Introduction to Genetic Epidemiology

Shea J Andrews, PhD

3/14/24

Table of contents

I Introduction 5

Schedule 7

Week 1 04/22 - 04/26 7
Monday 04/22 7
Wensday 04/24 7
Friday 04/26 8
Week 2 04/29 - 05/03 9
Monday 04/29 9
Wensday 05/01 9
Friday 05/03 10
Week 3 05/06 - 05/10 10
Monday 05/06 10
Wensday 05/08 11
Friday 05/10 11

Conda Env 12

PLINK 12

HABS-HD 13

Phenotypes 13
Descriptive Table 15
Genotyping 24
HapMap III 24

Summary Statistics 26

Lipids 27
Willer et al 2013 27
Graham et al. 2021 27
Alzheimer's disease 29
Kunkle 2019 29
Bellenguez 2022 30
Educational Attainment 32
Lee et al 2018 32

II Genome-wide Association Studies 34

GWAS QC 35
SNP QC . . . 35
Call Rate & Allele frequency . . . 35
Hardy Weinberg Equilibrium . . . 37
Sample QC . . . 38
Call Rate . . . 38
Sex Discordance . . . 39
Pruning . . . 40
Heterozygosity check . . . 40
Cryptic Relatedness . . . 41
Population Substructure . . . 45
Scree Plot . . . 46
Population substructure . . . 47
Exclude Samples . . . 47

GWAS 48
Import . . . 48
Phenotype File . . . 48
Covariate File . . . 48
GWAS . . . 49
Manhattan Plot . . . 49

GWAS-SS 50
AD GWAS . . . 50
MungeSumstats . . . 51
Manhattan Plot . . . 51

Genetic Ancestry 54
Prinicpal Component Analysis . . . 54
PLINK PCA . . . 54
Cluster Assignment . . . 55
Visualization . . . 60
ADMIXTURE . . . 61
Reference Processing . . . 61
ADMIXTURE Procedure . . . 61
Visualization . . . 67

Heritability & Genetic Correlations 69
Methods . . . 69
Tools & Publications . . . 69
Munge . . . 70

LDSC 71
HDL 72

III Polygenic Risk Scores **73**

Polygenic Risk Scores **74**

IV Mendelian Randomization **75**

Mendelian Randomization **76**

Appendices **76**

References **77**

Acknowledgments **78**

Part I

Introduction

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

Schedule

Welcome to this BMI Mini-Course on an Methods in Genetic Epidemiology. The course structure is divided into three key methodologies: genome-wide association studies, polygenic risk scores, and Mendelian randomization. Each trainee will apply these methods to a trait of their choosing and, at the course's conclusion, present their findings to the class.

Week 1 04/22 - 04/26

Monday 04/22

Alzheimer's disease

In this mini-course, we will frequently reference Alzheimer's disease to illustrate various genetic epidemiology methods. This session aims to introduce Alzheimer's disease and explore significant findings related to its genetic architecture.

Readings

- Knopman, D. S. et al. Alzheimer disease. Nat Rev Dis Primers 7, 33 (2021).
- Alzheimer's Association. 2023 Alzheimer's disease facts and figures. Alzheimer's Dementia (2023).
- Kornblith, E. et al. Association of Race and Ethnicity With Incidence of Dementia Among Older Adults. Jama 327, 1488–1495 (2022).
- Andrews, S. J. et al. The complex genetic architecture of Alzheimer's disease: novel insights and future directions. eBioMedicine 90, 104511 (2023).
- Andrews, S. J., Fulton-Howard, B. & Goate, A. Interpretation of risk loci from genome-wide association studies of Alzheimer's disease. Lancet Neurology 19, 326–335 (2020).

Wensday 04/24

Genome-Wide Association Studies

Genome-Wide Association Studies (GWAS) are foundational to various genetic analysis methodologies. In this session, we will delve into what a GWAS entails, the process of conducting one, and then engage in a hands-on exercise to carry out our own GWAS.

Coursework

- Slides
- GWAS QC
- GWAS
- GWAS SS

Readings

- Uffelmann, E. et al. Genome-wide association studies. Nat Rev Methods Primers 1, 59 (2021).
- Abdellaoui, A., Yengo, L., Verweij, K. J. H. & Visscher, P. M. 15 years of GWAS discovery: Realizing the promise. Am J Hum Genetics (2023)
- Marees, A. T. et al. A tutorial on conducting genome-wide association studies: Quality control and statistical analysis. Int J Method Psych 27, e1608 (2018).
- MacArthur, J. A. L. et al. Workshop proceedings: GWAS summary statistics standards and sharing. Cell Genom 1, 100004 (2021).

Tools

- PLINK[1]
- BCFTools[2]
- MungeSumStats[3]

Friday 04/26

Genetic Ancestry

Genetic ancestry explores the lineage and heritage inferred from our DNA, providing insights into population history and individual heritage. This session will introduce the concepts and methodologies used in determining genetic ancestry, emphasizing their importance in genetic epidemiology research.

Coursework

- Slides
- Genetic Ancestry

Readings

- Lewis, A. C. F. et al. Getting genetic ancestry right for science and society. Science 376, 250–252 (2022).
- National Academies of Sciences, Engineering, and Medicine. 2023. Using Population Descriptors in Genetics and Genomics Research: A New Framework for an Evolving Field. Washington, DC: The National Academies Press.

Tools

- ADMIXTURE[4]
- RFmix[5]

Week 2 04/29 - 05/03

Monday 04/29

Heritability & Genetic Correlations

Heritability quantifies the proportion of phenotype variance attributable to genetic factors, whereas genetic correlations assess the extent of shared genetic architecture between traits. In this session, we will concentrate on the tools utilized to estimate these metrics from GWAS summary statistics.

Readings

- Rheenen, W. van, Peyrot, W. J., Schork, A. J., Lee, S. H. & Wray, N. R. Genetic correlations of polygenic disease traits: from theory to practice. Nat Rev Genetics 20, 567–581 (2019).
- Barry, C.-J. S. et al. How to estimate heritability: a guide for genetic epidemiologists. Int J Epidemiol (2022)

Tools

- LDSC[6]
- HDL[7]
- GenomicSEM[8]

Wensday 05/01

Polygenic Risk Scores I

Polygenic risk scores (PRS) measure an invididueals total genetic liability for a trait. This session will cover the process of constructing a PRS and assessing its performance in predicting the trait.

Readings

- Choi, S. W., Mak, T. S.-H. & O'Reilly, P. F. Tutorial: a guide to performing polygenic risk score analyses. Nat Protoc 15, 2759–2772 (2020).
- Wand, H. et al. Improving reporting standards for polygenic scores in risk prediction studies. Nature 591, 211–219 (2021).

- Lennon, N. J. et al. Selection, optimization and validation of ten chronic disease polygenic risk scores for clinical implementation in diverse US populations. Nat. Med. 1–8 (2024)

Tools

- PRSice2[9]
- PRSet[10]

Friday 05/03

Polygenic Risk Scores II

The accuracy of polygenic risk scores (PRS) diminishes as the genetic distance from the training population increases. This session will explore cross-ancestry PRS methods designed to enhance PRS accuracy across diverse populations.

Readings

- Kachuri, L. et al. Principles and methods for transferring polygenic risk scores across global populations. Nat. Rev. Genet. 1–18 (2023) doi:10.1038/s41576-023-00637-2.
- Ding, Y. et al. Polygenic scoring accuracy varies across the genetic ancestry continuum. Nature 618, 774–781 (2023).

Tools

- PRS-CSx[11]

Week 3 05/06 - 05/10

Monday 05/06

Mendelian Randomization I

Mendelian Randomization (MR) is a method employed to identify causal risk factors for diseases. This session will cover the fundamentals of MR and demonstrate how to execute a two-sample MR analysis.

Readings

- Sanderson, E. et al. Mendelian randomization. Nat Rev Methods Primers 2, 6 (2022).
- Davies, N. M., Holmes, M. V. & Smith, G. D. Reading Mendelian randomisation studies: a guide, glossary, and checklist for clinicians. BMJ 362, k601 (2017).
- Hemani, G. et al. The MR-Base platform supports systematic causal inference across the human phenome. Elife 7, e34408 (2018).

Tools

- TwoSampleMR[12]

Wensday 05/08

Mendelian Randomization II

A crucial aspect of Mendelian Randomization (MR) studies is assessing whether the causal associations derived from MR analyses remain valid despite potential violations of MR's underlying assumptions. This session will focus on diagnostic and sensitivity analyses in MR, along with guidance on effectively reporting MR findings.

Readings

- Skrivankova, V. W. et al. Strengthening the Reporting of Observational Studies in Epidemiology Using Mendelian Randomization. JAMA 326, 1614–1621 (2021).
- Skrivankova, V. W. et al. Strengthening the reporting of observational studies in epidemiology using mendelian randomisation (STROBE-MR): explanation and elaboration. BMJ 375, n2233 (2021).

Friday 05/10

Trainee Presentations

Upon concluding this mini-course, trainees will showcase the results of their analyses.

Conda Env

Clone the Introduction to Genetic Epi repo

```
#| eval: false
git clone https://github.com/AndrewsLabUCSF/IntroGeneticEpi.git`
```

We will use a conda environment to run our projects. If you dont already have conda installed, install miniconda or micromamba.

The following yaml file defines what software we will install in the conda environment named genetic_epi

```
#| eval: false
name: genetic_epi
channels:
  - conda-forge
  - bioconda
dependencies:
  - plink=1.90b6.21
```

Run the following code to install the conda environment

```
#| eval: false
conda env create -f envs/genetic_epi.yml
```

PLINK

Install plink

```
#| eval: false
sudo cp path/to/plink /usr/local/bin/`
```

HABS-HD

We will be using Health and Aging Brain Study-Health Disparities (HABS-HD) as an example data for conducting genome-wide assocation studies. Phenotypic and genetic data will be shared via Box.

Phenotypes

```
library(tidyverse)
setwd('~/gitcode/IntroGeneticEpi/')
SAVE_VISUALIZATIONS_PATH <- "results/figures"
```

```
aa_v1_path = 'resources/HABSHD/v5/HD 1 African American 50+ Request 355.csv'
ma_v1_path = 'resources/HABSHD/v5/HD 1 Mexican American 50+ Request 355.csv'
nhw_v1_path = 'resources/HABSHD/v5/HD 1 Non-Hispanic White 50+ Request 355.csv'

hd_cols = spec(read_csv(nhw_v1_path, guess_max = 10000))
aa_v1.raw = read_csv(aa_v1_path, col_types = hd_cols,  na = c("", "NA", "9999", "-9999", "
  janitor::clean_names()
ma_v1.raw = read_csv(ma_v1_path, col_types = hd_cols,  na = c("", "NA", "9999", "-9999", "
  janitor::clean_names()
nhw_v1.raw = read_csv(nhw_v1_path, col_types = hd_cols,  na = c("", "NA", "9999", "-9999",
  janitor::clean_names()

habshd.raw <- bind_rows(
    aa_v1.raw, ma_v1.raw, nhw_v1.raw
  ) %>%
  mutate(
    id_race_white = as.factor(id_race_white),
    id_race_black = as.factor(id_race_black),
    id_race_indian_alaska = as.factor(id_race_indian_alaska),
    id_race_asian = as.factor(id_race_asian),
    id_race_japanese = as.factor(id_race_japanese),
    id_race_korean = as.factor(id_race_korean),
    id_race_vietnamese = as.factor(id_race_vietnamese),
    id_race_native_hawaiian = as.factor(id_race_native_hawaiian),
    id_race_guam_chamorro = as.factor(id_race_guam_chamorro),
    id_race_samoan = as.factor(id_race_samoan),
    id_race_other_pacific = as.factor(id_race_other_pacific),
    id_race_other = as.factor(id_race_other),
    id_hispanic = as.factor(id_hispanic),
    id_hispanic_other = as.factor(id_hispanic_other),
    race = case_when(
      id_hispanic != 1 ~ "Hispanic",
      id_race_white == 1 & id_hispanic != 2 ~ "NHW",
      id_race_black == 1 ~ "Black",
      TRUE ~ "Other")
  )
```

```
habshd <- habshd.raw %>%
  mutate(
    abeta40 = ifelse(is.na(r3_qtx_plasma_abeta42), r5_qtx_plasma_abeta40, r3_qtx_plasma_ab
    abeta42 = ifelse(is.na(r3_qtx_plasma_abeta42), r5_qtx_plasma_abeta42, r3_qtx_plasma_ab
    ptau181 = ifelse(is.na(r3_qtx_plasma_p_tau181), r5_qtx_plasma_p_tau181, r3_qtx_plasma_
    total_tau = ifelse(is.na(r3_qtx_plasma_total_tau), r5_qtx_plasma_total_tau, r3_qtx_pla
    nfl = ifelse(is.na(r3_qtx_plasma_nf_l), r5_qtx_plasma_nf_l, r3_qtx_plasma_nf_l)
  ) %>%
  select(med_id, age, id_gender, interview_language, adi_state_rank, race,
         id_education, smoke_ever, cdx_cog, cdx_depression,cdx_hypertension,
         cdx_diabetes, cdx_dyslipidemia, cdr_sum,
         om_bp1_dia, om_bp1_sys,
         om_height, om_weight, om_bmi, om_ab_circumference,
         bw_chol_total, bw_ld_lchol, bw_hdl_chol, bw_hba1c, gds_total,
         abeta40, abeta42, ptau181, total_tau, nfl,
         apoe4_snp
         )

write_csv(habshd, "work/habshd_pheno.csv")
```

```
#descriptive table
description <- c("Medical ID","","1 = Female <br> 0 = Male",
                 "Language in which <br> interview was administered",
                 "Area Deprivation Index",'Black, Hispanic, NHW',
                 "Years of Education","Ever Smoked<br> (1:Yes, 0:No)",
                "Cognitive Disorder:<br> 0: Cognitively Unimpaired <br>1:Mild Cognitive
```

