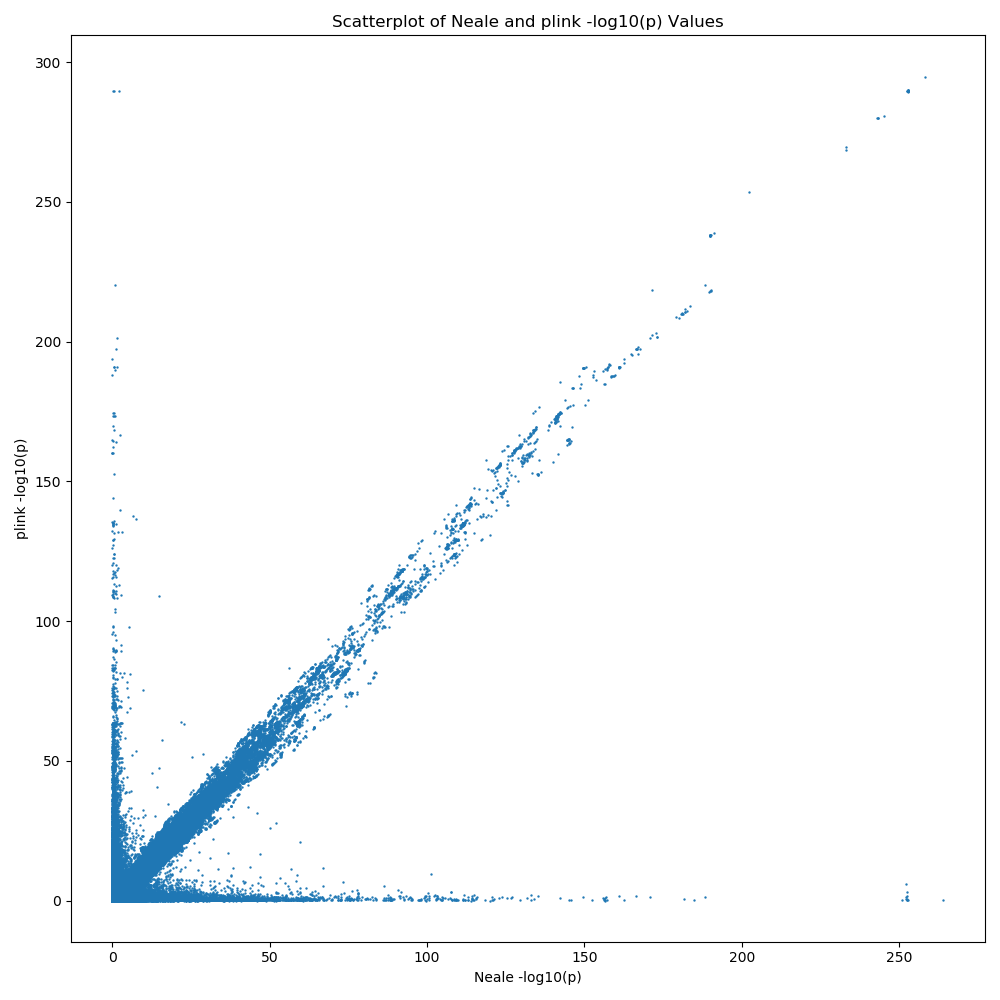
* Transfer files from ls5 to frontera
* Redo height gwas with ‘hide-covar’ modifier and combine the results
* Manhattan plot for height after hiding covariates



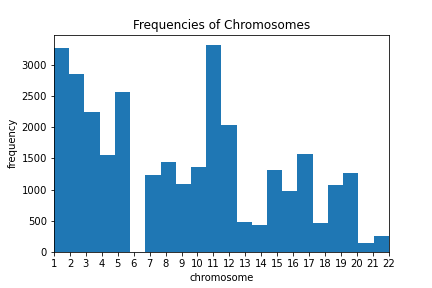
* Correlation between Neale and plink2 gwas results for height
  + Order of Correlations  
    plink and Neale: 0.708 ; rows = 132740  
    plink and own: 0.443 ; rows = 869  
    Neale and own: 0.319 ; rows = 860
  + I need more of my own P values to compare the p-values better. For plink and own, I used the 10 covariates, plus birth year and sex as covariates. Otherwise, the QC is the same for plink and own since I converted from the final QC file to the raw file
  + Neale Covariates: 1st 20 PCs + sex + age + age^2 + sexage + sexage2

### 5/26/2021

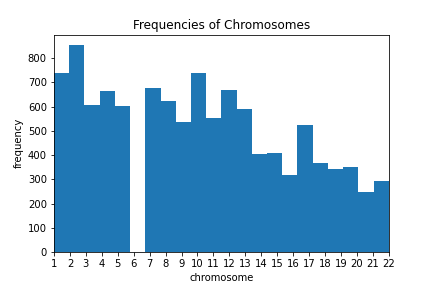
* Scatterplot of p-values for Neale and plink2 results of gwas for height



* + Clear diagonal line; plink2’s negative log10 p-values are consistently higher than Neale’s
    - Plink2 results come from QC with larger sample size
  + Two ‘prongs’ concentrated at the sides
* Histogram of chromosome frequencies of side prongs
  + Vertical prong



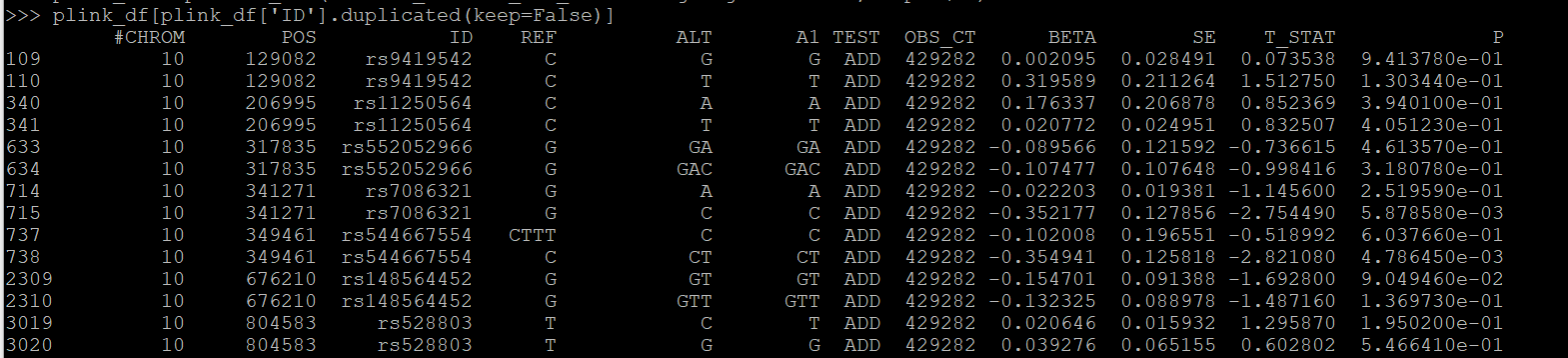
* + Horizontal prong



* Possible duplication when joining plink2 and Neale results, since joined on position, which is not unique
  + Join on combination of chromosome, position, ref and alt allele

5/27/2021

* 756 duplicated rows when checking variant
* 18943 duplicated rows when checking ID



* Neale has no duplicated rows
* Remove duplicates and indels from QC
  + Duplicates likely to be indels or multiallelic SNPs
  + Indels signified by ID column <chr>:<position>:<REF ALLELE>:<ALT ALLELE>

