Data Dictionary And Documentation

February 3, 2013
Version: 5.0

Long Life Family Study
Data Coordinating Center
Division of Statistical Genomics
Washington University School of Medicine
4444 Forest Park Blvd., Campus Box 8506
St. Louis, MO, 63108

Table of Contents

Introduction

This document describes the analysis datasets and derived variables for the Long Life Family Study. In general, each form/procedure in either Pre-Clinic (PTSI, Relatives), Clinic or Follow-Up is stored as its own SAS dataset (e.g. blood, meds, physical, venip, etc.), with one record (observation) per person. The few exceptions to this rule are noted in the appropriate places. For the most part, each dataset retains the original, raw form variables as collected on each subject. These are mnemonically named (e.g. SEX instead of Q7), with accompanying SAS labels. A user-defined format library is also included to provide value-labels to codes. Thus, PROC CONTENTS, along with PROC FORMAT with the FMTLIB option can be used in conjunction with the official book of forms and QxQs (also supplied by the Coordinating Center) to provide documentation for the raw form data itself. We concentrate here instead on documenting the dataset organization, and derived analysis variables created at the Coordinating Center.

Derived Variables

For the most part, all derived variables are named beginning with an underscore, to readily distinguish them from the raw form variables. The derived variables are stored in the "natural" form/procedure dataset from which they were created, e.g. _BMI is in BPHR dataset, total cholesterol is in the BLOOD dataset, etc. In the various sections below, we summarize each dataset, followed by a description of each derived variable in it. In many cases, we give not only the algorithm used in defining the variable, but also the actual SAS code which implements it, for completeness, and to assure that the final coding is as intended. Each of the derived variables is described first, then the SAS code that created all of them is presented.

Common Variables

Most datasets have a few, common identifying variables.

ID

ID is the original key identifying variable, at the time of data collection. ID is usually in each dataset. It is the 8 character LLFS ID. The structure of ID is:

CNNNNNNN
C is the single digit field center number:
1 = Denmark, DK 2 = Boston, MA
3 = Pittsburgh, PA 4 = New York, NY

TOUCHDAT

TOUCHDAT is a SAS datetime variable automatically updated by the Data Entry System at Field Centers to give the date/time when the record was last changed.

DATE

DATE is a SAS date variable giving the date the form/procedure was collected.

FC

FC is the field center.
1 = Denmark, DK 2 = Boston, MA
3 = Pittsburgh, PA 4 = New York, NY

SUBJECT

SUBJECT is a de-identified, unique identifier for each Participant. It is a 5 digit number. Datasets with one obs/subject are uniquely identified by ID and can be merged/linked using this variable.

PHASE I

Pre-In Person Visit

Family Longevity Collection Score Instrument – Proband (FLoSS) (PTSI) Dataset

This dataset contains the answers to eligibility questions, including those that were used to calculate the Family Longevity Selection Score (FLoSS), and some socio-demographic data. This dataset has no derived variables.

Proband Relative Contact Information (Relatives) Dataset

This questionnaire collected information on the Proband's relatives, such as contact information, and permission to contact them.
This dataset has no derived variables.

Sibling Information (SIBINFO) Dataset

This dataset contains information on the Proband's siblings.
This dataset has no derived variables.

In Person Visit

Blood (Blood) Dataset

This dataset contains the data regarding the blood chemistries, the date the blood was drawn, and the LLFS Subject. Special assays and candidate genes were added as well. This dataset has three derived variables:

_fteststrn - Free Testosterone (ng/dL)

Free T (mol/L)$= \frac{-b+\sqrt{b^2+4a[TT]}}{2a}$,
where a = ka + ks + (ka × ks) ([SHBG (mol/L)] + [Alb (mol/L)] − [TT (mol/L)]),
b = 1 + ks [SHBG] + ka [Alb] − (ka + ks) [TT],

ka = 3.6×10^4 L/mol,
ks = 1×10^9 L/mol.
Total testosterone (TT) (mol/L) = (Reported TT (ng/dl)/288.4) x 10^{-8}
Molecular weight of Testosterone = 288.4
SHBG (mol/L) = Reported SHBG (nmol/L) x 10^{-9}
Alb (mol/L) = (Reported Alb (g/dl) x 10)/69000
Molecular weight of Albumin = 69000