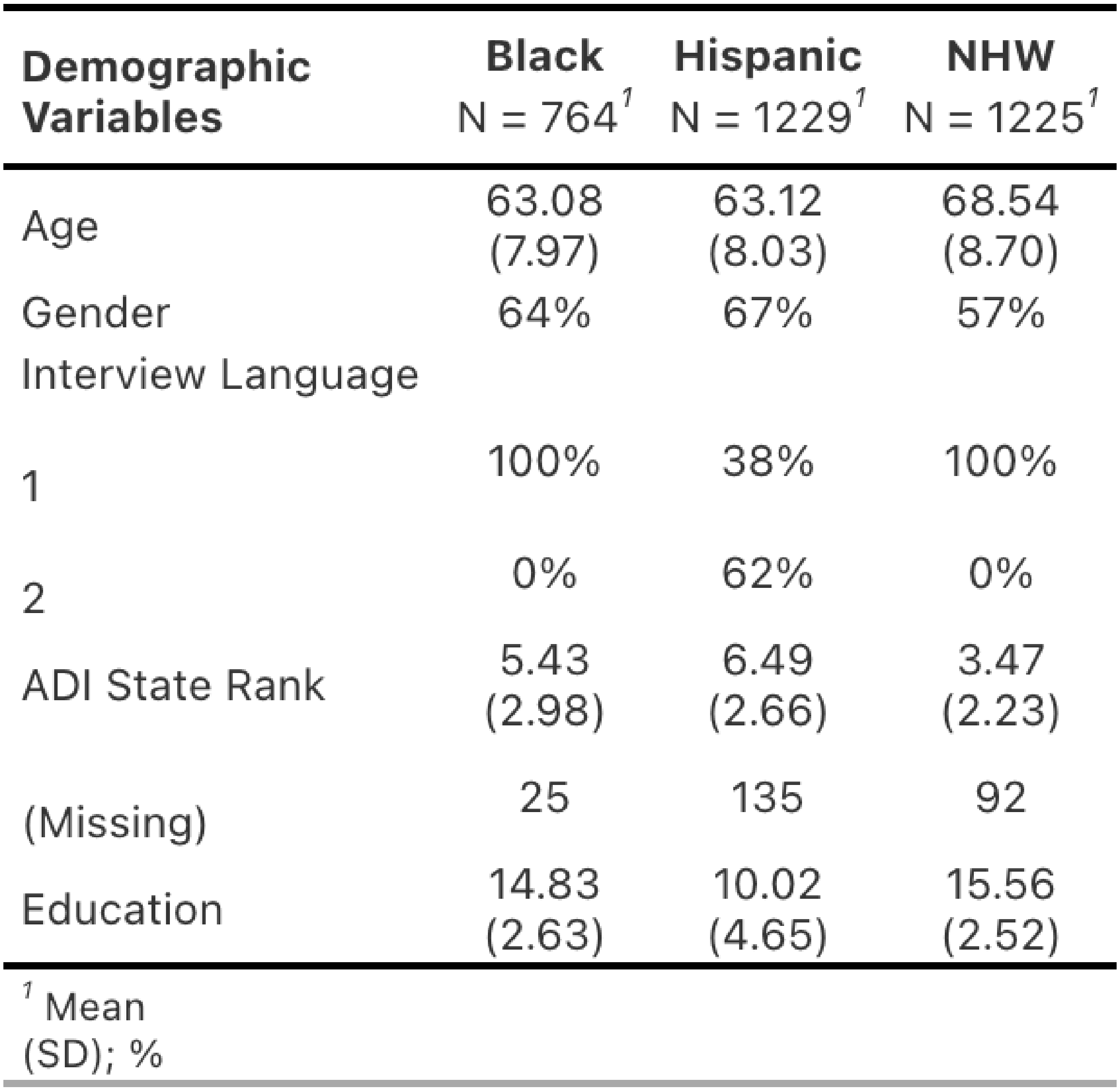
```
                 Impairment<br>2: Dementia", "Depression<br> (1:Yes, 0:No)",
                 "Hypertension <br> (1:Yes,0:No)", "Diabetes <br> (1:Yes,0:No)",
                  "High Cholesterol (1:Yes,0:No)",
                 "Clinical Dementia Rating (CDR):<br> Sum of Boxes",
                 "Diastolic BP", "Systolic BP",  "Height (in)", "Weight (lbs)",
                 "BMI", "Abdominal circumference (in)", "Total Cholesterol",
                "LDL Cholesterol<br> (bad)","HDL Cholesterol<br> (good)",
                "Hemoglobin","Geriatric Depression Scale (GDS)",
                "abeta40","abeta42","ptau181", "total_tau","nfl",
                "APOE Genotype")

table_desc <- data.frame(cbind(names(habshd), description))
table_desc %>% kbl(caption = '', col.names = c("Variable", "Description"),
                   escape = FALSE) %>%
  kable_classic(full_width = FALSE, html_font = "Ariel") %>%
  kable_styling(font_size = 16, position = "center") %>%
  column_spec(1:2, border_left = F, border_right = F) %>%
  pack_rows("Demographics",1,7) %>%
  pack_rows("Clinical",8,25) %>%
  pack_rows("Imaging",26,30) %>%
  pack_rows("Genomics",31,31)
```

Descriptive Table

Variable	Description
Demographics	
med_id	Medical ID number Not an MRN
age	Age (yrs)
id_gender	1:Female 0: Male
interview_language	Language in which interview was administered 1:English 2:Spanish
adi_state_rank	Area Deprivation Index Levels: 1,2,..,10 1: least disadvantaged 10: most disadvantaged
race	Black, NHW, Hispanic
id_education	Years of Education

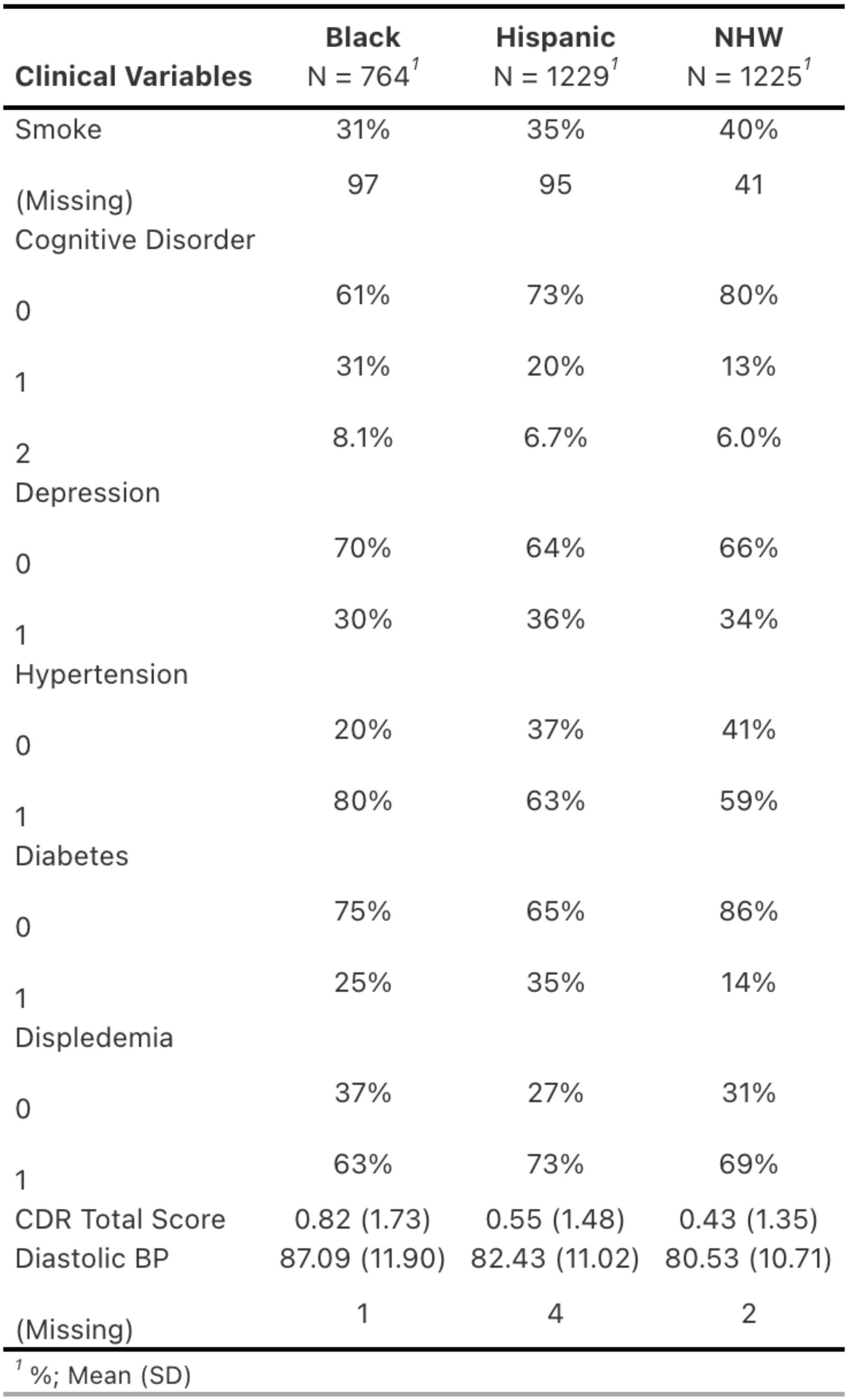
Clinical	
smoke_ever	Ever smoked? 0:No 1:Yes
cdx_cog	Cognitive Disorder 0: Cognitively Unimpaired 1:Mild Cognitive Impairment 2: Dementia
cdx_depression	Depression 0:No 1:Yes
cdx_hypertension	Hypertension 0:No 1:Yes
cdx_diabetes	Diabetes 0:No 1:Yes
cdx_dyslipidemia.	High Cholesterol 0:No 1:Yes
cdr_sum	Clinical Dementia Rating (CDR) Sum of Boxes
gds_total	Geriatric Depression Scale (GDS) sum of GDS 1 to GDS 30
om_bp1_dia	Diastolic BP
om_bp1_sys	Systolic BP
om_height	Height (in)
om_weight	Weight (lbs)
om_bmi	Body Mass Index (BMI)
om_ab_circumference	Abdominal Circumference (in)
bw_chol_total	Total Cholesterol (mg/dL)
bw_ld_lchol	LDL Cholesterol (mg/dL) (bad)
bw_hdl_chol	HDL Cholesterol (mg/dL) (good)
bw_hba1c	Hemoglobin A1C% of total Hgb
Biomarkers	
abeta40	$A\beta_{40}$
abeta42	$A\beta_{42}$
ptau181	Phospho-Tau (pg/mL) Average CV: 0.07065 Avgerage LLOD: 0.016 Average HLOD:349
total_tau	Total Tau
nfl	Neurofilament Light (pg/mL) Average CV: 0.038 Avgerage LLOD: 0.038 Average HLOD:1800
Genetics	
apoe4_snp	APOE Genotype E2E3, E2E4, E3E3, E3E4, E4E4

```
habshd[which(habshd$adi_state_rank=="GQ"),] <- NA
habshd[which(habshd$adi_state_rank=="PH"),] <- NA
habshd[which(habshd$adi_state_rank=="Invalid Address"),] <- NA
habshd[which(habshd$smoke_ever==2),] <- NA
habshd$adi_state_rank <- as.integer(habshd$adi_state_rank)

theme_gtsummary_compact()
demographics_table <- habshd %>% select(age,id_gender,interview_language,
                                          adi_state_rank, race, id_education) %>%
                      tbl_summary(., by = race,
                            statistic = list(
                                    all_continuous() ~ "{mean}<br> ({sd})",
                                    all_categorical()~ "{p}%"),
                             digits = all_continuous()~2,
                            label = c(age~"Age", id_gender ~ "Gender",
                                      interview_language ~ "Interview Language",
                                      adi_state_rank~ "ADI State Rank",
                                      id_education~"Education"),
                            missing_text = "(Missing)") %>%
                          modify_header(label = "**Demographic <br> Variables**",
                          all_stat_cols() ~ "**{level}**<br> N = {n}") %>%
                    as_gt() %>%
                  tab_options(column_labels.border.top.color = "black",
                              column_labels.border.bottom.color = "black",
                              table_body.border.bottom.color = "black",
                              table_body.hlines.color = "white",
                              table.font.size = 12,
                              container.width = 500,
                              container.height =500) %>%
                  fmt_markdown(columns = everything())

gtsave(demographics_table,filename = file.path(SAVE_VISUALIZATIONS_PATH, "demographics_sum

#convert to factors
cdx_cols <- names(habshd %>% select(starts_with("cdx_")))
habshd[cdx_cols] <-lapply(habshd[cdx_cols], factor)

clinical_table1 <- habshd %>% select(smoke_ever,cdx_cog,cdx_depression,
                                     cdx_hypertension,cdx_diabetes,
                                     cdx_dyslipidemia,cdr_sum, om_bp1_dia,
                                     race) %>%
                  tbl_summary(., by = race,
                              statistic = list(
                                    all_continuous() ~ "{mean} ({sd})",
                                    all_categorical()~ "{p}%"),
                              digits = all_continuous()~2,
                              label = c(smoke_ever ~ "Smoke ",
                                      cdx_cog ~ "Cognitive Disorder",
                                      cdx_depression ~ "Depression",
                                      cdx_hypertension ~ "Hypertension",
                                      cdx_diabetes ~ "Diabetes",
                                      cdx_dyslipidemia ~ "Displedemia",
                                      cdr_sum ~ "CDR Total Score",
                                      om_bp1_dia ~ "Diastolic BP"),
                 missing_text = "(Missing)") %>%
               modify_header(label = "**Clinical Variables**",
                              all_stat_cols() ~ "**{level}**<br> N = {n}") %>%
              as_gt() %>%
              tab_options(
                      column_labels.border.top.color = "black",
                      column_labels.border.bottom.color = "black",
                      table_body.border.bottom.color = "black",
                      table_body.hlines.color = "white",
                      table.font.size = 12,
                      container.height = 700,
                      container.width = 700) %>%
              fmt_markdown(columns = everything())

gtsave(clinical_table1,filename = file.path(SAVE_VISUALIZATIONS_PATH, "clinical_table1.png

clinical_table2 <- habshd %>% select(om_bp1_sys,om_height,om_weight,
                                     om_bmi,om_ab_circumference,bw_chol_total,
                                     bw_ld_lchol,bw_hdl_chol,race,bw_hba1c,
                                     gds_total,race) %>%
                   tbl_summary(., by = race,
                              statistic = list(
                                    all_continuous() ~ "{mean} ({sd})",
                                    all_categorical()~ "{p}%"),
                              digits = all_continuous()~2,
                              label = c(om_bp1_sys~"Systoliuc BP",
                                      om_height ~ "Height (in)",
                                      om_weight~ "Weight(lbs)",
                                      om_bmi ~"BMI",
                                      om_ab_circumference~ "Abdominal <br>
                                      Circumference (in)",
                                      bw_chol_total ~ "Total Cholesterol",
                                      bw_ld_lchol~ "LDL <br> Cholesterol",
                                      bw_hdl_chol~ "HDL <br> Cholesterol",
                                      bw_hba1c~ "Hemoglobin",
                                      gds_total ~ "GDS Total"),
                             missing_text = "(Missing)") %>%
                 modify_header(label = "**Clinical <br> Variables**",
                              all_stat_cols() ~ "**{level}**<br> N = {n}") %>%
                as_gt() %>%
                tab_options(
                        column_labels.border.top.color = "black",
                        column_labels.border.bottom.color = "black",
                        table_body.border.bottom.color = "black",
                        table_body.hlines.color = "white",
                        table.font.size = 12,
                        container.height = 700,
                        container.width = 700) %>%
                fmt_markdown(columns = everything())
gtsave(clinical_table2,filename = file.path(SAVE_VISUALIZATIONS_PATH, "clinical_table2.png

habshd$apoe4_snp = as.factor(habshd$apoe4_snp)
imaging_genetics_table <- habshd %>% select(abeta40, abeta42,ptau181, total_tau,
                                            nfl, apoe4_snp,race) %>%
                          tbl_summary(., by = race,
                                      statistic = list(
                                            all_continuous() ~ "{mean} ({sd})",
                                            all_categorical()~ "{p}%"),
                                      digits = all_continuous()~2,
                                      label = c(abeta40~"AB40",
                                                abeta42~"AB42",
                                                ptau181 ~ "pTau",
                                                total_tau ~ "Total Tau",
                                                nfl~ "Plasma NFL",
                                                apoe4_snp ~ "APOE4 SNP"),
                                     missing_text = "(Missing)") %>%
                        modify_header(label = "**Imaging & Genetic <br>
                                      Variables**",
                                  all_stat_cols() ~ "**{level}**<br> N = {n}") %>%
                        as_gt() %>%
                        tab_options(
                                column_labels.border.top.color = "black",
                                column_labels.border.bottom.color = "black",
                                table_body.border.bottom.color = "black",
                                table_body.hlines.color = "white",
                                table.font.size = 12,
                                container.width = 700,
                                container.height =700) %>%
                      fmt_markdown(columns = everything())

gtsave(imaging_genetics_table,filename = file.path(SAVE_VISUALIZATIONS_PATH, "imaging_gene
```