### 6/1/2021

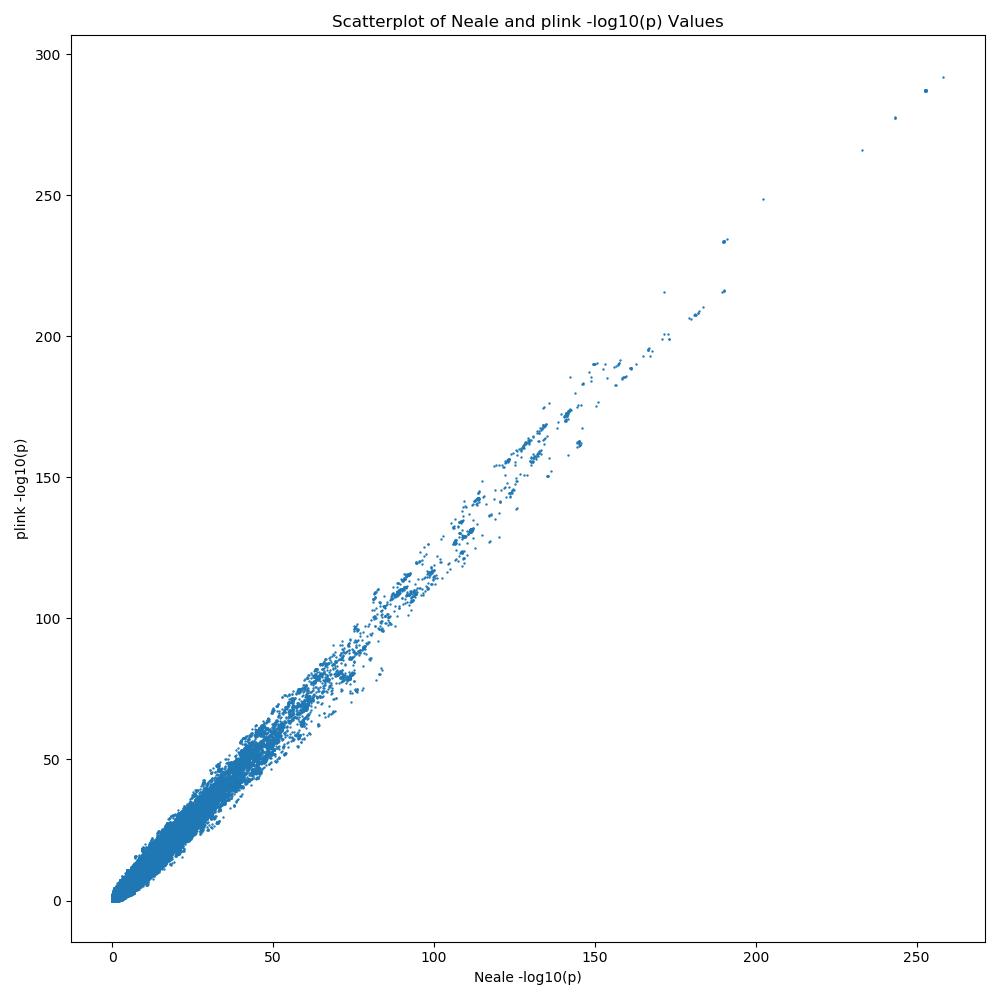
* Add genotype measurement batch, plate, and wells as covariates
  + Why: each study site might have specific preference/bias for calling allele which can drive significant associations between SNP and phenotype
* Error with new association



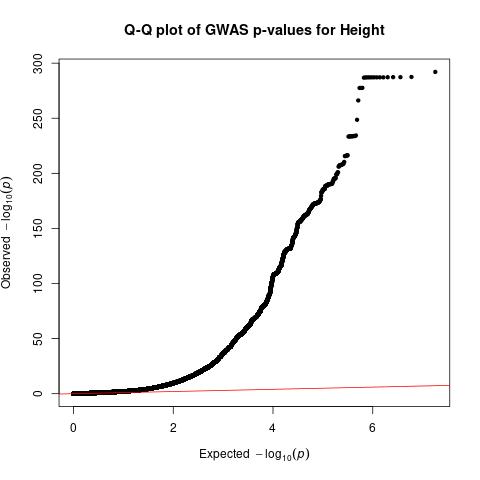
* + Only kept ‘genotype measurement batch out of the three genotype measurement options
* Messaged Jared about shifted metadata and ukbgene permissions
* Combine updated gwas results

### 6/2/2021

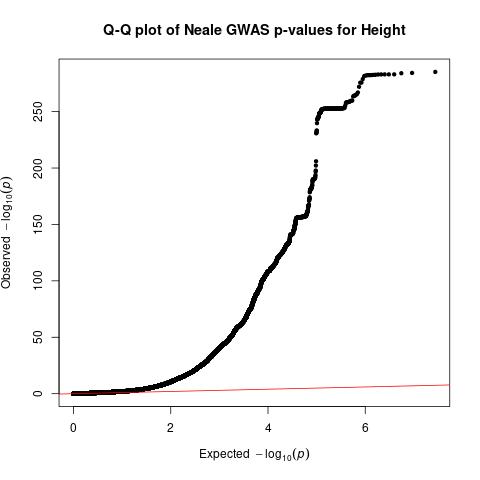
* Message TACC consulting about corral-repl not being available in development environment
* Ukbconv and ukbgene usage – Jared’s email
* Created new scatterplot joined on variant
  + Side clusters are gone, now see a pretty clear linear relationship
  + Correlation: 0.694