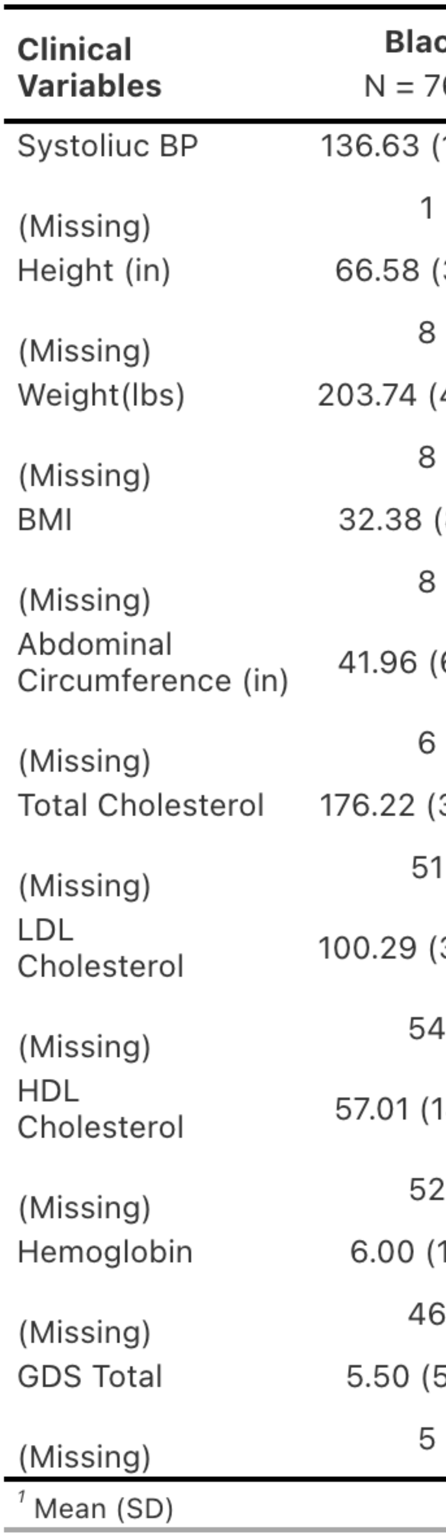
Demographic Variables	Black N = 764[1]	Hispanic N = 1229[1]	NHW N = 1225[1]
Age	63.08 (7.97)	63.12 (8.03)	68.54 (8.70)
Gender	64%	67%	57%
Interview Language			
1	100%	38%	100%
2	0%	62%	0%
ADI State Rank	5.43 (2.98)	6.49 (2.66)	3.47 (2.23)
(Missing)	25	135	92
Education	14.83 (2.63)	10.02 (4.65)	15.56 (2.52)

[1] Mean (SD); %



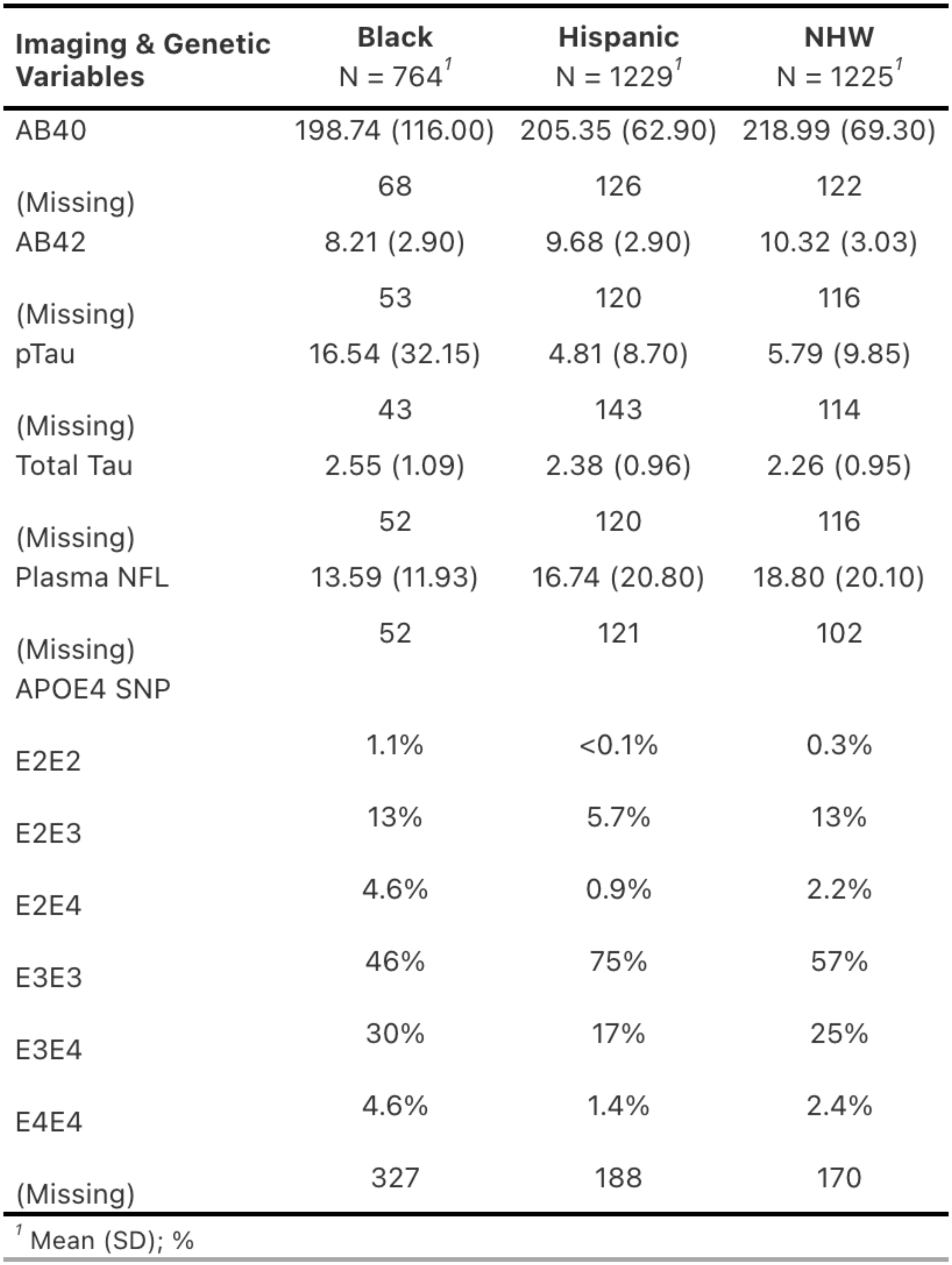
Clinical Variables	**Black** N = 764[1]	**Hispanic** N = 1229[1]	**NHW** N = 1225[1]
Smoke	31%	35%	40%
(Missing)	97	95	41
Cognitive Disorder			
0	61%	73%	80%
1	31%	20%	13%
2	8.1%	6.7%	6.0%
Depression			
0	70%	64%	66%
1	30%	36%	34%
Hypertension			
0	20%	37%	41%
1	80%	63%	59%
Diabetes			
0	75%	65%	86%
1	25%	35%	14%
Displedemia			
0	37%	27%	31%
1	63%	73%	69%
CDR Total Score	0.82 (1.73)	0.55 (1.48)	0.43 (1.35)
Diastolic BP	87.09 (11.90)	82.43 (11.02)	80.53 (10.71)
(Missing)	1	4	2

[1] %; Mean (SD)



Clinical Variables	**Blac** N = 7
Systoliuc BP	136.63 (
(Missing)	1
Height (in)	66.58 (
(Missing)	8
Weight(lbs)	203.74 (
(Missing)	8
BMI	32.38 (
(Missing)	8
Abdominal Circumference (in)	41.96 (
(Missing)	6
Total Cholesterol	176.22 (
(Missing)	51
LDL Cholesterol	100.29 (
(Missing)	54
HDL Cholesterol	57.01 (
(Missing)	52
Hemoglobin	6.00 (
(Missing)	46
GDS Total	5.50 (
(Missing)	5

[1] Mean (SD)



Imaging & Genetic Variables	Black N = 764[1]	Hispanic N = 1229[1]	NHW N = 1225[1]
AB40	198.74 (116.00)	205.35 (62.90)	218.99 (69.30)
(Missing)	68	126	122
AB42	8.21 (2.90)	9.68 (2.90)	10.32 (3.03)
(Missing)	53	120	116
pTau	16.54 (32.15)	4.81 (8.70)	5.79 (9.85)
(Missing)	43	143	114
Total Tau	2.55 (1.09)	2.38 (0.96)	2.26 (0.95)
(Missing)	52	120	116
Plasma NFL	13.59 (11.93)	16.74 (20.80)	18.80 (20.10)
(Missing)	52	121	102
APOE4 SNP			
E2E2	1.1%	<0.1%	0.3%
E2E3	13%	5.7%	13%
E2E4	4.6%	0.9%	2.2%
E3E3	46%	75%	57%
E3E4	30%	17%	25%
E4E4	4.6%	1.4%	2.4%
(Missing)	327	188	170

[1] Mean (SD); %

Genotyping

Samples in HABS-HD were genotyped on the Illumina GSA array. These files have undergone basic variant and sample QC and then imputed on using the TOPMed imputation server. I have then filtered the imputed files to only HapMap III SNPS to make

HapMap III

Download the hapmap_3.3.hg38.vcf.gz file from the Broad's google bucket

```
bcftools view -i 'AF > 0 && TYPE="snp" && N_ALT=1' resources/genetic_epi/resources_broad_h
bcftools view -H > work/hapmap3_snps.txt
```

```
hm3.raw <- read_table("work/hapmap3_snps.txt", col_names = F)

hm3 <- hm3.raw %>%
  mutate(
    cpra = glue::glue("{X1}:{X2}:{X4}:{X5}"),
    X1 = as.numeric(str_replace(X1, 'chr', ''))
  ) %>%
  filter(!is.na(X1)) %>%
  rename(chr = X1, pos = X2, rsid = X3, ref = X4, alt = X5) %>%
  select(-X6)

out <- hm3 %>%
  distinct(cpra, .keep_all = T) %>%
  distinct(rsid, .keep_all = T)

out %>%
  select(cpra) %>%
  write_tsv(., 'work/hm3_extract.txt', col_names = F)

out %>%
  select(cpra, rsid) %>%
  write_tsv(., 'work/hm3_crpa_rsid.txt', col_names = F)
```

```
plink \
  --bfile resources/HABSHD/genotypes/all \
  --keep-allele-order \
  --extract work/hm3_extract.txt \
  --make-bed \
  --out work/habshd_hm3

plink \
  --bfile work/habshd_hm3 \
  --keep-allele-order \
  --update-name work/hm3_crpa_rsid.txt \
  --make-bed \
  --out work/habshd_rsid
```

Summary Statistics

```r
library(tidyverse)    # Data wrangling
library(ggman)
setwd('~/gitcode/IntroGeneticEpi')

# Define column types for summary statistics
coltypes = cols(
  ID = col_character(),
  CHROM = col_double(),
  POS = col_double(),
  REF = col_character(),
  ALT = col_character(),
  AF = col_double(),
  TRAIT = col_character(),
  BETA = col_double(),
  SE = col_double(),
  Z = col_double(),
  P = col_double(),
  N = col_double(),
  OR = col_double(),
  OR_L95 = col_double(),
  OR_U95 = col_double(),
  DIR = col_character(),
  G1000_ID = col_character(),
  G1000_VARIANT = col_character(),
  DBSNP_ID = col_character(),
  DBSNP_VARIANT = col_character(),
  OLD_ID = col_character(),
  OLD_VARIANT = col_character()
)

hm3 <- read_tsv('resources/genetic_epi/ld_ref/w_hm3.snplist')
```

Lipids

Willer et al 2013

A GWAS of low-density lipoprotein (LDL) cholesterol, high-density lipoprotein (HDL) cholesterol, triglycerides and total cholesterol levels conducted in 188,577 individuals that identified 157 loci were associated with lipid levels.

Willer, C. J. et al. Discovery and refinement of loci associated with lipid levels. Nat Genet 45, 1274–83 (2013).

Summary statistics

1. LDL Cholesterol
2. HDL Cholesterol
3. Triglycerides
4. Total cholesterol

```
ldl_path = "resources/genetic_epi/summary_statistics/Willer2013ldl.chrall.CPRA_b37.tsv.gz"
ldl_ss <- read_tsv(ldl_path, comment = "##", col_types = coltypes,
                        col_select = c(DBSNP_ID, CHROM, POS, REF, ALT, AF, BETA, SE, Z, P,

# Filter on p < 0.5 to reduce compute
Willer2013ldl.man <- ggman(filter(ldl_ss, P < 0.05 & P > 1e-100),
                           snp = "DBSNP_ID", bp = "POS", chrom = "CHROM", pvalue = "P", re
                           title = "LDL Cholesterol - Willer 2013") +
  theme_classic()

ggsave('results/figures/Willer2013ldl_ggman.png', plot = Willer2013ldl.man, units = 'in',

# HM3
willer_hm3 <- ldl_ss %>%
  semi_join(hm3, by = c('DBSNP_ID' = 'SNP'))

write_tsv(willer_hm3, 'work/summary_statistics/Willer2013ldl_hm3.tsv.gz')
```

Graham et al. 2021

A GWAS of low-density lipoprotein (LDL) cholesterol, high-density lipoprotein (HDL) cholesterol, triglycerides and total cholesterol levels conducted using a multi-ancestry, genome-wide genetic discovery meta-analysis of lipid levels in approximately 1.65 million individuals, including 350,000 of non-European ancestries that found 773 lipid-associated genomic regions

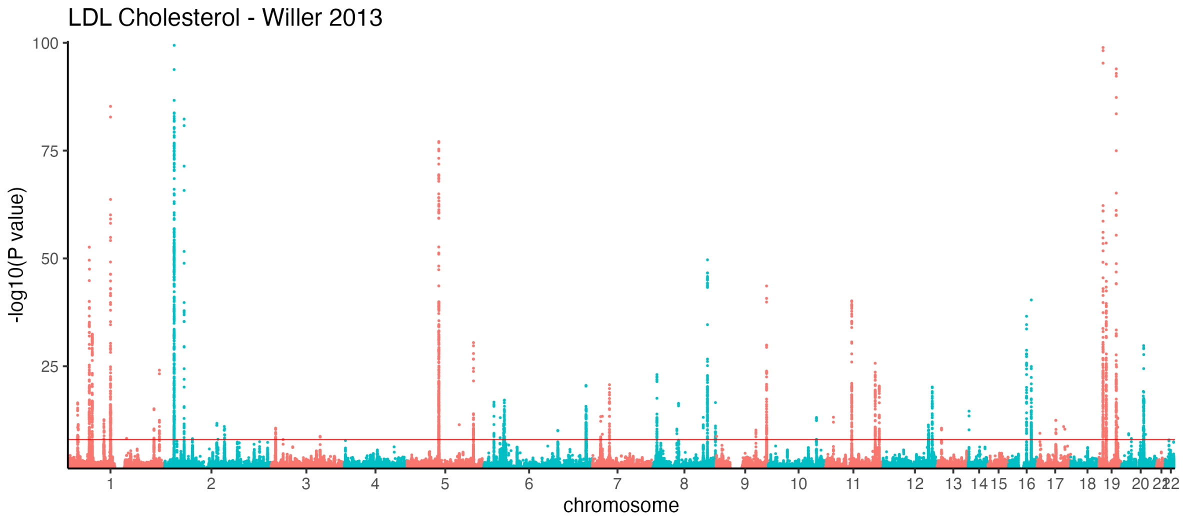


Figure 1: Willer2013ldl_ggman

that contained 1,765 distinct index variants that reached genome-wide significance for at least 1 ancestry group and lipid trait

Graham, S. E. et al. The power of genetic diversity in genome-wide association studies of lipids. Nature 600, 675–679 (2021).

Summary Statistics

1. LDL Cholesterol (EUR)
2. HDL Cholesterol (EUR)

```
# Manhattan Plot
Graham_ldl_path = "resources/genetic_epi/summary_statistics/Graham2021ldl.chrall.CPRA_b37.
Graham_ldl_ss <- read_tsv(Graham_ldl_path, comment = "##", col_types = coltypes,
                         col_select = c(DBSNP_ID, CHROM, POS, REF, ALT, AF, BETA, SE, Z, P,

# Filter on p < 0.5 to reduce compute
Graham2021ldl.man <- ggman(filter(Graham_ldl_ss, P < 0.05 & P > 1e-100),
                          snp = "DBSNP_ID", bp = "POS", chrom = "CHROM", pvalue = "P", re
                          title = "LDL Cholesterol - Graham 2021") +
  theme_classic()

ggsave('results/figures/Graham2021ldl_ggman.png', plot = Graham2021ldl.man, units = 'in',

# HM3
```

```
Graham_hm3 <- Graham_ldl_ss %>%
  semi_join(hm3, by = c('DBSNP_ID' = 'SNP'))

write_tsv(Graham_hm3, 'work/summary_statistics/Graham2021ldl_hm3.tsv.gz')
```

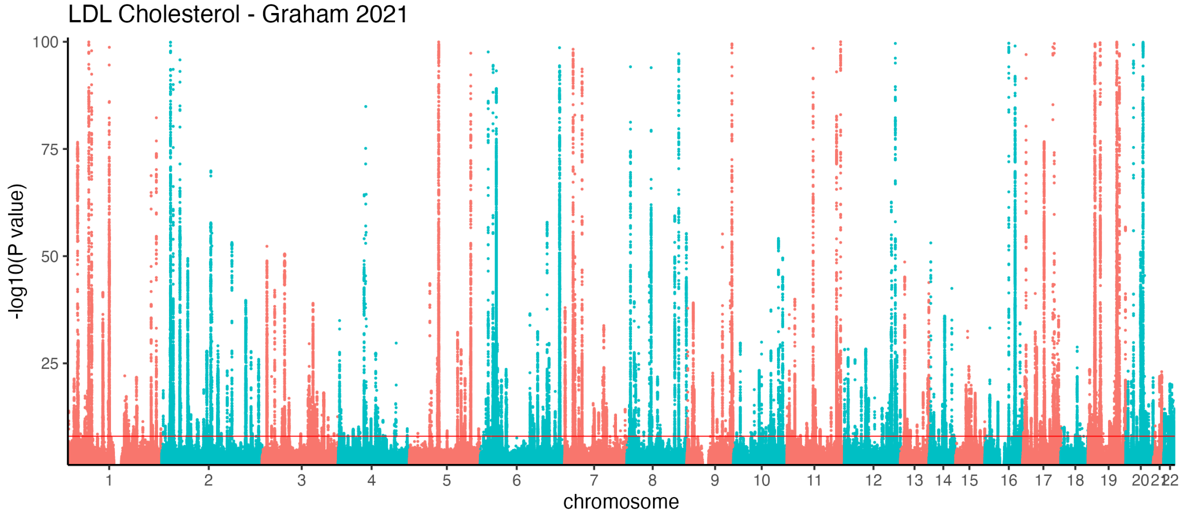


Figure 2: Graham2021ldl_ggman

Alzheimer's disease

Kunkle 2019

A GWAS of Alzheimer's disease conducted in 94,437 indivudles by the International Genomics Alzheiemr's Project that identified 20 genome-wide significant loci.

Kunkle, B. W. et al. Genetic meta-analysis of diagnosed Alzheimer's disease identifies new risk loci and implicates A , tau, immunity and lipid processing. Nat Genet 51, 414–430 (2019).

Summary statistics

1. Late-onset Alzheimer's disease (LOAD)

```
AD_path = "resources/genetic_epi/summary_statistics/Kunkle2019load_stage123.chrall.CPRA_b3
AD_ss <- read_tsv(AD_path, comment = "##",  col_types = coltypes,
                  col_select = c(DBSNP_ID, CHROM, POS, REF, ALT, AF, BETA, SE, Z, P,
# Filter on p < 0.5 to reduce compute
kunkle2019load.man <- ggman(filter(AD_ss, P < 0.05 & P > 1e-100),
                           snp = "DBSNP_ID", bp = "POS", chrom = "CHROM", pvalue = "P", r
                          title = "AD - Kunkle 2019") +
  theme_classic()

ggsave('results/figures/Kunkle2019load_ggman.png', plot = kunkle2019load.man, units = 'in'

# HM3
Kunkle_hm3 <- AD_ss %>%
  semi_join(hm3, by = c('DBSNP_ID' = 'SNP'))

write_tsv(Kunkle_hm3, 'work/summary_statistics/Kunkle2019load_hm3.tsv.gz')
```

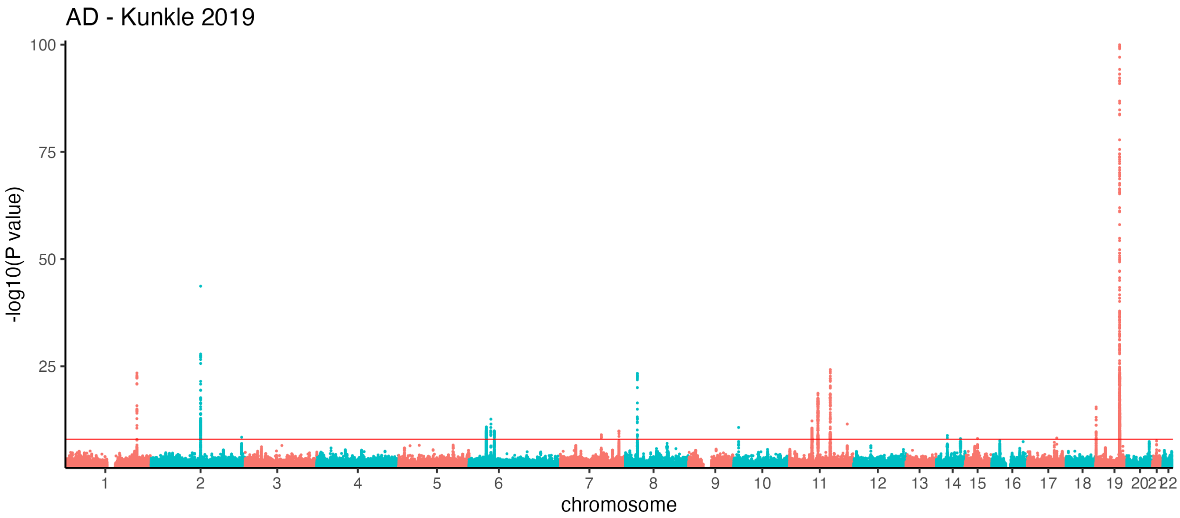


Figure 3: Kunkle2019load_ggman

Bellenguez 2022

A GWAS of Alzheimer's disease and related dementias conducted using 111,326 clinically diagnosed/'proxy' AD cases and 677,663 controls that identified 75 risk loci, of which 42 were novel.

Bellenguez, C. et al. New insights into the genetic etiology of Alzheimer's disease and related dementias. Nat Genet 54, 412–436 (2022).

Summary statistics

1. Alzheimer's disease and related dementias (ADRD)

```
ADRD_path = "resources/genetic_epi/summary_statistics/Bellenguez2022load.chrall.CPRA_b37.t
ADRD_ss <- read_tsv(ADRD_path, comment = "##",  col_types = coltypes,
                      col_select = c(DBSNP_ID, CHROM, POS, REF, ALT, AF, BETA, SE, Z, P,

# Filter on p < 0.5 to reduce compute
adard.man <- ggman(filter(ADRD_ss, P < 0.05 & P > 1e-100),
                   snp = "DBSNP_ID", bp = "POS", chrom = "CHROM", pvalue = "P", relative.p
                   title = "ADRD - Bellenguez 2022"
                   ) +
  theme_classic()

ggsave('results/figures/Bellenguez2022load_ggman.png', plot = adard.man, units = 'in', wid

# HM3
ADRD_hm3 <- ADRD_ss %>%
  semi_join(hm3, by = c('DBSNP_ID' = 'SNP'))

write_tsv(ADRD_hm3, 'work/summary_statistics/Bellenguez2022load_hm3.tsv.gz')
```

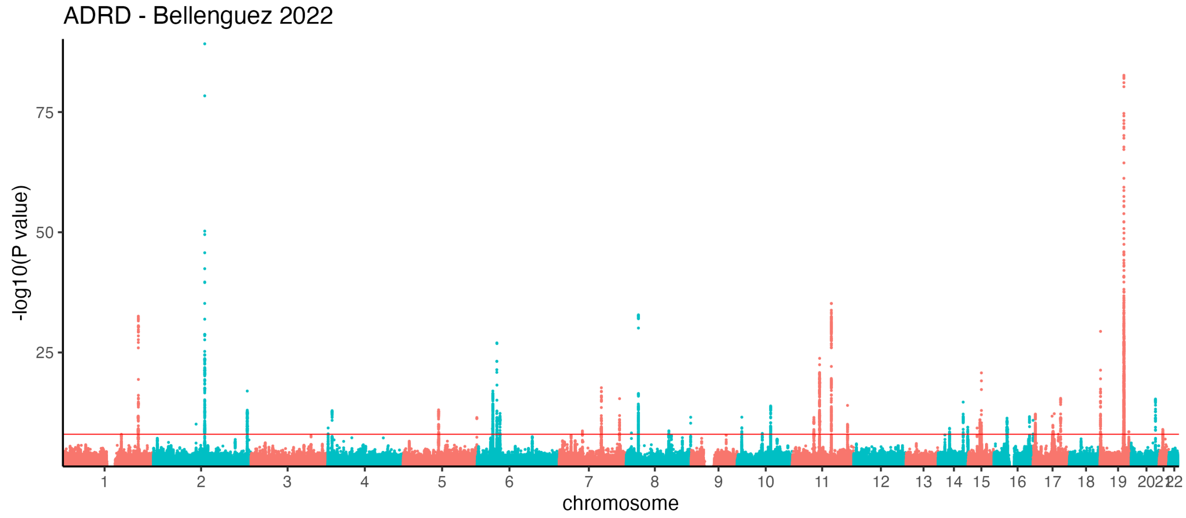


Figure 4: Bellenguez2022load_ggman

Educational Attainment

Lee et al 2018

A large-scale genetic association analysis of educational attainment in a sample of approximately 1.1 million individuals and identify 1,271 independent genome-wide-significant SNPs

Lee, J. J. et al. Gene discovery and polygenic prediction from a genome-wide association study of educational attainment in 1.1 million individuals. Nat Genet 50, 1112–1121 (2018).

Summary statistics

1. Years of Education

```
educ_path = "resources/genetic_epi/summary_statistics/Lee2018educ.chrall.CPRA_b37.tsv.gz"
educ_ss <- read_tsv(educ_path, comment = "##",  col_types = coltypes,
                       col_select = c(DBSNP_ID, CHROM, POS, REF, ALT, AF, BETA, SE, Z, P,

# Filter on p < 0.5 to reduce compute
educ.man <- ggman(filter(educ_ss, P < 0.05 & P > 1e-100),
                   snp = "DBSNP_ID", bp = "POS", chrom = "CHROM", pvalue = "P", relative.p
                   title = "Education - Lee 2018"
                   ) +
  theme_classic()

ggsave('results/figures/Lee2018educ_ggman.png', plot = educ.man, units = 'in', width = 9,

# HM3
educ_hm3 <- educ_ss %>%
  semi_join(hm3, by = c('DBSNP_ID' = 'SNP'))

write_tsv(educ_hm3, 'work/summary_statistics/Lee2018educ_hm3.tsv.gz')
```

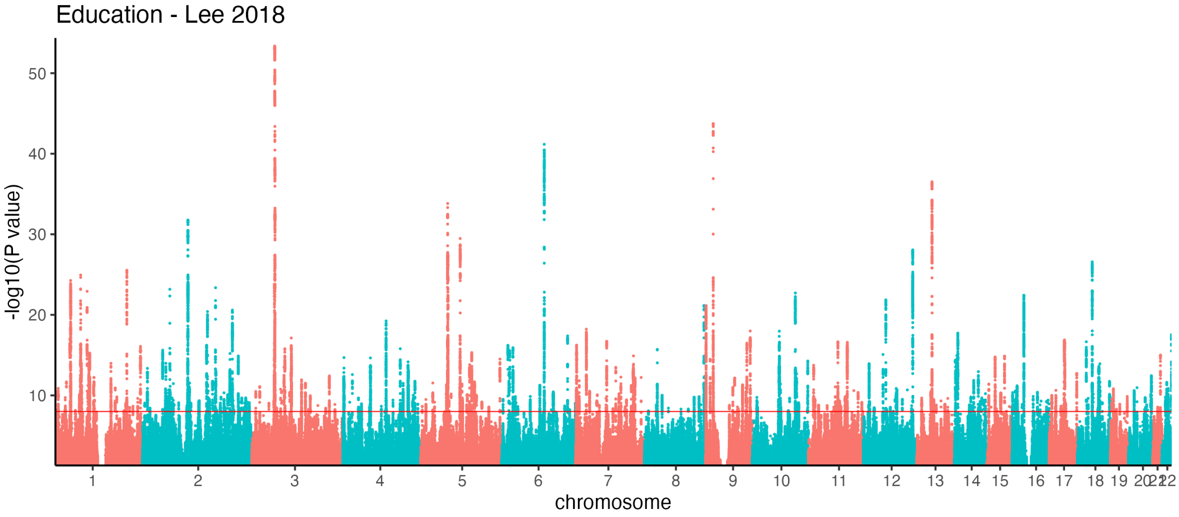


Figure 5: Lee2018educ_ggman

Part II

Genome-wide Association Studies

GWAS QC

TBD

SNP QC

SNP level QC consists of removing markers with excessive missingness or low allele frequency. This QC increases the power to identify true associations with disease risk by removing sub-optimal markers that can increase false positives.