* QQ plot of plink2 gwas results on height

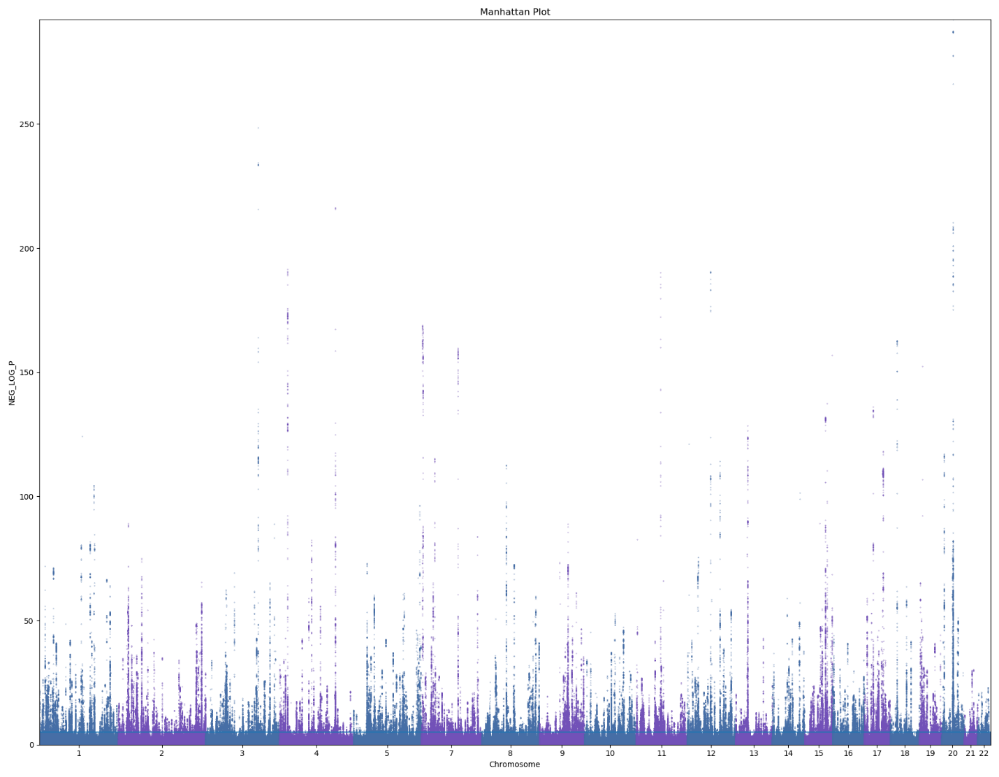


* QQ plot of Neale Lab gwas results on height

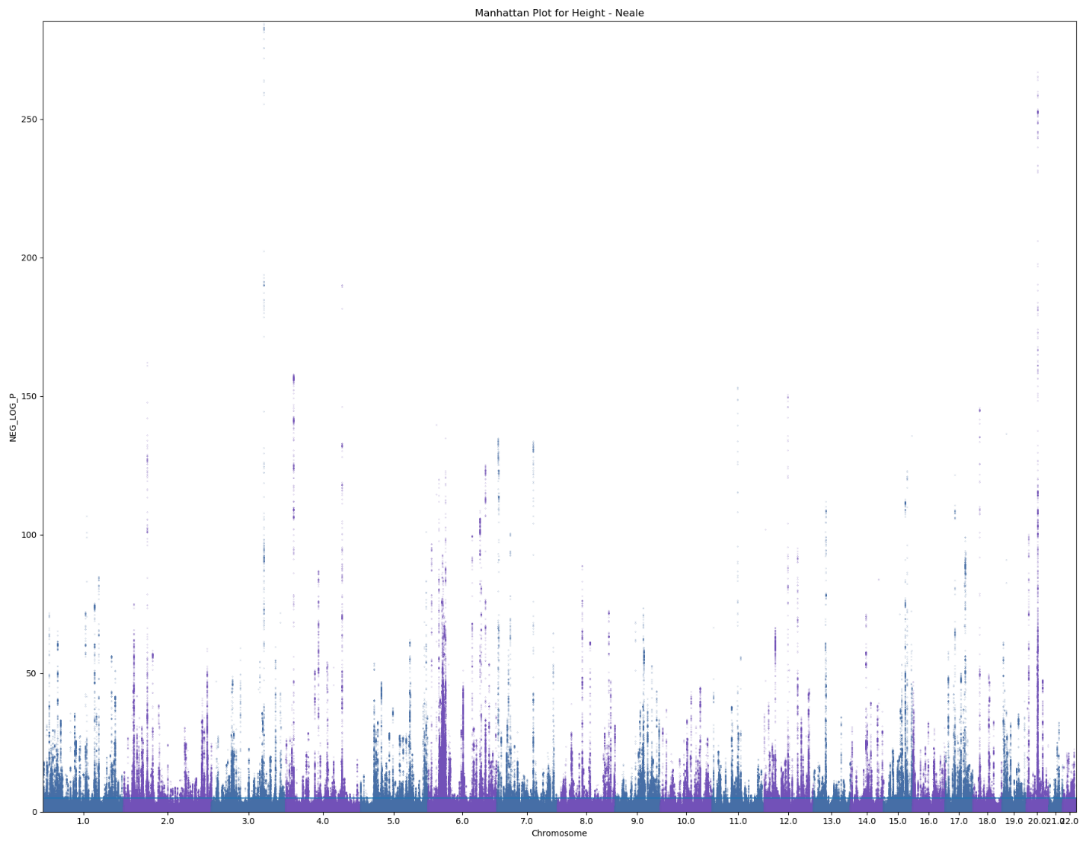


### 6/3/2021

* /corral-repl and ukbconv
* Manhattan plots
  + Plink2

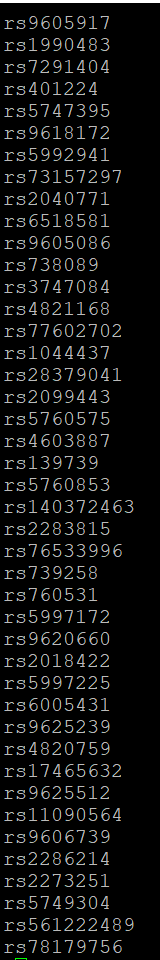


* + Neale

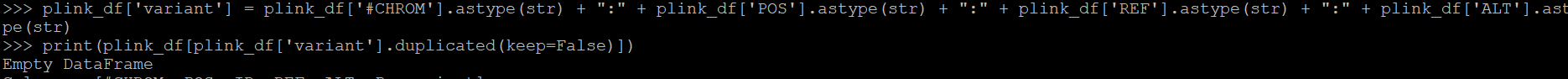


### 6/7/2021

Examine Duplicates



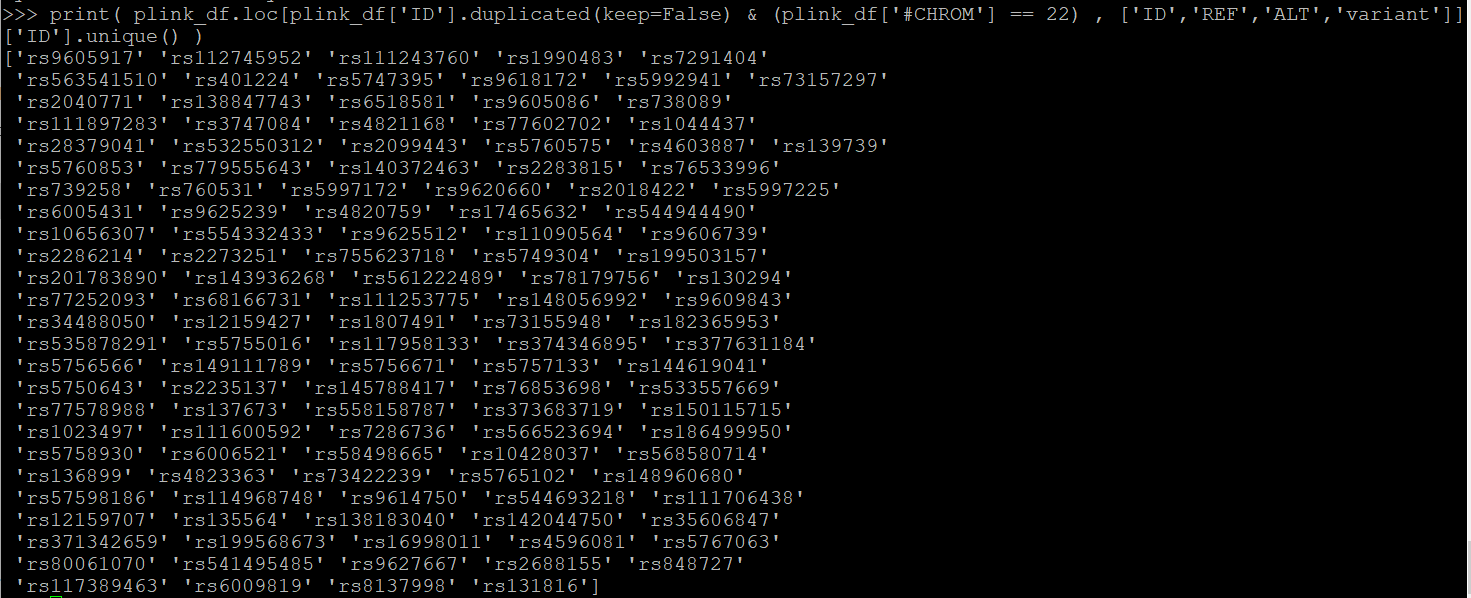
* Rm-dup file for chr 22 : 93 ids



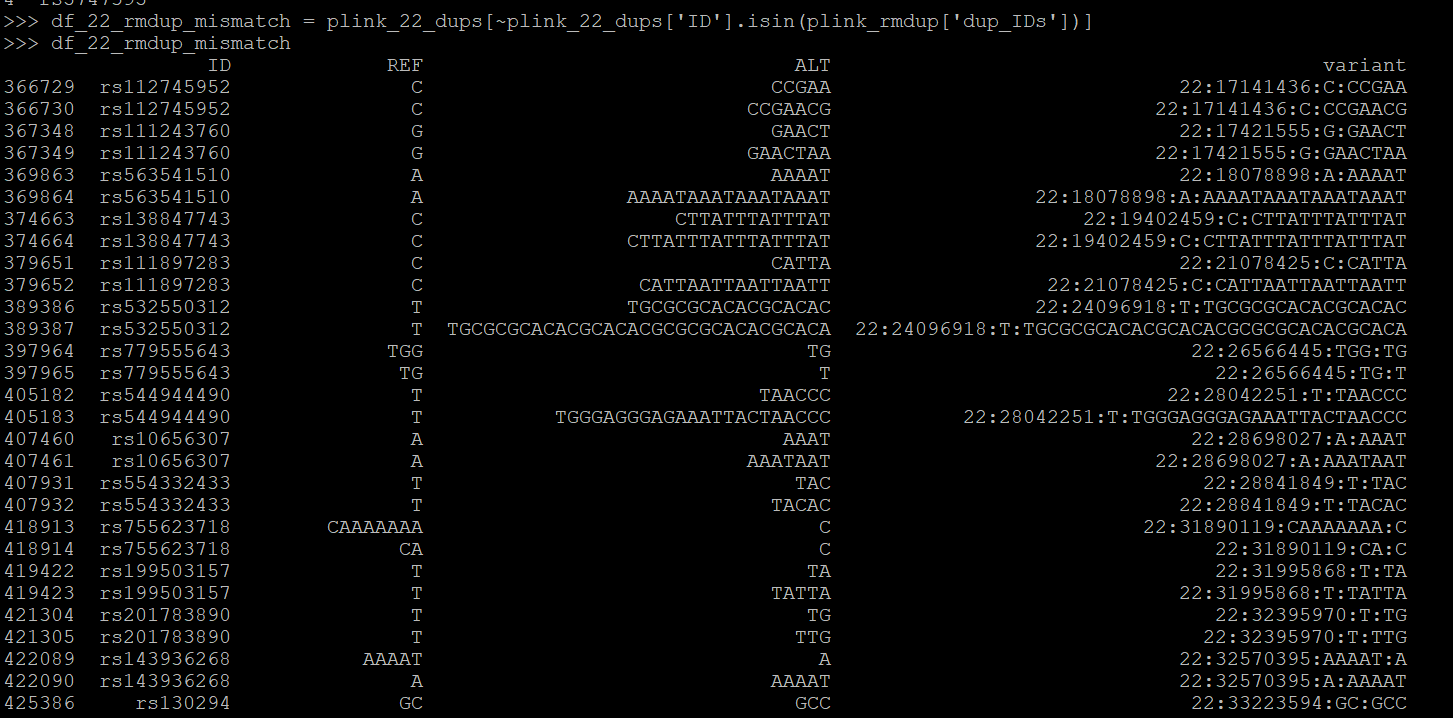
* None duplicated on variant



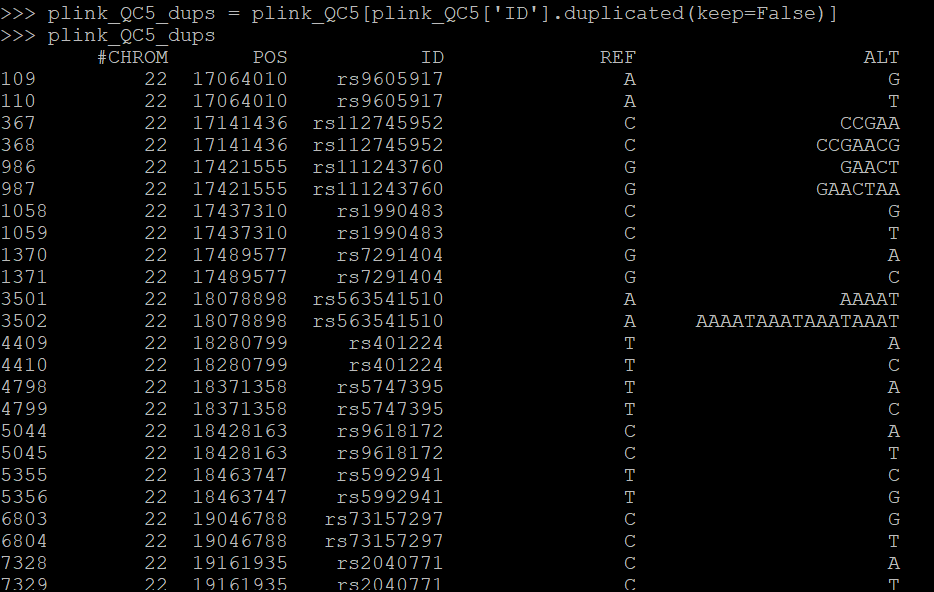
* For chr22 plink results file : 256 duplicated rows