Call Rate & Allele frequency

95% was used as the SNP call rate threshhold (usually 95% or higher), and 1% was used as the MAF threshold (usually 1% or higher).

Filtering SNPs on MAF and call rate can be done in `PLINK 1.9` by typing the following (or similar) at the shell prompt. This uses 95% and 1% for the call-rate and MAF, respectively:

```
# Generate frequency reports
plink \
    --bfile work/habshd_rsid \
    --keep-allele-order \
    --freq \
    --out work/habshd_snpqc

plink \
    --bfile work/habshd_rsid \
    --keep-allele-order \
    --freqx \
    --out work/habshd_snpqc

# Filter on call rate and maf
plink \
    --bfile work/habshd_rsid \
    --keep-allele-order \
    --geno 0.05 --maf 0.01 \
    --make-bed --out work/habshd_snpqc
```

```
## ==== SNP Level Filtering ====
# ---- readin plink .frq ---- ##
message("reading plink frq file")
freq.raw <- read_table('work/habshd_snpqc.frq', col_names = T,
  col_types = cols(
  CHR = col_double(),
  SNP =col_character(),
  A1 = col_character(),
  A2 = col_character(),
  MAF = col_double(),
  NCHROBS = col_double()
))

# ---- readin plink .frqx ---- ##
message("reading plink frqx file")
freqx.raw <- read_tsv('work/habshd_snpqc.frqx', col_names = T,
  col_types = cols(
  CHR = col_double(),
  SNP = col_character(),
  A1 = col_character(),
  A2 = col_character(),
  `C(HOM A1)` = col_double(),
  `C(HET)` = col_double(),
  `C(HOM A2)` = col_double(),
  `C(HAP A1)` = col_double(),
  `C(HAP A2)` = col_double(),
  `C(MISSING)` = col_double()
))

# ---- SNP level statisitcs ----
snps <- freq.raw %>%
  full_join(freqx.raw, by = c("CHR", "SNP", "A1", "A2")) %>%
  rename(AA = `C(HOM A1)`, AB = `C(HET)`, BB = `C(HOM A2)`, missing = `C(MISSING)`) %>%
  mutate(Call.rate = 1 - (missing / c(AA + AB + BB + missing))) %>%
  mutate(Call = Call.rate >= 1 - 0.05) %>%
  mutate(Call.maf = MAF < 0.01)
```

Figure @ref(fig:MAFxcallrate) shows the SNP call rate versus minor allele frequncy across all typed SNPs in the study. The dashed lines denote the MAF and call rate QC thresholds. xxx

SNPs were removed due to low call rate and xxx SNPs were removed due to low minor allele frequency.

```
MAFxcallrate.p <- ggplot(data = snps, aes(x = MAF, y = Call.rate)) +
  geom_point(alpha = 0.3, size = 0.5) +
  geom_hline(yintercept = 1 - 0.05, linetype = 2, colour = 'red') +
  geom_vline(xintercept = 0.01, linetype = 2, colour = 'red') +
  scale_x_log10(breaks = scales::trans_breaks("log10", function(x) round(10^x, 3))) +
  labs(y = 'Proportion of called genotypes', x = 'Minor Allele Frequency (log)') +
  theme_bw() + annotation_logticks()

ggsave('results/plots/MAFxcallrate.png', plot = MAFxcallrate.p, height = 4, width = 6, uni
```

Hardy Weinberg Equilibrium

Violations of Hardy Weinberg Equilibrium can indicate either the presence of population substructure, or the occurence of genotyping error. It is common practice to assume that violoations are indicative of genotyping error and remove SNPs in which the HWE test statistic has a corresponding p-value of less then 1x10-6. A threshold of xxx is used here.

For case-control data, HWE is generally not tested in cases to not exclude real selection against a phenotype, so it is best to include case-control status in the PLINK files.

Filtering SNPs on Hardy Weinberg Equilibrium for autosomes only can be done in PLINK by typing the following at the shell prompt:

```
plink \
    --bfile work/habshd_snpqc  \
    --keep-allele-order \
    --autosome \
    --hardy \
    --hwe 0.000001 \
    --make-bed --out work/habshd_hwe
```

```
# ---- readin plink .hwe ---- ##
message("reading plink hwe file")
hwe.raw <- read_table2('work/habshd_hwe.hwe', col_types = cols(
  CHR = col_integer(),
  SNP = col_character(),
  TEST = col_character(),
  A1 = col_character(),
  A2 = col_character(),
  GENO = col_character(),
  `O(HET)` = col_double(),
  `E(HET)` = col_double(),
  P = col_double()
))

snps <- snps %>%
  full_join(hwe.raw, by = c("CHR", "SNP", "A1", "A2")) %>%
  mutate(hwe = P > 0.000001) %>%
  as_tibble()

suppressPackageStartupMessages(library(ggtern))

hweplot <- snps %>%
  filter(!is.na(P)) %>%
  mutate(alph = ifelse(hwe, 0.2, 0.8),
         hwe = ifelse(hwe, "Pass", "Fail")) %>%
  ggtern::ggtern(aes(x = AA, y = AB, z = BB, colour = hwe, alpha = alph)) +
   geom_point(size = 0.5)  +
   scale_colour_manual(name= 'Hardy Weinberg \n Equilibrium',
                       values = c(Pass = "#377EB8", Fail = "#E41A1C")) +
   scale_alpha_continuous(guide = "none", range = c(0.8, 0.2)) +
   theme_bw() + theme(legend.position = 'bottom')

hweplot

detach("package:ggtern", unload=TRUE)

ggsave('results/plots/hweplot.png', plot = hweplot, height = 4, width = 6, units = 'in')
```

Sample QC

Call Rate

A low genotyping call rate in a sample can be indicative of poor DNA sample quality, so samples with a call rate < xxx% are excluded from further analysis.

Filtering samples on a call rate of 95% can be done in PLINK by typing the following at the shell prompt:

```
plink \
    --bfile work/habshd_hwe \
    --keep-allele-order \
    --mind 0.05 \
    --make-bed --out work/habshd_sampleQC
```

Sex Discordance

Samples with discordance between self-reported and genetically predicted sex likely have errors in sample handling, such as sample swaps. Predicted sex can be determined by calculating X chromosome heterozygosity using an F test, because biological men have one X chromosome and women have two. An F value of ~0.99 indicates males, and an F value of ~0.03 indicates females. Furthermore, checking X chromosome heterozygosity may reveal sex chromosome anomalies (~0.28 in reported females; ~0.35 in males).

Since sex discordance may be due to sample swaps or to incorrect phenotyping, sex discordant samples should generally be removed unless a swap can be reliably resolved.

Identification of individuals with discordent sex can be done in PLINK 1.9 by typing the following at the shell prompt, which will produce a list of individuals with discordent sex data.

```
plink \
    --bfile resources/HABSHD/genotypes/HABLE_GSA_20230418a_FINAL  \
    --check-sex --out work/HABLE_GSA_20230418a

plink \
    --bfile resources/HABSHD/genotypes/HABLE_GSA_20220602_FINAL  \
    --check-sex --out work/HABLE_GSA_20220602

awk 'FNR==1 && NR==1 {print; next} FNR>1 {print}' work/HABLE_GSA_20220602.sexcheck work/HA
```

```
## ---- Read in Data ----##
sexcheck.raw <- read_table('work/habshd_sexcheck.txt')

##  recode sex varibles
sexcheck <- sexcheck.raw %>%
  mutate(PEDSEX = recode(PEDSEX, '2' = 'Female', '1' = 'Male'))

##  Exclude samples with no sex inconsistencies
sex_exclude.samples <- sexcheck %>%
  filter(STATUS == 'PROBLEM') %>%
  mutate(PEDSEX = recode(PEDSEX, '2' = 'Female', '1' = 'Male'))
```

The following plot (Fig. @ref(fig:sexplot)) displays the X Chromosome heterozygosity for self reported sex, with samples with problems highlighted in red. Table @ref(tab:sextab) displays the individule records that should be excluded from further downstream analysis.

```
sexcheck.p <- ggplot(data = sexcheck, aes(x = as.factor(PEDSEX), y = F, colour = STATUS, s
  geom_jitter() +
  scale_color_manual(values = c( "#377EB8", "#E41A1C")) +
  theme_bw() + labs(x = 'Self reported sex', y = 'X CHR Heterozygocity (F)') + theme(legen

ggsave('results/plots/sexcheck.png', plot = sexcheck.p, height = 4, width = 6, units = 'in
```

Pruning

Pruning is typically done to remove linkage disequilibrium (LD) between SNPs, which is often a necessary step in various genetic analyses to ensure the independence of markers and is necessary for estimating heterozygosity, realtedness, and population stratification.

```
plink \
  --bfile work/habshd_sampleQC \
  --indep-pairwise 50 5 0.2 \
  --out work/indepSNP
```

Heterozygosity check

Insufficient heterozygosity can indicate inbreeding or other family substructures, while excessive heterozygosity may indicate poor sample quality.

Individuals with outlying heterozygosity rates can be identified in PLINK 1.9 by typing the following command at the shell prompt:

```
plink \
    --bfile work/habshd_sampleQC  \
    --extract work/indepSNP.prune.in \
    --het --out work/habshd
```

This produces a file containing Method-of-moments F coefficient estimates, which can be used to calculate the observed heterozygosity rate in each individual. Analysis is performed using an LD pruned snplist.

We calculate a heterozygocity similarly using observed and expected counts from the PLINK output [(Observed - Expected)/N) and exclude samples that are ± 3 sd from the cohort mean.

```
## ---- Read in Data ----##
het.raw <- read_table('work/habshd.het')

## caluclate heterozygosity
het <- het.raw %>%
  rename(O = `O(HOM)`, E = `E(HOM)`, N = `N(NM)`) %>%
  mutate(Het = (N - O) / N)

##  Calculate exclusion thresholds
upper.het <- mean(het$Het) + sd(het$Het)*3
lower.het <- mean(het$Het) - sd(het$Het)*3

##  Exclusion of samples
het <- het %>%
  mutate(exclude = ifelse(Het >= upper.het | Het <= lower.het, TRUE, FALSE))

het_exclude.samples <- het %>% filter(exclude == TRUE)
```

Figure @ref(fig:plothet) displays the distrubution of heterozygosity in xxx. Samples with excessive (Het > xxx) or deficient (Het < xxx) heterozygosity are colored red. Table @ref(tab:het) displays samples that are to be excluded.

```
heterozygosity.p <- ggplot(het, aes(x = Het, fill = exclude)) + geom_histogram(binwidth =
  geom_vline(xintercept = upper.het, colour = 'red', linetype = 2) +
  geom_vline(xintercept = lower.het, colour = 'red', linetype = 2) +
  theme_bw() + scale_fill_manual(values = c("#377EB8", "#E41A1C")) +
  theme(legend.position = 'bottom') +
  labs(x = 'Heterozygosity')

ggsave('results/plots/heterozygosity.png', plot = heterozygosity.p, height = 4, width = 6,
```

Cryptic Relatedness

Population based cohorts are often limited to unrelated individuals as associations statistics often assume independence across individuals. Closely related samples will share more of their genome and are likely to be more phenotypically similar than than two individuals chosen randomly from the population. A common measure of relatedness is identity by descent (IBD), where a kinship correlation coefficient (pi-hat) greater 0.1 suggests that samples maybe related or duplicates samples.

```
# IBD relationship table
# https://github.com/WheelerLab/GWAS_QC/blob/master/example_pipelines/QC%20Analysis%20-%20

rel_tab <- tibble(relationship = c("unrelated", "identical-twins",
                                   "parent-child", "full-siblings",
                                   "half-siblings", "grandparent-grandchild",
                                   "avuncular", "half-avuncular",
                                   "first-cousin", "half-first-cousin",
                                   "half-sibling-first-cousin"),
  pi_hat = c(0, 1, 0.5, 0.5, 0.25, 0.25, 0.25, 0.125, 0.125, 0.0625, 0.375),
  z0 = c(1, 0, 0, 0.25, 0.5, 0.5, 0.5, 0.75, 0.75, 0.875, 0.375),
  z1 = c(0, 0, 1, 0.5, 0.5, 0.5, 0.5, 0.25, 0.25, 0.125, 0.5),
  z2 = c(0, 1, 0, 0.25, 0, 0, 0, 0, 0, 0, 0.125)
)

dup_relationships <- c("grandparent-grandchild", "avuncular", "half-avuncular")
rel_tab_filt <- rel_tab %>%
  filter(relationship %nin% dup_relationships) %>%
  mutate(relationship = ifelse(relationship == "half-siblings", "2nd degree",
                               ifelse(relationship == "first-cousin",
                                      "3rd degree", relationship)))
```

Identifying duplicated or related samples can be done in PLINK 1.9 by typing the following command at the shell prompt.

```
plink \
    --bfile work/habshd_sampleQC \
    --extract work/indepSNP.prune.in \
    --genome --min 0.05 --out work/habshd.ibd
```

```
# select samples with kinship cofficents > 0.1875
# https://link.springer.com/protocol/10.1007/978-1-60327-367-1_19
pi_hat_thres = 0.1875

# Find closest match
closest <- function(vals, ref) {
  fc <- Vectorize(function(x) {
    ref[which.min(abs(ref - x))]
  }) #finds closest
  fc(vals)
}

# Iteratively Remove related samples
remove_samples <- function(ibdcoeff, fam, msg = "closely related to") {
  fam_fi <- fam %>%
    mutate(FI = paste0(FID, "_-_-tempsep-_-_", IID)) %>%
    mutate(status = ifelse(status > 2, 0.5, status))

  ibdcoeff %<>%
    mutate(FI1 = paste0(FID1, "_-_-tempsep-_-_", IID1),
           FI2 = paste0(FID2, "_-_-tempsep-_-_", IID2))
  related_samples <- NULL
  excluded <- c()
  fam_table <- tibble(FID = c("deleteme"),
                      IID = c("deleteme"),
                      Related = c("deleteme"))
  while (nrow(ibdcoeff) > 0) {
    test_tab <- plyr:::count(c(ibdcoeff$FI1, ibdcoeff$FI2))
    if (!("x" %in% names(test_tab))) {
      print(ibdcoeff)
    }
    sample.counts <- plyr:::count(c(ibdcoeff$FI1, ibdcoeff$FI2)) %>%
      as_tibble %>%
      rename(FI = x) %>%
      mutate(FI = as.character(FI)) %>%
      inner_join(fam_fi, by = "FI") %>%
      arrange(desc(qc_failed), status, desc(freq))
    rm.sample <- sample.counts[[1, "FI"]]
    id_ <- str_split(rm.sample, "_-_-tempsep-_-_")[[1]]
    fid <- id_[1]
    iid <- id_[2]
    remtxt <- sprintf("%s %i other samples.",
                      msg,
                      sample.counts[[1, "freq"]])
    message(paste("Removing sample", iid, remtxt))
    ft <- tibble(FID = fid, IID = iid, Related = remtxt)
    fam_table <- fam_table %>%
      bind_rows(ft)
    ibdcoeff <- ibdcoeff[ibdcoeff$FI1 != rm.sample &
```

```
                           ibdcoeff$FI2 != rm.sample, ]
    related_samples <- c(as.character(rm.sample), related_samples)
  }
  return(
    list(related_samples = related_samples,
         fam_table = filter(fam_table, Related != "deleteme"),
         exclude_samples = tibble(FI = as.character(related_samples)) %>%
           separate(FI, c("FID", "IID"), sep = "_-_-tempsep-_-_")))
}


# Import data
fam <- "work/habshd_sampleQC.fam" %>%
  read_table(col_types = "cc---i", col_names = c("FID", "IID", "status")) %>%
  mutate(qc_failed = FALSE)

relatedness.raw = read_table("work/habshd_ibd.genome")

ibdcoeff <- relatedness.raw %>%
  filter(PI_HAT > pi_hat_thres) %>%
  mutate(
    pi_hat = closest(PI_HAT, rel_tab_filt$pi_hat),
    z0 = closest(Z0, rel_tab_filt$z0),
    z1 = closest(Z1, rel_tab_filt$z1),
    z2 = closest(Z2, rel_tab_filt$z2),
  ) %>%
  left_join(rel_tab_filt)

ibdcoeff_unrelated <- remove_samples(ibdcoeff, fam)
```

The following histogram (Fig. @ref(fig:kinplot)) shows the distribution of proportion of IBD sharing (pi-hat in PLINK; PropIBD in KING) between all pairs.

```
ggplot(relatedness.raw, aes(x = PI_HAT)) +
  geom_histogram(binwidth = 0.01, fill = "#377EB8") +
  scale_y_continuous(trans = 'log10', breaks = scales::trans_breaks("log10", function(x) r
  coord_cartesian(xlim = c(min(relatedness.raw$PI_HAT) - 0.05, 1)) +
  annotation_logticks() +
  theme_bw() +
  labs(x = "IBD Sharing (pi-hat in PLINK)") +
  geom_vline(xintercept = pi_hat_thres,
             colour = "red", linetype = 2)

ggsave("results/plots/ibd.png", width = 4, height = 4, units = 'in')
```

The following plot (Fig. @ref(fig:relplot)) shows the xxx by the proportion of loci where individuals share zero alleles (Z0), where the proportion of IBD sharing is greater than 0.05. In family based studies, pairs are colored by IBD relationship. Table @ref(tab:ibdfail) displays the individuals where the kinship coefficient was greater than xxx in population based studies OR how were duplicates in family based studies.

```
ggplot(ibdcoeff, aes(x = Z0, y = Z1, color = relationship)) +
  geom_point() +
  labs(x = 'P(IBD=0)', y = "P(IBD=0)") +
  theme_bw()

ggsave("results/plots/relatedness.png", width = 6, height = 4, units = 'in')
```

Population Substructure

After excluding population outliers from the dataset, population substructure will remain due to the presence of genetic diversity within apparently homogenous populations. Within a single ethnic population, even subtle degrees of population stratification can bias results due to differences in allele frequencies between subpopulations. Principal components based on the observed genotypes in the dataset of interest can be used to capture information on substructure and be included as covariates in downstream analysis.

To obtain the principal components for the sample dataset after population outliers have been removed, type the following PLINK 1.9 commands at the shell prompt to generate the principal component eigenvalue and eigenvector files.

```
plink \
    --bfile work/habshd_sampleQC \
    --extract work/indepSNP.prune.in \
    --pca 10 \
    --out work/habshd
```

```
# PCA file from plink

zscore = function(x){(x - mean(x, na.rm = TRUE)) / sd(x, na.rm = TRUE)}
```

```
# Read in eigenvectors and z-score transform
pca <- read_delim('work/habshd.eigenvec',
                  delim = " ", col_names = c("FID", "IID", paste0("PC", 1:10)),
                  col_types = cols(.default = "d", FID = "c", IID = "d")) %>%
         mutate_at(paste0("PC", 1:10), zscore) %>%
  left_join(habshd %>% select(med_id, race), by = c('IID' = "med_id"))

# read in egienvalues
eigenval.raw <- parse_number(read_lines('work/habshd.eigenval'))

eigenval <- tibble(eigenval = eigenval.raw,
                   PC = 1:length(eigenval.raw)) %>%
              mutate(PVE = round(eigenval / sum(eigenval), 3)) %>%
              select(PC, eigenval, PVE)
```

Scree Plot

The below scree plot (Fig. @ref(fig:ScreePlotStrat)) shows the amount of variation retained by each principal component (Left) and the cumualtive proportion of variance explained by each principal compoent (Right).

```
#Include the number of PC for where the cumualtive PVE is 95%
PC.inc <-  findInterval(0.95, cumsum(eigenval$PVE)) + 1

## ---- Plot scree plot of proportion of varaince explained by Principal components ---- #
p1 <- ggplot(data = eigenval, aes(x = PC, y = PVE, group = factor(1))) +
  geom_point(colour = '#377EB8') + geom_path(colour = '#377EB8') +
  scale_x_continuous(breaks = c(1:10)) +
  labs(x = 'Principal Components') +
  theme_bw() + coord_cartesian(ylim = c(0,1), default = T)


p2 <- ggplot(data = eigenval, aes(x=PC, y=cumsum(PVE), group = factor(1))) +
  geom_point(colour = '#377EB8') + geom_path(colour = '#377EB8') +
  scale_x_continuous(breaks = c(1:10)) +
  labs(x = 'Principal Components', y = 'cumulative PVE') +
  theme_bw() + coord_cartesian(ylim = c(0,1), default = T) +
  geom_hline(yintercept = 0.95, colour = '#E41A1C', linetype = 2)

p3 <- gridExtra::grid.arrange(p1, p2, ncol = 2)
```

```
ggsave("results/plots/screeplot.png", plot = p3, width = 9, height = 4, units = 'in')
```

Population substructure

The following plots show the population structure of xxx based on the first two (Fig. @ref(fig:2PCstrat)) and three (Fig. @ref(fig:3PCstrat))) principal components compared with the reference populations from 1000 Genomes.

```
##  Plot Superpopulations, PC1 + PC2
ggplot(pca, aes(x = PC2, y = PC1, color = race)) +
  geom_point() +
  scale_color_brewer(palette = "Set1") +
  theme_bw() + theme(legend.position = 'right')

ggsave("results/plots/pca.png", width = 6, height = 4, units = 'in')
```

Exclude Samples

```
bind_rows(
  sex_exclude.samples,
  het_exclude.samples,
  ibdcoeff_unrelated$exclude_samples %>% mutate_at(c('FID', 'IID'), as.numeric)
) %>%
  select(FID, IID) %>%
  distinct(FID, IID) %>%
  write_tsv('work/habshd.ExcludeSamples.tsv', col_names = F)
```

```
plink \
    --bfile work/habshd_sampleQC \
    --keep-allele-order \
    --remove work/habshd.ExcludeSamples.tsv \
    --make-bed --out work/habshd_gwas
```

GWAS

TBD

Import

```
library(tidyverse)
```

```
fam <- read_table('work/habshd_gwas.fam', col_names = c('FID', 'IID'))
habshd <- read_csv("work/habshd_pheno.csv") %>% distinct(med_id, .keep_all = T)
pc <- pca <- read_delim('work/habshd.eigenvec',
                  delim = " ", col_names = c("FID", "IID", paste0("PC", 1:10)),
                  col_types = cols(.default = "d", FID = "c", IID = "d"))
```

Phenotype File

```
pheno <- fam %>%
  left_join(select(habshd, med_id, cdr_sum), by = c('IID' = 'med_id'))

pheno %>%
  write_tsv('work/habshd_gwas.pheno', col_names = F)
```

Covariate File

```
covar <- fam %>%
  left_join(select(habshd, med_id, age, id_gender), by = c('IID' = 'med_id')) %>%
  left_join(select(pc, IID, PC1, PC2, PC3, PC4), by = 'IID')

covar %>%
  write_tsv('work/habshd_gwas.covar', col_names = F)
```

GWAS

```
plink \
  --bfile work/habshd_gwas \
  --pheno work/habshd_gwas.pheno \
  --covar work/habshd_gwas.covar \
  --linear hide-covar \
  --out results/habshd_cdr_gwas
```

Manhattan Plot

```
gwas.raw <- read_table('results/habshd_cdr_gwas.assoc.linear') %>%
  select(-X10)

gwas.raw %>% arrange(P)

cdr_gwas.p <- ggman::ggman(gwas.raw, snp = "SNP", bp = "BP", chrom = "CHR", pvalue = "P",
  theme_classic()

ggsave("results/plots/cdr_gwas.png", plot = cdr_gwas.p, width = 9, height = 4, units = 'in
```

GWAS-SS

Polygenic risk scores, Two-sample Mendelian Randomization, and Genetic Correlation methods require the use of summary statistics from genome-wide association studies, including single nucleotide polymorphisms (SNPs), beta coefficients, standard errors, p-values, and allele frequencies. However, the historical lack of standards for data content and file formats in GWAS summary statistics has resulted in heterogeneous data sets. To address this issue, standardizing and harmonizing the GWAS summary statistics is crucial before conducting MR analyses. The GWAS Catalog and OpenGWAS platforms have developed formats such as GWAS-SSF (Hayhurst et al. 2022) and GWAS-VCF (Lyon et al. 2021) to facilitate sharing of GWAS SumStats. Tools like MungeSumstats (Murphy et al 2021) and GWAS2VCF (Lyon et al. 2021) are available that provide rapid standardization and quality control of GWAS SumStats.

AD GWAS

We download the International Genomics of Alzheimer's Project (IGAP) Alzheimer's disease GWAS of Kunkle et al. Nat Genet, 2019. from the GWAS catalouge. These summary statistics correspond to the meta-analysis results obtained in stage 1 including genotyped and imputed data (11,480,632 variants, phase 1 integrated release 3, March 2012) of 21,982 Alzheimer's disease cases and 41,944 cognitively normal controls.

The Summary statistics consists of the following information for each SNP and its association to Alzheimer's disease based on meta-analysis in the publication mentioned below.

- Chromosome: Chromosome of the SNP (Build 37, Assembly Hg19)
- Position: Position of the SNP (Build 37, Assembly Hg19)
- MarkerName: SNP rsID or chromosome:position:I/D if rsID not available. I/D indicates indel or deletion respectively.
- Effect_allele: Reference allele (coded allele)
- Non_Effect_allele: Non reference allele (non coded allele)
- Beta: Overall estimated effect size for the effect allele
- SE: Overall standard error for effect size estimate
- Pvalue: Meta-analysis Pvalue using regression coefficients (beta and standard error)

```
curl https://ftp.ebi.ac.uk/pub/databases/gwas/summary_statistics/GCST007001-GCST008000/GCS
```

MungeSumstats

The MungeSumstats package is designed to facilitate the standardization of GWAS summary statistics.

```r
library(tidyverse)
```

```r
load.raw <- read_table('resources/Kunkle_etal_Stage1_results.txt')

load <- load.raw %>%
  filter(nchar(Effect_allele) == 1 & nchar(Non_Effect_allele) == 1) %>%
  mutate(
    Ncaas = 21982,
    Nctrl = 41944,
    N = 63926
  )

reformatted <-
  MungeSumstats::format_sumstats(path=load,
                                 ref_genome="GRCh37",
                                 dbSNP = 144,
                                 return_data = TRUE
                                 ) %>%
  as_tibble()

write_tsv(reformatted, 'work/Kunkle2019load.tsv.gz')
```

Manhattan Plot

```r
load.df_p <- reformatted %>%
  filter(., P < 0.1) %>%
  filter(., P > 1e-50) %>%
  select(SNP, CHR, BP, P)

don <- load.df_p %>%

  # Compute chromosome size
  group_by(CHR) %>%
  summarise(chr_len=max(BP)) %>%
  # Calculate cumulative position of each chromosome
  mutate(tot=cumsum(chr_len)-chr_len) %>%
  select(-chr_len) %>%

  # Add this info to the initial dataset
  left_join(load.df_p, ., by=c("CHR"="CHR")) %>%

  # Add a cumulative position of each SNP
  arrange(CHR, BP) %>%
  mutate( BPcum=BP+tot)

# Prepare X axis
axisdf <- don %>% group_by(CHR) %>% summarize(center=( max(BPcum) + min(BPcum) ) / 2 )

# Make the plot
load.p <- ggplot(don, aes(x=BPcum, y=-log10(P))) +

    # Show all points
    geom_point( aes(color=as.factor(CHR)), size=0.5) +
    scale_color_manual(values = rep(c("grey50", "steelblue"), 22 )) +

    # custom X axis:
    scale_x_continuous( label = axisdf$CHR, breaks= axisdf$center ) +
    scale_y_continuous(expand = c(0, 0) ) +     # remove space between plot area and x axi

    # Add highlighted points
    # geom_point(data=subset(don, is_highlight=="yes"), color="orange", size=2) +

    # Add label using ggrepel to avoid overlapping
    # geom_label_repel( data=subset(don, is_annotate=="yes"), aes(label=SNP), size=2) +

    # Custom the theme:
    theme_bw() +
    labs(title = "LOAD - Kunkle et al. Nature Genetics 2019") +
    theme(
      legend.position="none",
      panel.border = element_blank(),
      panel.grid.major.x = element_blank(),
      panel.grid.minor.x = element_blank(),
      axis.title.x = element_blank()
    )
```

```r
load.p <- ggman::ggman(load.df_p, snp = "SNP", bp = "BP", chrom = "CHR", pvalue = "P", rel
  labs(title = "LOAD - Kunkle et al. Nature Genetics 2019") +
  theme_classic()

ggsave("results/plots/kunkle2019load_manhattan_plot.png", height = 4, width = 6, units = '
```

Genetic Ancestry

Genetic ancestry plays a pivotal role in genome-wide association studies (GWAS), providing insights into the population-specific genetic variations that may contribute to disease phenotypes. Accurate assessment of ancestry allows for the control of population stratification, which can confound results if not properly accounted for. Principal Component Analysis (PCA) is commonly used to visualize and correct for ancestry-related differences by identifying axes of genetic variation. Additionally, understanding admixture improves our interpretation of genetic data, enabling more precise localization of disease-associated variants in diverse populations.

Prinicpal Component Analysis

PLINK PCA

PLINK enables us to conduct Principal Component Analysis (PCA) on genetic data. In this case, we have merged the HABS-HD dataset with the 1000 Genomes (1KG) dataset to initially derive principal components from the 1KG, which are then used to project the genetic data of the HABS-HD dataset onto. This will generate two files `work/imputed_1kG_merged.eigenvec` - the eigenvectors - and `work/imputed_1kG_merged.eigenval` the PCs.