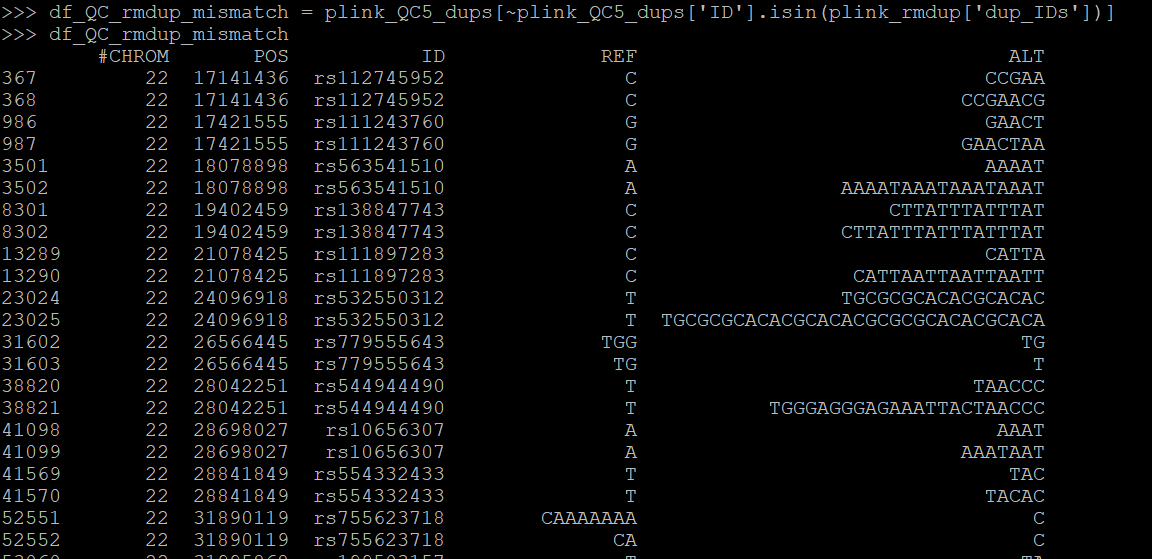
* For chr22 plink results file : 127 unique duplicates



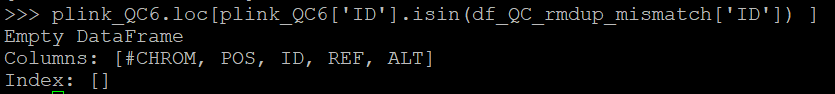
* Duplicates from chr 22 results file that were not removed by rm-dup : all are multiallelic SNPs – 70 rows



* Duplicates in QC file for chr 22 before rm-dup: also has 127 unique duplicates

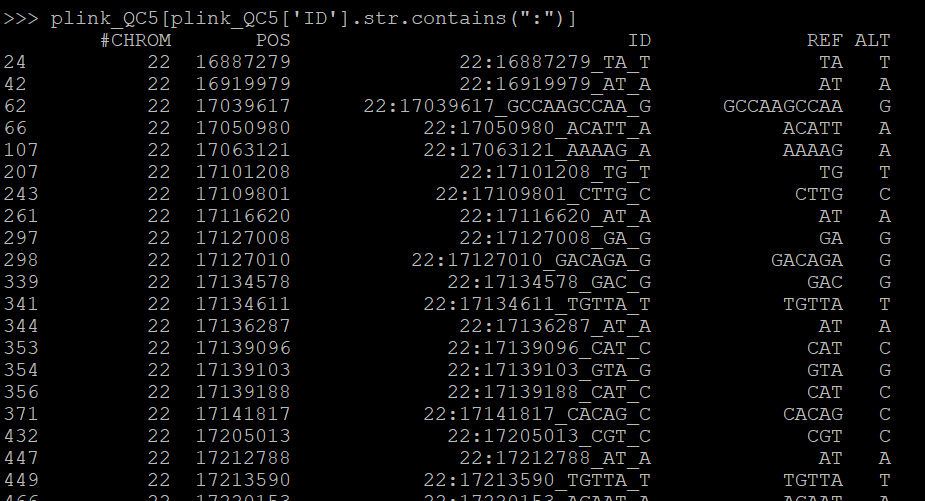


* Duplicates from QC5 file for chr 22 that were not removed by rm-dup : same as mismatch between results file and rmdup file (70 rows)

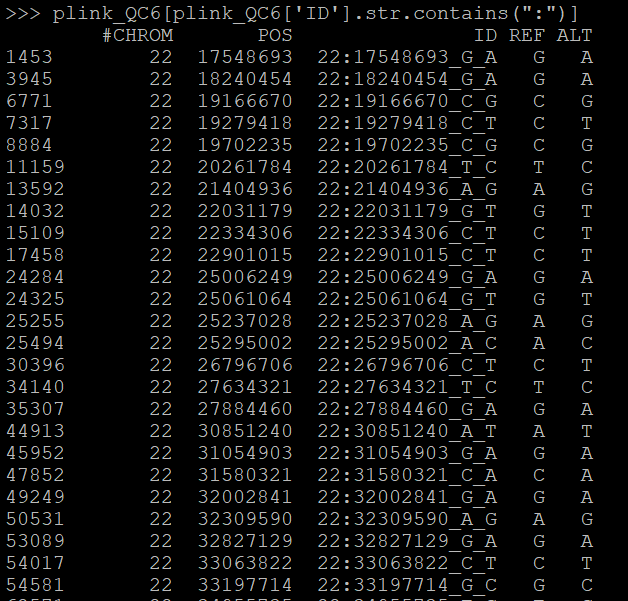


* QC6 does not contain any of the multiallelic SNPs (from the mismatched df)
* Snp-only tag removes all the multi-allelic ids

Check Indels



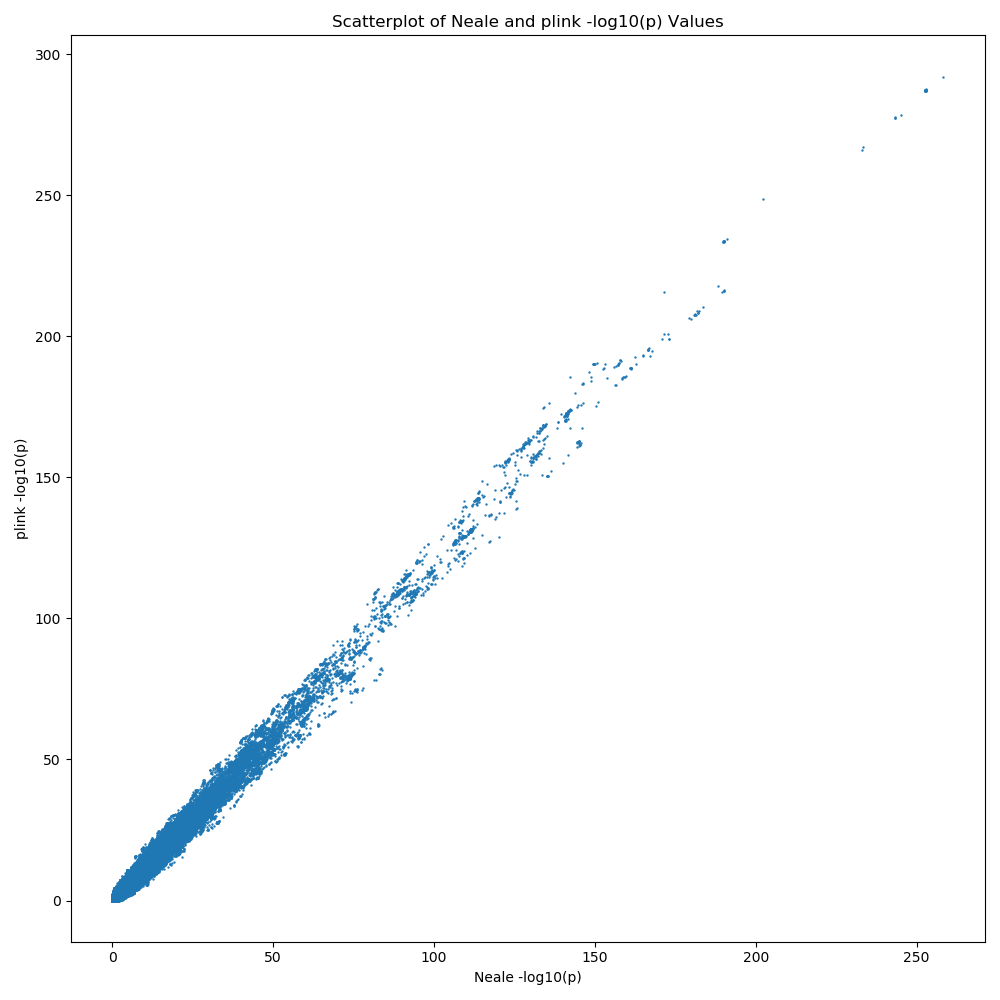
* QC5 file for chr22 contains 4852 unique rows of indels (checked with unique())



* QC6 file for chr 22 contains 70 unique indels (they are not the same as the 70 multiallelic SNPs from before)
* Only ones with a single allele in REF and ALT are kept

Summary: The duplicates in the QC5 files (before rm-dup and snps-only) are duplicates on rsids, but they have different REF and ALT alleles. So there are no full duplicates. –rm-dup modifier removes duplicate IDs for SNPs and not for multiallelic IDs. The –snps-only modifier removes multiallelic SNPs/indels but does not remove single allele indels

Remake p-value scatterplot with correct ‘variant’

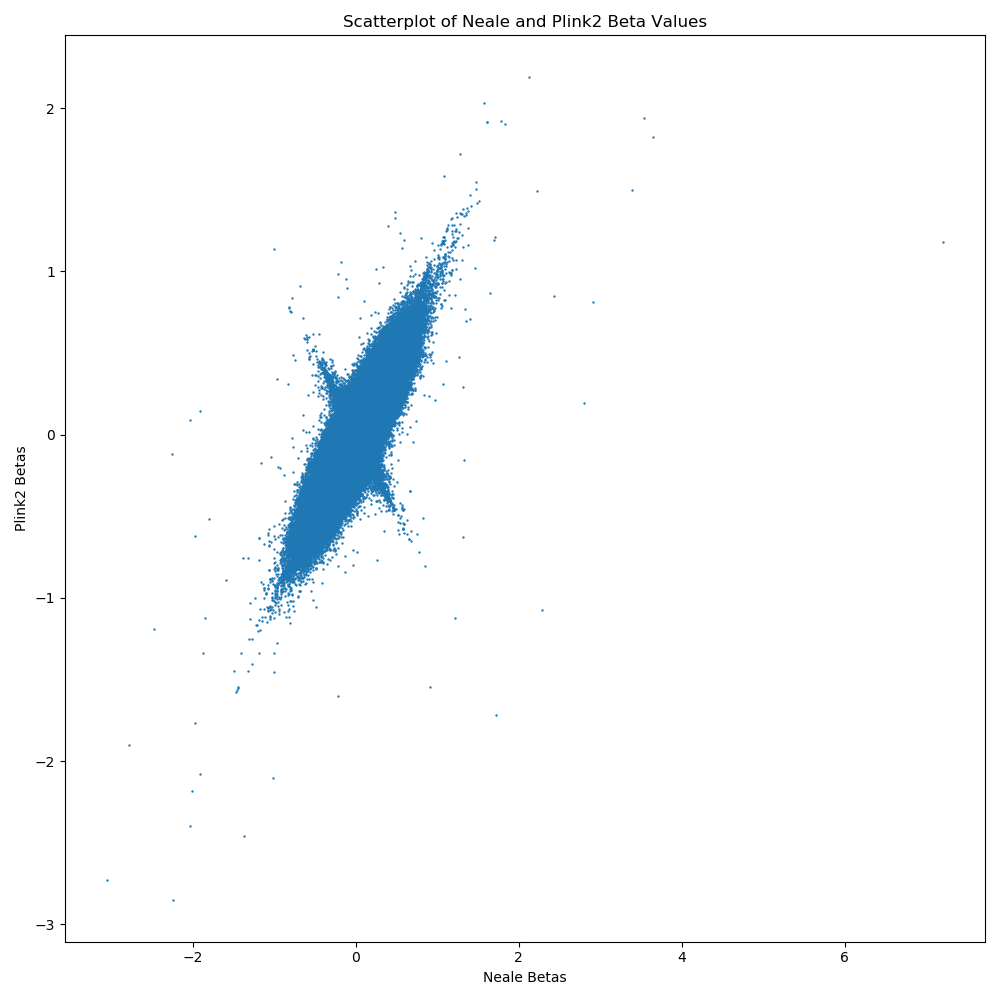


### 6/10/2021

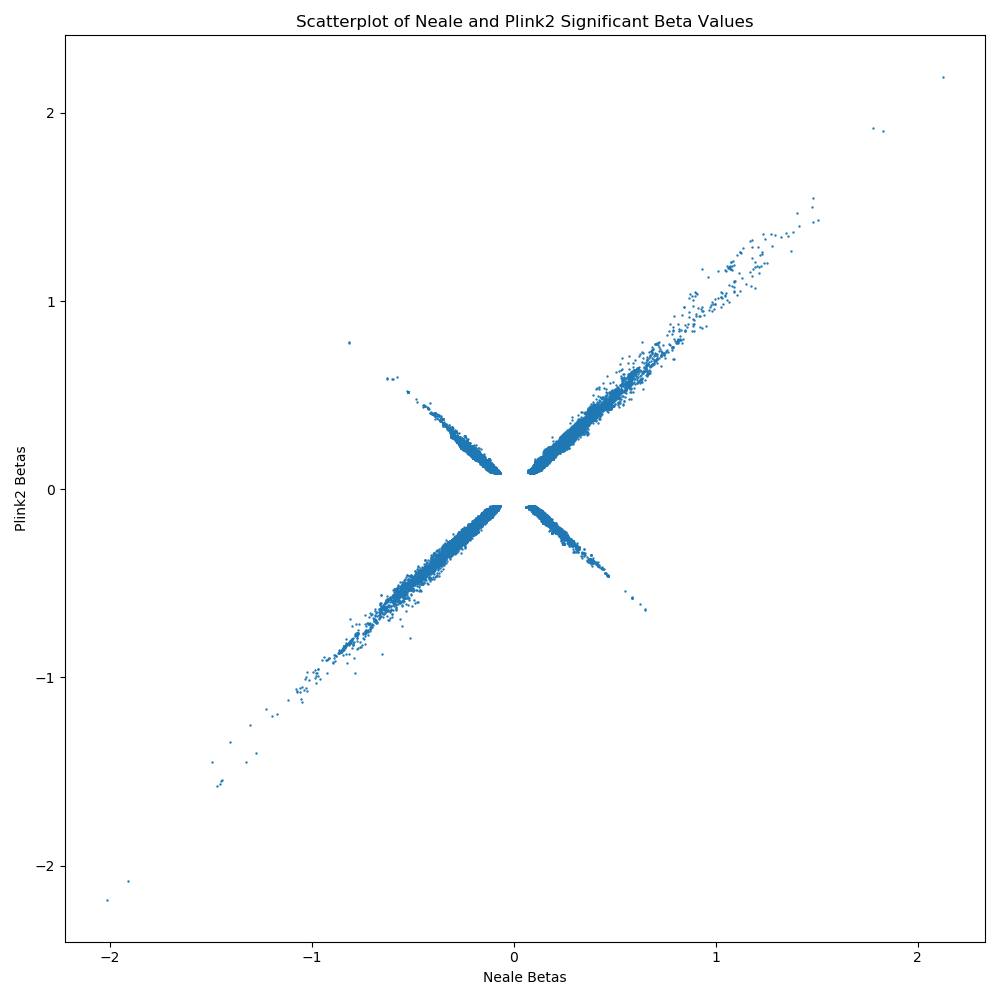
* New association with corrected QC, remove genotype measurement batch as a covariate