To run the following PLINK command you will need the following files.

- `resources/genetic_epi/imputed_1kG_merged.bim`
- `resources/genetic_epi/imputed_1kG_merged.bed`
- `resources/genetic_epi/imputed_1kG_merged_fixed.fam`
- `resources/genetic_epi/1kG_pops.txt`
- `resources/genetic_epi/1kG_pops_unique.txt`

You can simplfy this by softlinking the Box `genetic_epi` directory into the resources directory of the IntroGeneticEpi git repository.

```
ln -s ~/Box-Box/AndrewsLab/data/genetic_epi ~/gitcode/IntroGeneticEpi/resources
```

```
plink \
    --keep-allele-order \
    --bfile resources/genetic_epi/imputed_1kG_merged \
    --fam resources/genetic_epi/imputed_1kG_merged_fixed.fam \
    --pca 10 \
    --within resources/genetic_epi/1kG_pops.txt \
    --pca-clusters resources/genetic_epi/1kG_pops_unique.txt \
    --out work/imputed_1kG_merged
```

Cluster Assignment

To categorize HABS-HD samples into the closest 1KG superpopulation, we calculate the geometric median for each cluster, assess the Euclidean distance from each sample to these medians, and assign samples to the nearest cluster accordingly.

```
#!/usr/bin/env Rscript

message("Loading packages")

suppressPackageStartupMessages(library(dplyr))
library(readr)
library(magrittr)
suppressPackageStartupMessages(library(tidyr))
library(stringr)
library(tibble)
suppressPackageStartupMessages(library(purrr))
library(ggplot2)

# Get geometric median
## rdocumentation.org/packages/bigutilsr/versions/0.3.3/topics/geometric_median

geometric_median <- function(u, tol = 1e-10, maxiter = 1000, by_grp = NULL) {
  if (!is.null(by_grp))
    return(do.call("rbind", by(u, by_grp, geometric_median)))
  u_old <- colMeans(u)
  for (k in seq_len(maxiter)) {
    norm <- sqrt(rowSums(sweep(u, 2, u_old, "-")^2))
    u_new <- colSums(sweep(u, 1, norm, "/")) / sum(1 / norm)
    diff <- max(abs(u_new - u_old))
    if (diff < tol)
      break
    u_old <- u_new
  }
  if (k == maxiter)
    warning("The maximum number of iterations has been reached.")
  u_new
}

# assign sample to cluster
## https://www.biorxiv.org/content/10.1101/2020.10.06.328203v2.full
## adomingues.github.io/2015/09/24/finding-closest-element-to-a-number-in-a-list

find_cluster <- function(df, clusters) {
  superpops <- clusters$superpop
  samp_pcs <- select(df, starts_with("PC"))
  mat <- bind_rows(clusters, samp_pcs) %>% {suppressWarnings(dist(.))}
  # mat
  clus <- which.min(as.matrix(mat)[6, 1:5])
  dplyr::mutate(df, superpop_infered = superpops[clus])
}
```

You will need to ensure the following files are in your `resources` and `work` directories.

```
vec <- 'work/imputed_1kG_merged.eigenvec'
val <- 'work/imputed_1kG_merged.eigenval'
base <- 'resources/genetic_epi/1kG_pops.txt'
target <- 'work/habshd_sampleQC.fam'
sample <- 'HABS-HD'
population <- 'all'
pcs_out_path <- 'work/habs_pca.tsv'
tg_pops_file <- 'resources/genetic_epi/tg_subpops.tsv'


#load("output/ADGC/x_present_AA/ADC8-AA_exclude.pca.params.Rdata")

if (tolower(population) == "all") {
  all_pops <- T
} else {
  all_pops <- F
}

##---- Read in Data ----##
message("Reading data files")
```

```r
# count columns and PCs
n_eig <- count_fields(vec, tokenizer_delim(delim = " "), n_max = 1) - 2

# Generate colnames
pc_names <- paste0("PC", 1:n_eig)
names_col <- c("FID", "IID", pc_names)

# Read in eigenvectors and z-score transform
pca_orig <- read_delim(vec,
                  delim = " ", col_names = names_col,
                  col_types = cols(.default = "d", FID = "c", IID = "c")) %>%
         mutate_at(pc_names, function(x) as.vector(scale(x)))

# read in egienvalues
eigenval <- val %>%
  read_lines %>%
  parse_number %>%
  tibble(eigenval = .,
         PC = factor(pc_names, levels = pc_names)) %>% #PC Names
  mutate(PVE = round(eigenval / sum(eigenval), 3)) %>% #PVE
  select(PC, eigenval, PVE) #Reorder columns

# population data file, usually from 1000 genomes and potentially with extra ref
base_pops_raw <- read_table(base, col_types = cols(.default = "c"))

# population data from target set
famcols <- c("FID", "IID", "PID", "MID", "Sex", "Pheno")
target_pops_raw <- read_table(target, col_names = famcols,
  col_types = "ccccii")

message("Processing data")
# ---- Data wrangling ---- #

# Read in populations and superpops
tg_pops <- read_tsv(tg_pops_file, col_types = "cccc")
populations <- tg_pops %>% select(pop, spop) %>% deframe %>% as.list
superpops <- unlist(populations) %>% unique()

# Deal with invalid cohort names

if (sample %in% names(populations)) {
  sample <- paste0("s_", sample)
}

if (sample %in% populations) {
  sample_s <- paste0("s_", sample)
} else {
  sample_s <- sample
}


##  Munge population dataframes from 1000 genomes
base_pops <- base_pops_raw %>%
  mutate(cohort = "Reference",
         superpop = recode(.$Population, !!!populations))

##  Munge target population dataframes
target_pops <- target_pops_raw %>%
  select(FID, IID) %>%
  mutate(Population = sample, superpop = sample_s,
    cohort = sample_s)

## Check this
remove_tg <- TRUE
if (remove_tg) {
  target_pops <- target_pops %>%
    filter(!(IID %in% base_pops$IID & FID %in% base_pops$FID))
}

# fix improperly split FID_IID
pca_fidiid <- pca_orig %>%
  unite("FIDIID", FID, IID, sep = "_")


##  Munge PCA, base pop and target pop
both_pops <- target_pops %>%
  bind_rows(base_pops) %>%
  ##### FIX BAD FID_IID SPLIT #####
  unite("FIDIID", FID, IID, sep = "_", remove = F)

pca_corrected <- pca_fidiid %>%
  left_join(both_pops, by = "FIDIID") %>%
  select(any_of(names(both_pops)), everything(), -FIDIID) %>%
  mutate(FID = str_remove(FID, "^1000g___"))

## Colours for plots
pca_col <- pca_corrected %>%
  count(superpop) %>%
  mutate(color = ifelse(superpop == sample_s, "black", NA)) %>%
  mutate(color = ifelse(superpop == "AFR", "#E69F00", color)) %>%
  mutate(color = ifelse(superpop == "AMR", "#0072B2", color)) %>%
  mutate(color = ifelse(superpop == "EAS", "#009E73", color)) %>%
  mutate(color = ifelse(superpop == "EUR", "#CC79A7", color)) %>%
  mutate(color = ifelse(superpop == "NFE", "#CC79A7", color)) %>%
  mutate(color = ifelse(superpop == "FIN", "#960018", color)) %>%
  mutate(color = ifelse(superpop == "SAS", "#D55E00", color)) %>%
  mutate(color = ifelse(superpop == "MID", "#56B4E9", color)) %>%
  mutate(color = ifelse(superpop == "AMI", "#F0E442", color)) %>%
  add_row(
    superpop = c("Black", "Hispanic", "NHW"),
    n = 0,
    color = c("#E69F00", "#0072B2", "#CC79A7")
  )


# Pull out 1000 genomes samples
kg <- filter(pca_corrected, cohort == "Reference")

# find geometric median of each PC for each cluster
clusters <-
  select(kg, starts_with("PC")) %>%
  geometric_median(by_grp = kg$superpop) %>%
  as_tibble(rownames = "superpop")

# extract sample information and assign to cluster
pca <- pca_corrected %>%
  group_split(IID) %>%
  map_df(find_cluster, clusters)

# Export PCA
write_tsv(pca, pcs_out_path)
```

Visualization

Now we can visualize the PCA to see how HABS-HD clusters with 1KG

```
color_vector <- setNames(pca_col$color, pca_col$superpop)

# PC1 x PC2
ga_pc1 <- ggplot() +
  geom_point(data = filter(pca, cohort == 'Reference'),
             aes(x = PC1, y = PC2, color = superpop), shape = 15, size = 2) +
  geom_point(data = filter(pca, cohort == 'HABS-HD'),
             aes(x = PC1, y = PC2, color = superpop), size = 0.75) +
  scale_color_manual(values = color_vector) +
  theme_bw()

# PC3 x PC4
ga_pc3 <- ggplot() +
  geom_point(data = filter(pca, cohort == 'Reference'),
             aes(x = PC3, y = PC4, color = superpop), shape = 15, size = 2) +
  geom_point(data = filter(pca, cohort == 'HABS-HD'),
             aes(x = PC3, y = PC4, color = superpop), size = 0.75) +
  scale_color_manual(values = color_vector) +
  theme_bw()

habshd_ga.p <- cowplot::plot_grid(
  ga_pc1, ga_pc3
)

ggsave("results/plots/habs_hd_ga.png", plot = habshd_ga.p, units = "in", width = 9, height
```

Lets join with the phenotype data to compare ancestry and race

```
pca_pheno <- pca %>%
  filter(cohort == "HABS-HD") %>%
  mutate(IID = as.numeric(IID)) %>%
  left_join(read_csv('work/habshd_pheno.csv'), by = c('IID' = 'med_id'))

# PC1 x PC2
habshd_ga <- ggplot(pca_pheno, aes(x = PC1, y = PC2, color = superpop_infered)) +
  geom_point() +
  scale_color_manual(values = color_vector) +
  theme_bw() +
  labs(title = "Genetic Ancestry")

# PC1 x PC2
habshd_race <- ggplot(pca_pheno, aes(x = PC1, y = PC2, color = race)) +
  geom_point() +
  scale_color_manual(values = color_vector) +
  theme_bw() +
  labs(title = "Race")

habs_hd_race_ga.p <- cowplot::plot_grid(
  habshd_ga, habshd_race
)


ggsave("results/plots/habs_hd_race_ga.png", plot = habs_hd_race_ga.p, units = "in", width
```

ADMIXTURE

Reference Processing

To prepare the gnomAD reference, we did the following:

- Remove samples without a population inference from gnomAD or without high_quality set to TRUE.
- Make a column (`spop`) by doing the following with the populations inferred by gnomAD:
    - Merge "nfe" and "fin" into "EUR"
    - Move oceanic subjects from "oth" to their own "OCE" category.
    - Capitalize all other superpopulations.
- Make a column (`spop_checked`) where the original superpopulations match the inferred superpopulations:
    - The new `spop` column is used for inferred superpopulation.
    - The `genetic_region` column is used for original superpopulation.
    - "CSA" in `genetic_region` is considered a match to "SAS" in `spop`. "SAS" is used in the new column.
    - All subjects where there is no match are set to "NA"

ADMIXTURE Procedure

The following steps are used to generate Global Ancestry Inference (GAI) estimates:

1. Process the reference label data as described above.
2. Obtain the intersection of the reference and target varients, then prune the reference with a 100kb window and R^2 of 0.1.
3. Restrict sample genotypes to those present in the pruned reference, then merge with the reference samples. Check that the `.bim` files are identical.
4. Run unsupervised ADMIXTURE with K = 12 on the reference dataset.
5. Run ADMIXTURE projection on the merged reference and target samples.
6. Read in the processed reference labels, ADMIXTURE cluster estimates (Q files), and PLINK `.fam` files.
7. Merge the reference labels with the ADMIXTURE cluster estimates and extract the reference samples for labeling, excluding Middle Eastern reference samples.
8. Label the clusters by assigning to each cluster the superpopulation with the highest average proportion within that cluster. The checked superpopulation labels are used for this labeling process.
9. Using the cluster labels, calculate GAI proportions and maximum superpopulation for all samples.
10. Visualize below.

We can execute ADMIXTURE using the code below; however, it requires approximately 24 hours of compute time. We have determined that K=12 is the optimal number of ancestral populations for 1KG + HGDP datasets. Our goal is to project the HABS-HD samples onto this reference dataset. This will produce `.Q` and `.P` file that contain the estimated ancestry fractions for each individual across the inferred populations and the allele frequencies for each population respectivly.

```
admixture -P -s 42 habshd_merged_gnomad-hgdp-1kg.hg38.bed 12 -j1
```

Lets visualize the global ancestry of the HABS-HD dataset. Make sure the following files are in your `work` directory.