### 6/11/2021

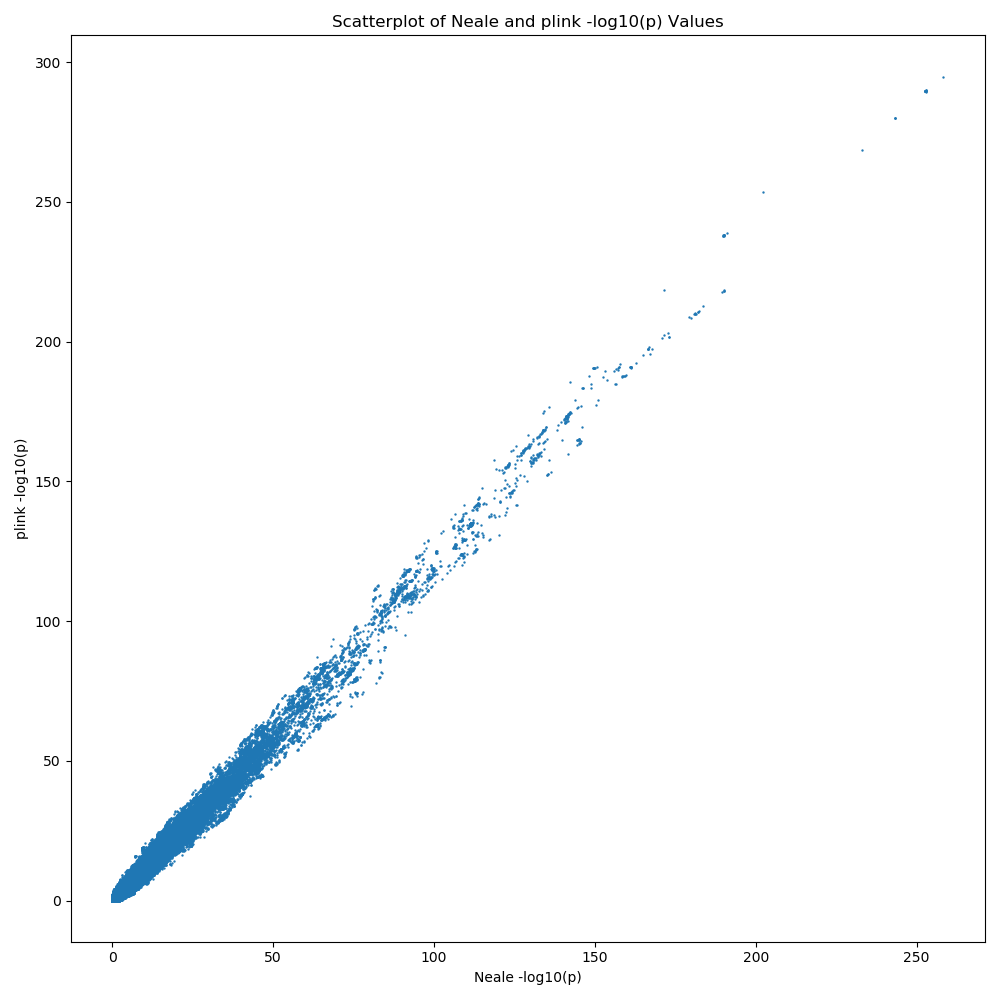
* Added Chromosome 6
* Beta scatterplot (compare Neale and Plink betas)



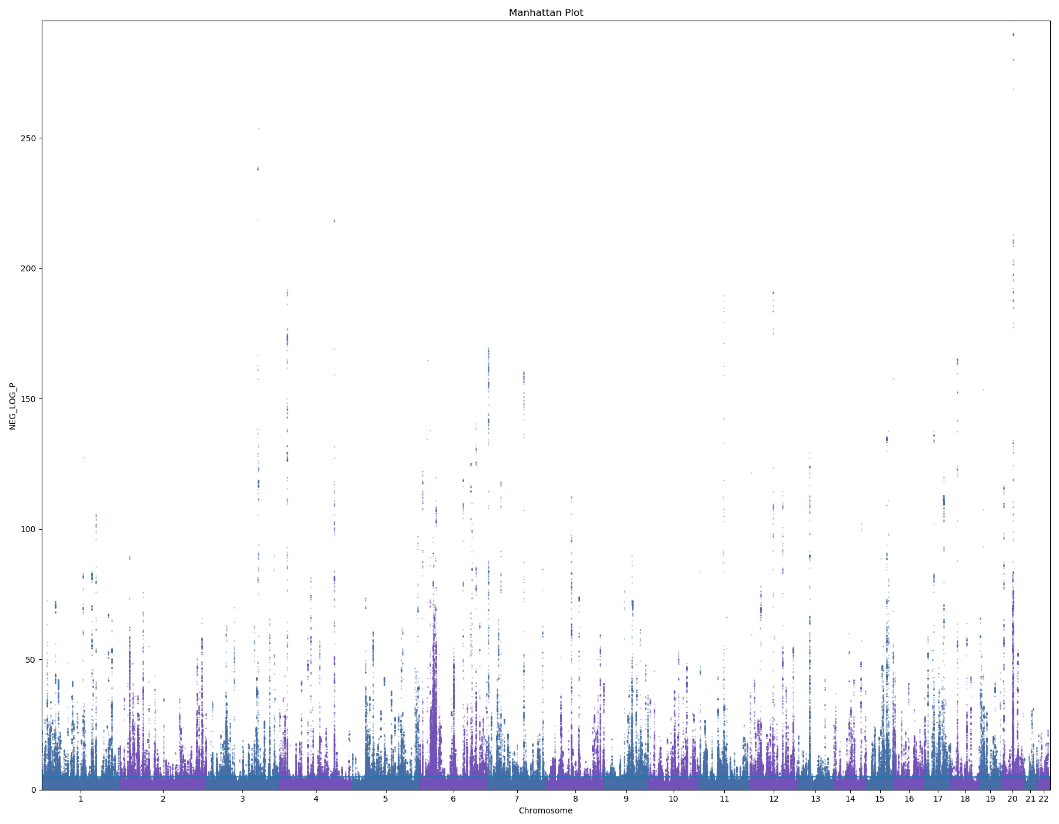
* Beta scatterplot for low p-values

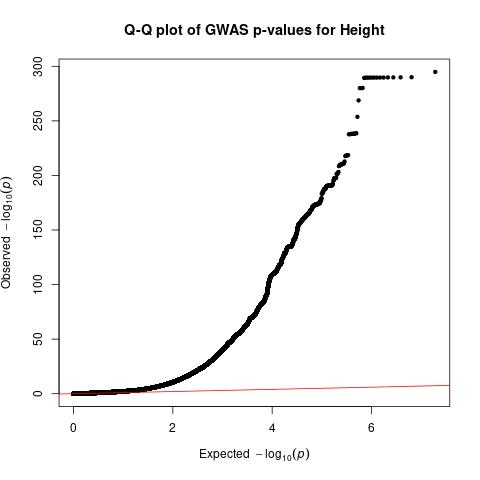


* Scatterplot p-values again



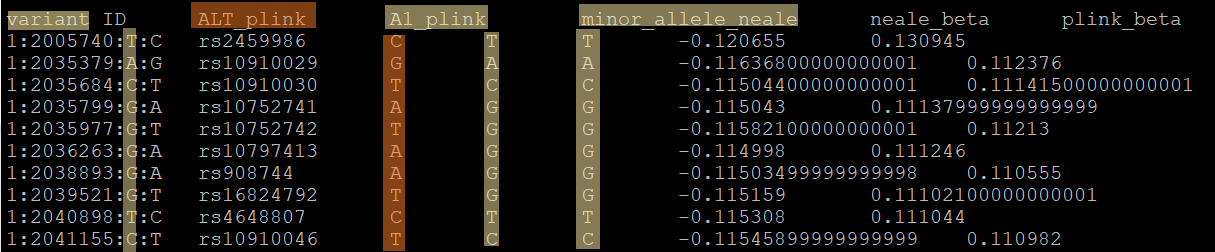
* Manhattan and QQ plot final



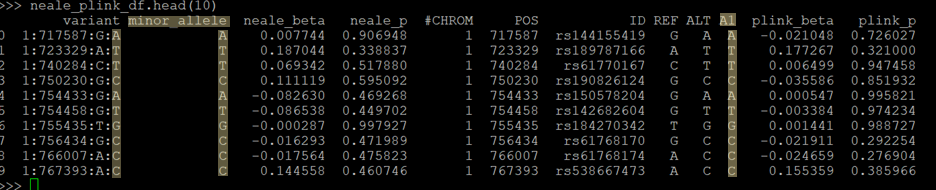


### 6/14/2021

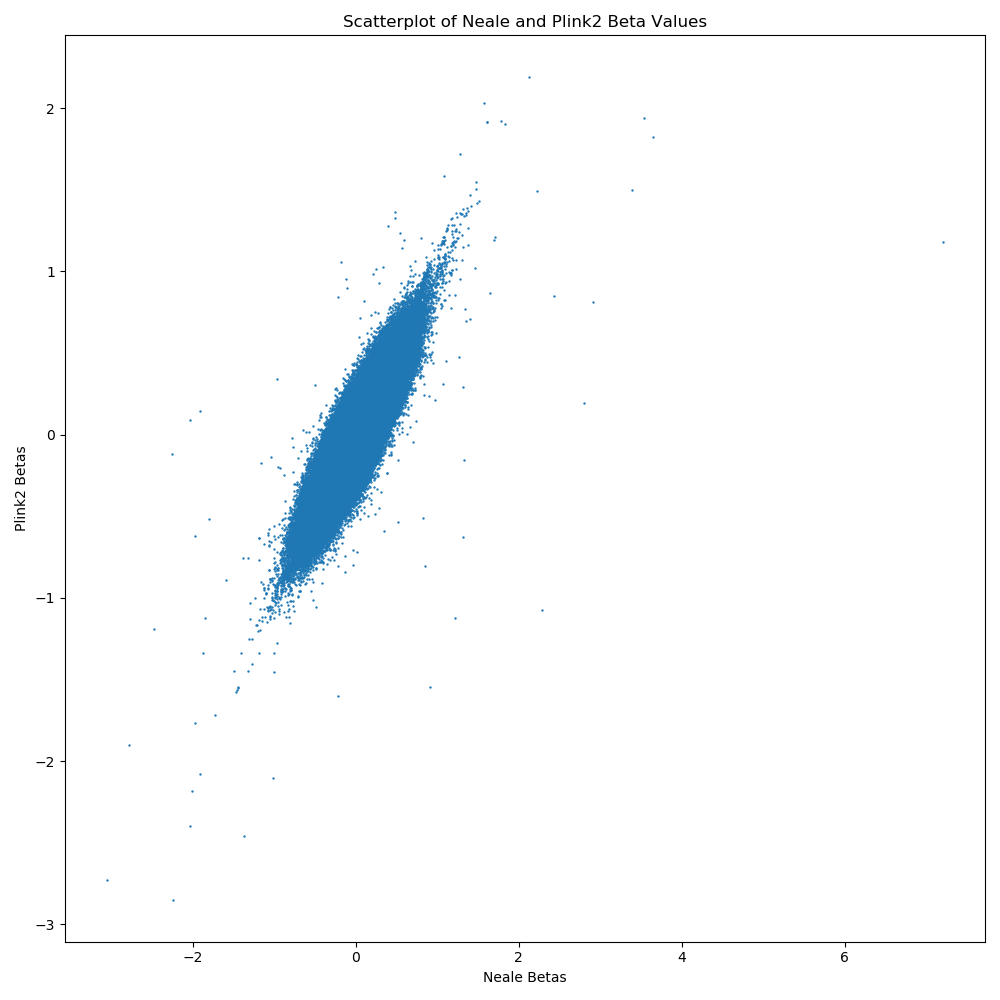
* Focus on the downwards diagonal in beta scatterplot
  + Length prong1: 14886 – all of them, the ALT and A1 allele are not the same
  + Length prong3: 11962 – all of them, the ALT and A1 allele are not the same
  + Total low-p betas: 104036
  + When the counted allele is the ref allele and not the alt allele, the sign of beta from plink is opposite from neale’s beta signs
* Prong 1 sample

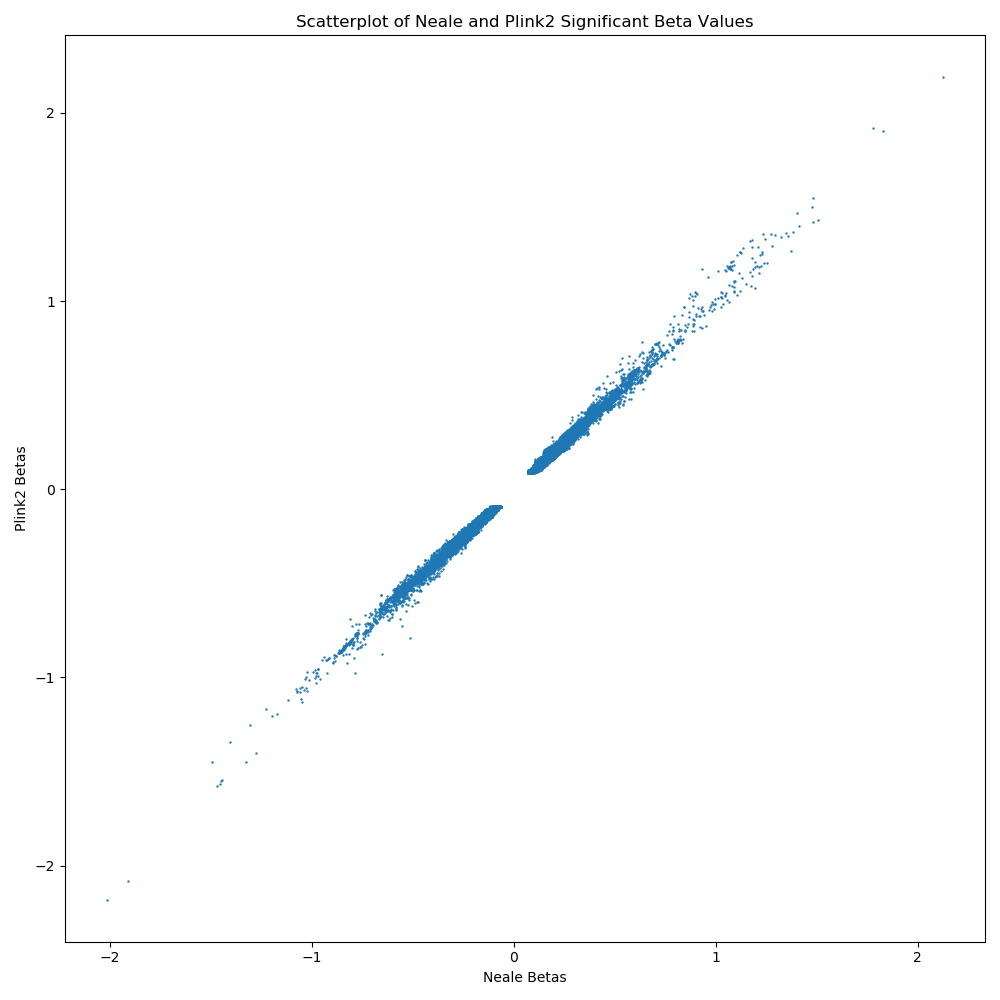


* Normal sample



* Neale Lab uses the alternative allele as the effect allele while plink2 uses the A1 (minor allele) as the effect allele
  + Alt allele does not always mean minor allele
  + To have plink2 use the alt allele in regression, use the ‘omit-ref’ command
* New beta scatterplot: switched the sign for neale betas if the ALT and A1 allele didn’t match in plink2





### 6/15/2021

* Add AX column (non-effect allele)
* GWAS of height for male and female separately

### 6/16/2021

* Modify combine\_results.sh for new AX column in results
* Modify association script to make more concise, easier to use
* Made Manhattan script for universal, automatically gives female, male, and both sex plots; pass in phenotype as argument; old Manhattan script:

#!/usr/bin/env python3

# import packages

import matplotlib

matplotlib.use("Agg")

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import os

# set working directory

os.chdir("/scratch1/08005/cz5959/Association\_Height\_50")

# DATAFRAME

# load file to dataframe

results\_df = pd.read\_csv("linear\_results\_all\_chrom.height.glm.linear", sep="\t", usecols =['#CHROM','POS','P','ID'])

# drop rows with any column having null/missing data

results\_df = results\_df.dropna()

# sort by column then position; reset index

results\_df = results\_df.sort\_values(['#CHROM', 'POS'])

results\_df.reset\_index(inplace=True, drop=True)

# change P column to float type

results\_df['P'] = pd.to\_numeric(results\_df['P'])

# create column with negative log p value

results\_df['NEG\_LOG\_P'] = -np.log10(results\_df['P'])

# make chromosome column into type category

results\_df['#CHROM'] = results\_df['#CHROM'].astype('category')

# index; used for x axis; assume uniform SNP distrubtion across chromosome

results\_df['index'] = range(len(results\_df))

# group by chromosome

grouped\_df = results\_df.groupby(('#CHROM'))

#PLOT

fig = plt.figure(figsize=(18,14))

# axes of figure - 1row,1col,1idx

ax = fig.add\_subplot(111)

colors = ['#466EA6','#7251B8']

x\_labels = []

x\_labels\_pos = []

# create subplots for each chromosome (name = #CHROM)

for num, (name, group) in enumerate(grouped\_df):

    ##### plot, x is index and y is neg log p ######

    group.plot(kind='scatter', x='index', y='NEG\_LOG\_P',color=colors[num % len(colors)], ax=ax, s=0.1, marker = '.')

    # name of chr

    x\_labels.append(name)

    # tick marks; middle of group

    x\_labels\_pos.append((group['index'].iloc[-1] - (group['index'].iloc[-1] - group['index'].iloc[0])/2))

#line

ax.plot([0,len(results\_df)],[5,5])

# figure labels

ax.set\_xticks(x\_labels\_pos)

ax.set\_xticklabels(x\_labels)

ax.set\_xlim([0, len(results\_df)])

y\_max = results\_df['NEG\_LOG\_P'].max()

print(y\_max)

ax.set\_ylim([0, y\_max])

ax.set\_xlabel('Chromosome')

ax.set\_title('Manhattan Plot')