- `work/gnomad-hgdp-1kg_pruned_habshd.hg38.fam`
- `work/habshd_merged_gnomad-hgdp-1kg.hg38.12.Q`
- `work/habshd_merged_gnomad-hgdp-1kg.hg38.fam`
- `work/hgdp_1kg.popdata.tsv.gz`

```
suppressPackageStartupMessages(library(dplyr))
library(readr)
library(tidyr)
library(purrr)
library(tibble)
library(stringr)
```

```r
## Input and output files
in_fam_ref <- 'work/gnomad-hgdp-1kg_pruned_habshd.hg38.fam'
in_q_samp <- 'work/habshd_merged_gnomad-hgdp-1kg.hg38.12.Q'
in_fam_samp <- 'work/habshd_merged_gnomad-hgdp-1kg.hg38.fam'
in_pops <- 'work/hgdp_1kg.popdata.tsv.gz'
out_anc <- 'work/admixture_cohort-habshd_ref-gnomad-hgdp-1kg.hg38.tsv'

# Fam and popfiles
## ======================================##
message("Reading pop file \n")
pops <- in_pops |>
  read_tsv(col_types = cols(.default = "c")) |>
  rename(ID = IID)

message("Reading fam files \n")
read_fam <- function(in_fam) {
  in_fam |>
    read_table(col_names = c("ID"), col_types = "-c----") |>
    mutate(order = row_number())
}

famfile_ref <- read_fam(in_fam_ref) |>
  mutate(partition = "reference")
famfile_samp <- read_fam(in_fam_samp) |>
  mutate(partition = "sample")

famfile <- bind_rows(famfile_ref, famfile_samp)

# Interpreting unsupervised admixture output #
## ======================================##
message("Reading unsupervised admixture output \n")

read_q <- function(in_q, fam) {
  in_q |>
    read_table(col_names = FALSE, col_types = cols(.default = "d")) |>
    bind_cols(fam) |>
    rename_with(~ str_replace(.x, "^X", "k"))
}

tbl_admix_samp <- read_q(in_q_samp, famfile_samp)
```

```r
overlap <- intersect(famfile_ref$ID, tbl_admix_samp$ID)
if (length(overlap) == nrow(famfile_ref)) {
  tbl_admix <- tbl_admix_samp |>
    left_join(pops, by = "ID") |>
    mutate(partition = ifelse(ID %in% famfile_ref$ID, "reference", partition))
  tbl_admix_ref <- tbl_admix |>
    filter(ID %in% famfile_ref$ID) |>
    mutate(FID = "reference")
  tbl_admix_samp <- tbl_admix |>
    filter(!(ID %in% tbl_admix_ref$ID))
} else if (length(overlap) != 0) {
  stop("Missing reference samples")
} else {
  tbl_admix_ref <- read_q(in_q_ref, famfile_ref)
  tbl_admix_ref <- tbl_admix_ref |>
    left_join(pops, by = "ID") |>
    mutate(FID = "reference")
  tbl_admix <- bind_rows(tbl_admix_ref, tbl_admix_samp)
}

# Determining cluster labels

cluster_cols <- names(tbl_admix)[str_detect(names(tbl_admix), "^k\\d+$")]

assign_labels <- function(tbl_admix) {
  if ("spop_checked" %in% colnames(tbl_admix)) {
    assign_admix_raw <- tbl_admix |>
      select(any_of(c("FID", "IID", "ID")),
        spop = spop_checked, matches("^k\\d+$")) |>
      filter(spop != "MID") |> # remove middle eastern from assignment
      group_by(spop) |>
      summarise(across(where(is.numeric), mean)) |>
      filter(!is.na(spop))
  } else {
    assign_admix_raw <- tbl_admix |>
      group_by(spop) |>
      summarise(across(where(is.numeric), mean)) |>
      filter(!is.na(spop))
  }

  assign_admix_mat <- assign_admix_raw |>
```

```
    column_to_rownames(var = "spop") |>
    as.matrix()

  assign_admix <- assign_admix_mat |>
    t() |>
    as.data.frame() |>
    (\(.) mutate(., anc = colnames(.)[apply(., 1, which.max)]))() |>
    as_tibble(rownames = "cluster") |>
    rowwise() |>
    mutate(maxval = max(c_across(where(is.numeric)))) |>
    group_by(anc) |>
    arrange(-maxval) |>
    mutate(n = n(),
           cname = ifelse(n > 1, paste(anc, row_number(), sep = "_"), anc)) |>
    ungroup() |>
    select(-maxval, -n) |>
    arrange(cname) |>
    select(cname, cluster, anc, everything())

  assign_cname_vec <- pull(assign_admix, cname, cluster)

  heatmap_names <- colnames(assign_admix_mat) |>
    (\(x) sprintf("%s (%s)", assign_cname_vec[x], x))()

  heatmap_mat <- assign_admix_mat
  colnames(heatmap_mat) <- heatmap_names

  return(list(assign = assign_admix,
              heatmap = heatmap_mat,
              assign_cname_vec = assign_cname_vec))
}

admix_labs <- assign_labels(tbl_admix)

assign_admix <- admix_labs$assign
heatmap_mat <- admix_labs$heatmap
assign_cname_vec <- admix_labs$assign_cname_vec
assign_super_vec <- pull(assign_admix, anc, cluster)
assign_cname_vec_i <- pull(assign_admix, cluster, cname)
superpops <- rownames(heatmap_mat)
```

```r
rm(admix_labs, assign_labels)

# Assign individuals

collapse_superpop <- function(df, sp) {
  # Add overall proportion of each superpop, collapsing clusters
  get_clusters <- \(spop) names(assign_super_vec[assign_super_vec == spop])
  mutate(df, !!sp := rowSums(across(all_of(get_clusters(sp)))))
}

tbl_admix_collapsed <- tbl_admix
for (sp in superpops) {
  tbl_admix_collapsed <- collapse_superpop(tbl_admix_collapsed, sp)
}

tbl_admix_inf <- tbl_admix_collapsed |>
  rowwise(ID) |>
  mutate(maxval = max(c_across(all_of(cluster_cols))),
         matchval = which.max(c_across(all_of(cluster_cols))),
         max_spop_prop = max(c_across(all_of(superpops)))) |>
  ungroup() |>
  (\(.) mutate(.,
    maxclust = colnames(.)[max.col(select(., matches("^k\\d+$")))],
    "Maximum Cluster" = unname(assign_cname_vec[maxclust]),
    admixture_super_pop_max = map_chr(maxclust, \(x) assign_super_vec[[x]]),
    admixture_cluster_max = map_chr(maxclust, \(x) assign_cname_vec[[x]])))() |>
  arrange(spop, admixture_super_pop_max, matchval, -maxval) |>
  mutate(
    pop = forcats::fct_inorder(pop),
    spop = forcats::fct_inorder(spop),
    admixture_super_pop_max = factor(
      admixture_super_pop_max, levels = levels(spop))) |>
  arrange(matchval, -maxval) |>
  mutate(ID = forcats::fct_inorder(ID))

out_admix <- tbl_admix_inf |>
  select(-maxval, -matchval, -maxclust) |>
  select(any_of(c("FID", "IID", "ID")),
    all_of(superpops), matches("^k\\d+$"),
    everything()) |>
  filter(!is.na(pop) | partition == "sample")
```

```
# Filter samples

out_admix |>
  arrange(desc(partition), admixture_super_pop_max) |>
  select(any_of(c("FID", "IID", "ID")), partition, admixture_cluster_max,
         admixture_super_pop_max, max_spop_prop, everything()) |>
  select(-`Maximum Cluster`) |>
  rename_with(~ paste0("k_", assign_cname_vec[.x]), matches("^k\\d+$")) |>
  write_tsv(out_anc)
```

Visualization

And now we can visualize the global ancestry.

```
tbl_use <-  out_admix %>%
  filter(partition == "sample") %>%
  pivot_longer(all_of(c('AFR', 'AMR', 'EAS', 'EUR', 'OCE', 'SAS')), names_to = "Cluster",
  mutate(spop = ifelse(genetic_region == "CSA", "SAS", genetic_region))

tbl_use <-  out_admix %>%
  filter(partition == "sample") %>%
  pivot_longer(all_of(c('k1', 'k2', 'k3', 'k4', 'k5', 'k6', 'k7', 'k8', 'k9', 'k9', 'k10',
  mutate(spop = ifelse(genetic_region == "CSA", "SAS", genetic_region))


admixture.p <- ggplot(tbl_use, aes(x = ID, y = prop, fill = Cluster)) +
    geom_bar(position = "fill", stat = "identity", width = 1) +
    # scale_fill_manual(values = color_vector) +
    theme_classic() +
    labs(x = "Individual", y = "Global Ancestry", color = "Cluster") +
    theme(
      axis.text.x = element_blank(),
      axis.ticks.x = element_blank(),
      axis.title.y = element_blank(),
      axis.title.x = element_blank(),
      panel.grid.major.x = element_blank(),
      strip.text.x = element_text(angle = 90)) +
  facet_grid(~ admixture_super_pop_max, switch = "x",
                     scales = "free", space = "free")
```

```r
ggsave('results/plots/habshd_admixture.png', plot = admixture.p, units = 'in', width = 9,
```

Heritability & Genetic Correlations

```r
library(tidyverse)    # Data wrangling
library(GenomicSEM)
```

Methods

Tools & Publications

R, GenomicSEM, LDSC, HDL

- Bulik-Sullivan, B. et al. An atlas of genetic correlations across human diseases and traits. Nat Genet 47, 1236–1241 (2015).
- Ning, Z., Pawitan, Y. & Shen, X. High-definition likelihood inference of genetic correlations across human complex traits. Nat Genet 52, 859–864 (2020).

Genetic correlation (rg) refers to the degree to which the genetic determinants of two traits overlap - the proportion of variance that two traits share due to genetic causes. A positive genetic correlation between two traits implies that the same genetic variants are influencing both traits in the same direction. Conversely, a negative genetic correlation implies that the genetic variants influencing one trait are having the opposite effect on the other trait.

LDSC: Linkage disequilibrium score regression (LDSC) leverages linkage disequilibrium (LD), the non-random association of alleles at different loci, to estimate genetic correlations between two traits. This method operates on the premise that single nucleotide polymorphisms (SNPs) with a higher count of LD partners (thus having a higher LD score) are typically more associated with a trait due to polygenicity, a condition where numerous genetic variants each exert a minor effect.

HDL: High-definition likelihood (HDL) provides genetic correlation estimates that have higher accuracy and precision compared to LDSC. HDL achives this by using a full likelihood-based method that leverages LD information across the whole genome, where as LDSC only use partial information.

Munge

Here we will be using LDSC and HDL implemented using GenomicSEM - make sure to have it installed.

You will need to make sure the following summary statistics are in `resources/genetic_epi/summary_statistic`

- `Willer2013ldl.chrall.CPRA_b37.tsv.gz`
- `Graham2021ldl.chrall.CPRA_b37.tsv.gz`
- `Kunkle2019load_stage123.chrall.CPRA_b37.tsv.gz`
- `Bellenguez2022load.chrall.CPRA_b37.tsv.gz`

> ⚠ Warning
>
> With large GWAS summary statistic files your local machine may run out of memory. There are also HapMap3 filtered summary statistic files avaliable
>
> - work/summary_statistics/Willer2013ldl_hm3.tsv.gz
> - work/summary_statistics/Graham2021ldl_hm3.tsv.gz
> - work/summary_statistics/Kunkle2019load_hm3.tsv.gz
> - work/summary_statistics/Bellenguez2022load_hm3.tsv.gz
>
> You may also need to apply `GenomicSEM::munge` to a single summary statistic file at a time

And that the LD Reference Panels are available in `resources/genetic_epi/ld_ref/`

First we need to munge the GWAS summary statistics so they are in the format required for LDSC.

```
## Summary statistics - full summary stats, may cause memory failure
# Willer2013ldl = "resources/genetic_epi/summary_statistics/Willer2013ldl.chrall.CPRA_b37.
# Graham2021ldl = "resources/genetic_epi/summary_statistics/Graham2021ldl.chrall.CPRA_b37.
# KunkleAD = "resources/genetic_epi/summary_statistics/Kunkle2019load_stage123.chrall.CPRA
# BellenguezAD = "resources/genetic_epi/summary_statistics/Bellenguez2022load.chrall.CPRA_

## Summary statistics - HapMap3 filtered SNPs
Willer2013ldl = "work/summary_statistics/Willer2013ldl_hm3.tsv.gz"
Graham2021ldl = "work/summary_statistics/Graham2021ldl_hm3.tsv.gz"
KunkleAD = "work/summary_statistics/Kunkle2019load_hm3.tsv.gz"
BellenguezAD = "work/summary_statistics/Bellenguez2022load_hm3.tsv.gz"

## LD Structure
```

```
ld_path = "resources/genetic_epi/ld_ref/eur_w_ld_chr/"

## HAPMAP3 SNPs
hm3_path = "resources/genetic_epi/ld_ref/w_hm3.snplist"


GenomicSEM::munge(
  files = c(Willer2013ldl, Graham2021ldl, KunkleAD, BellenguezAD),
  hm3 = hm3_path,
  trait.names = c("Willer2013ldl", "Graham2021ldl", "KunkleAD", "BellenguezAD"),
  maf.filter = 0.05,
  column.names = list(
    SNP='DBSNP_ID',
    MAF='AF',
    A1='ALT',
    A2='REF',
    effect='BETA',
    N = "N"
  ),
  overwrite=FALSE
)
```

LDSC

We can then apply LDSC to estimate h2 and pairwise rg. As we are using binary outcomes, we need to specify sample and population prevalence.

Trait Name	Sample Prevalence	Population Prevalence
Willer2013ldl	NA	NA
Graham2021ldl	NA	NA
BellenguezAD	0.18	0.31
KunkleAD	0.37	0.31

```
ldsc.covstruct <- GenomicSEM::ldsc(
     traits = c("Willer2013ldl.sumstats.gz", "Graham2021ldl.sumstats.gz", "BellenguezAD.su
     trait.names = c("Willer2013ldl", "Graham2021ldl", "BellenguezAD", "KunkleAD"),
     sample.prev = c(NA, NA, 0.18, 0.37),
     population.prev = c(NA, NA, 0.31, 0.31),
     ld = ld_path,
     wld = ld_path,
     stand = TRUE
     )
```

HDL

We can then apply HDL to estimate h2 and pairwise rg.

```
hdl.covstruct <- GenomicSEM::hdl(
     traits = c("Willer2013ldl.sumstats.gz", "Graham2021ldl.sumstats.gz", "BellenguezAD.su
     trait.names = c("Willer2013ldl", "Graham2021ldl", "BellenguezAD", "KunkleAD"),
     sample.prev = c(NA, NA, 0.18, 0.37),
     population.prev = c(NA, NA, 0.31, 0.31),
     LD.path="resources/UKB_imputed_hapmap2_SVD_eigen99_extraction/",
     method = "piecewise"
     )
```

Part III

Polygenic Risk Scores

Polygenic Risk Scores

This is a book created from markdown and executable code.

See[13] for additional discussion of literate programming.

Part IV

Mendelian Randomization

Mendelian Randomization

TBD

References

1. Purcell S, Neale B, Todd-Brown K, et al. PLINK: A Tool Set for Whole-Genome Association and Population-Based Linkage Analyses. *The American Journal of Human Genetics*. 2007;81(3):559-575.
2. Danecek P, Bonfield JK, Liddle J, et al. Twelve years of SAMtools and BCFtools. *GigaScience*. 2021;10(2):giab008.
3. Murphy AE, Schilder BM, Skene NG. MungeSumstats: A Bioconductor package for the standardisation and quality control of many GWAS summary statistics. *Bioinformatics*. 2021;37(23):btab665-.
4. Alexander DH, Novembre J, Lange K. Fast model-based estimation of ancestry in unrelated individuals. *Genome Research*. 2009;19(9):1655-1664.
5. Maples BK, Gravel S, Kenny EE, Bustamante CD. RFMix: A Discriminative Modeling Approach for Rapid and Robust Local-Ancestry Inference. *The American Journal of Human Genetics*. 2013;93(2):278-288.
6. Bulik-Sullivan B, Finucane HK, Anttila V, et al. An atlas of genetic correlations across human diseases and traits. *Nature Genetics*. 2015;47(11):1236-1241.
7. Ning Z, Pawitan Y, Shen X. High-definition likelihood inference of genetic correlations across human complex traits. *Nature Genetics*. 2020;52(8):859-864.
8. Grotzinger AD, Rhemtulla M, Vlaming R de, et al. Genomic structural equation modelling provides insights into the multivariate genetic architecture of complex traits. *Nature human behaviour*. 2019;3(5):513-525.
9. Choi SW, O'Reilly PF. PRSice-2: Polygenic Risk Score software for biobank-scale data. *GigaScience*. 2019;8(7).
10. Choi SW, García-González J, Ruan Y, et al. PRSet: Pathway-based polygenic risk score analyses and software. *PLOS Genetics*. 2023;19(2):e1010624.
11. Ruan Y, Lin YF, Feng YCA, et al. Improving polygenic prediction in ancestrally diverse populations. *Nature Genetics*. 2022;54(5):573-580.
12. Hemani G, Zheng J, Elsworth B, et al. The MR-Base platform supports systematic causal inference across the human phenome. *eLife*. 2018;7:e34408.
13. Knuth DE. Literate programming. *Comput J*. 1984;27(2):97-111.

Acknowledgments

The code used to conduct the genotyping QC and ancestry assignments was developed by Brian Fulton-Howard, PhD for the QAIC workflow.