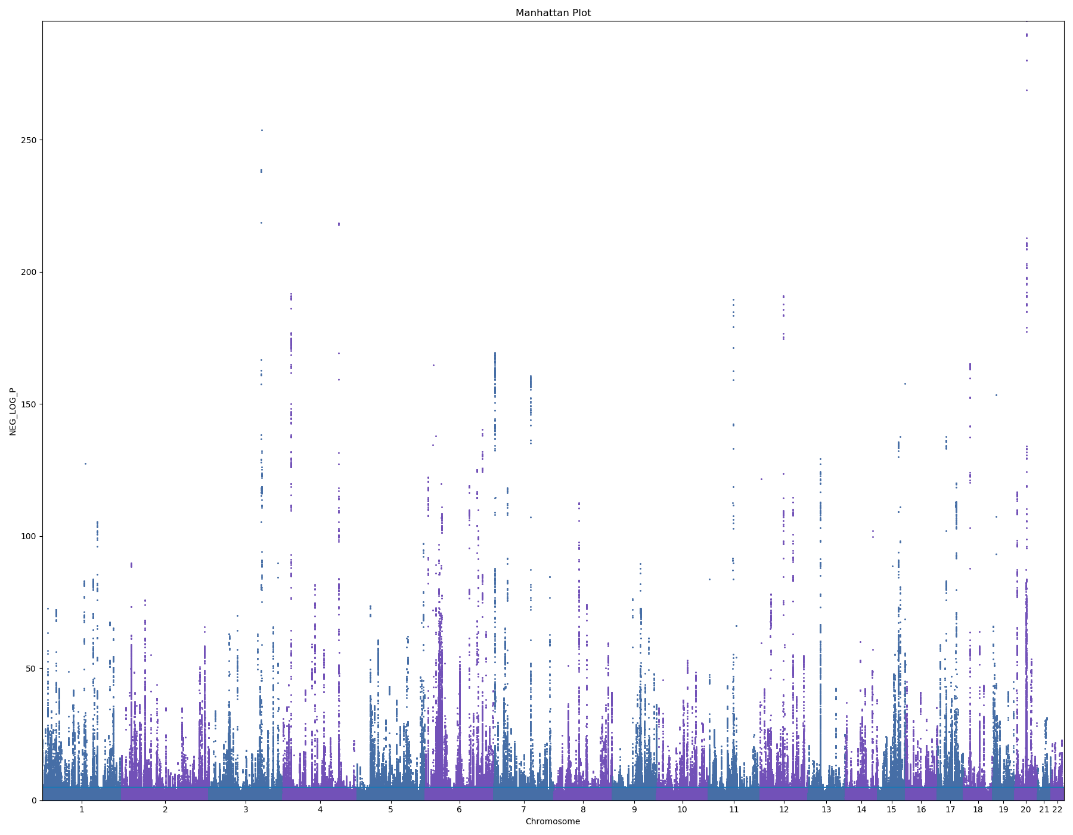
# save as png

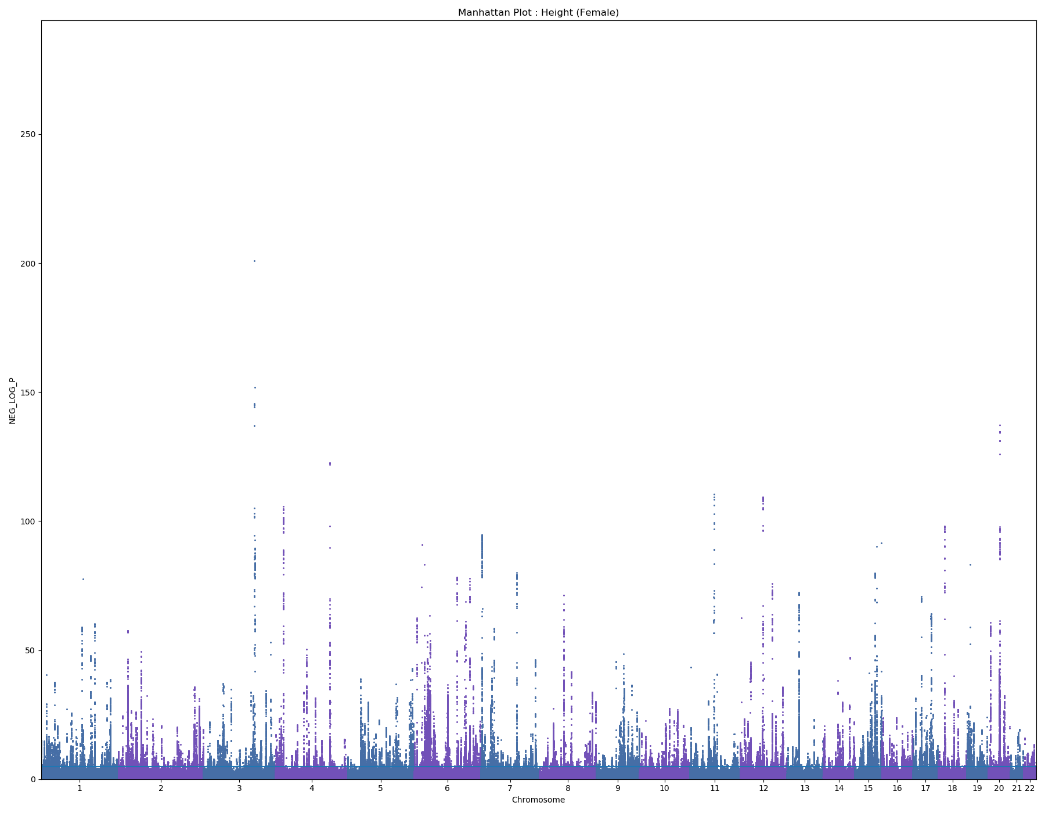
plt.tight\_layout()

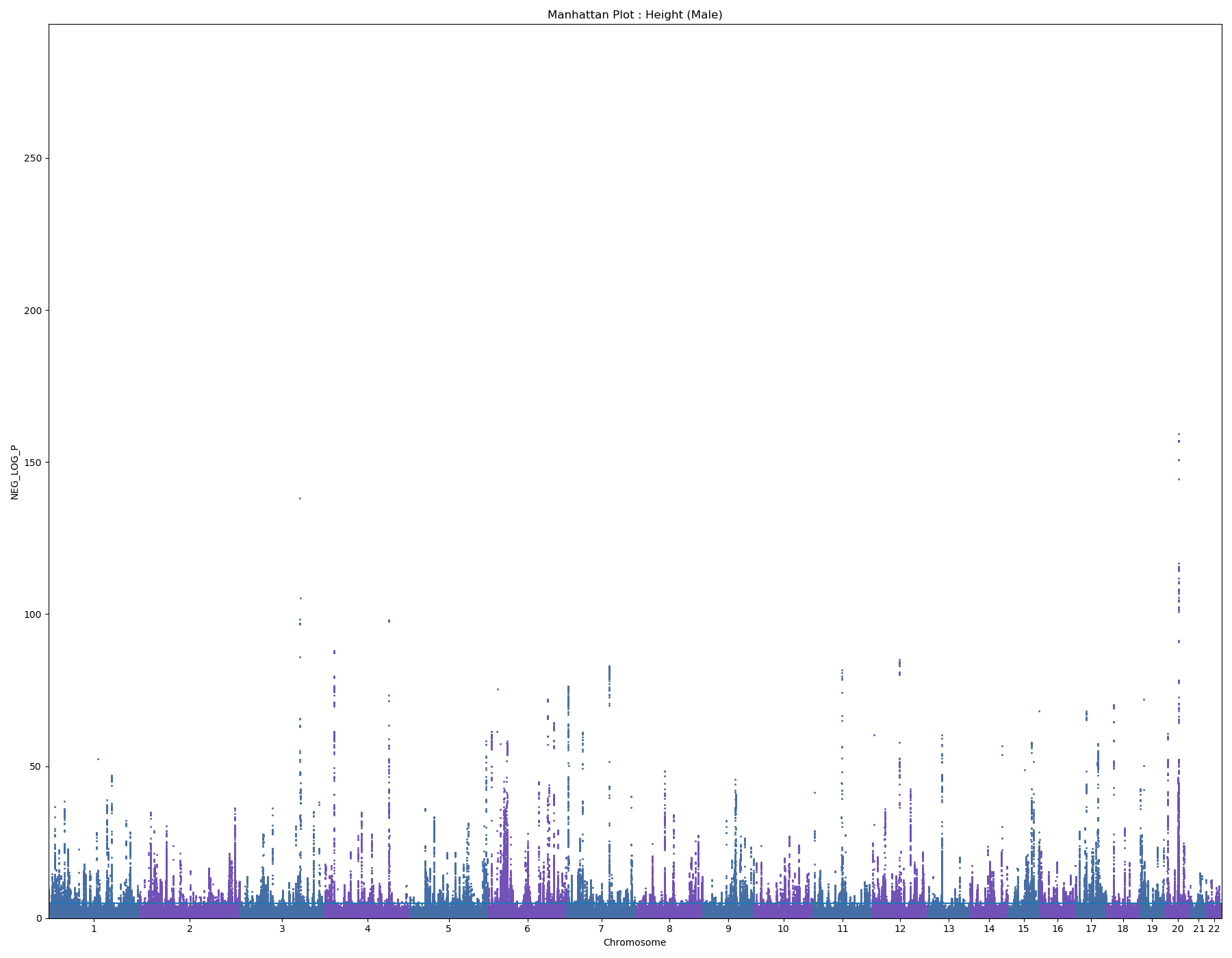
plt.savefig("manhattan\_height.png")

### 6/17/2021

* Both, Female, and Male Manhattan Plots
  + Make y-max the same for all : max of both\_sex plot
  + The peaks are pretty similar overall
  + Peaks shorter for male and female because smaller sample size (not as small p-value, not as many tagging causal)







* Gwas of testosterone