_albBT - Albumin Bound Testosterone (ng/dL)

Albumin Bound T (ng/dl) = [ka x ($C_{onc.}$Alb(mol/L)]*Free T
where ka = 3.6 x 10^4 L/mol

_biotestrn - Bioavailable Testosterone (ng/dL)

Bioavailable Testosterone = Free Testosterone + Albumin bound Testosterone

Blood Pressure, Heart Rate, Height, Weight, Waist (BPHR) Dataset

This dataset contains the data regarding the blood pressure, height, knee height and waist circumference.

_HEIGHT

_HEIGHT is the Average Standing Height in cm calculated from stand1, stand2, stand3, and stand4.

```
if stand1>0 and stand2>0 and (not(stand3>0) or not(stand4>0))
      then _height=mean(stand1,stand2);

    else if stand3>0 and stand4>0
      then _height=mean(stand3,stand4);
```

_BMI

_BMI is Body Mass Index calculated from weight and height on BPHR.

```
if (weight >0 and _height>0) then
     _BMI=(weight/((_height/100)**2));
```

_WAIST

_WAIST is the Average Abdominal Circumference in cm calculated from waist1, waist2, waist3, and waist4.

```
if waist1>0 and waist2>0 and (not(waist3>0) or not(waist4>0))
      then _waist=mean(waist1,waist2);
    else if waist3>0 and waist4>0
      then _waist=mean(waist3,waist4);
```

_SIT

_SIT is the Average Height while sitting in cm calculated from sit1, sit2, sit3, and sit4.

```
if sit1>0 and sit2>0 and (not(sit3>0) or not(sit4>0))
      then _sit=mean(sit1,sit2);
else if sit3>0 and sit4>0
      then _sit=mean(sit3,sit4);
```

_HTIN25 & _HTCM25

_HTIN25 is the Height when 25 years old in total inches.
_HTCM25 is the Height when 25 years old in cm.

```
if htft25>0 and htin25>=0 then  _htin25= (htft25 * 12) + htin25;
if htft25>0 and htin25<0 then _htin25= (htft25 * 12);
if htcm25>0 then _htin25= htcm25 / 2.54;
_htcm25 = _htin25 * 2.54;
```

_KNEE

_KNEE is the Average Length of Leg from Heel to Knee in cm.

```
_knee=mean(knee1,knee2);
```

_SBP

_SBP is the Average Sitting Systolic Blood Pressure in mmHg.

```
_sbp=mean(sys1,sys2,sys3);
```

_DBP

_DBP is the Average Sitting Diastolic Blood Pressure in mmHg.

```
_dbp=mean(dia1,dia2,dia3);
```

_PULSE

_PULSE is the Average Sitting Pulse.

```
_pulse=mean(pulse1,pulse2,pulse3);
```

Mood and Personality Assessment (CES-D and NEO 5-Factor) Dataset

This dataset contains the data regarding depressive symptomatology, personality dimensions of neuroticism and conscientiousness. Derived variables _NRAW and _CRAW are sums of different questions on the form.

_NRAW and _CRAW

```
array dv{60} worrier ppl_around daydream courteous neat inferior laugh stick2 argue pacing
     stress lightheart patterns selfish methodical blue talking controversy cooperate conscientious
     tense action poetry cynical cleargoals worthless alone newfoods takeadv wastetime
     anxious energy moods liked acmplshgoals angry cheerful moral cold commitment
     discouraged optimist excitement hardhead reliable sad fast universe considerate productive
     helpless active curious dislike organized ashamed leader abstract manipulate excellence;
array score{1:60} score1-score60;
do i=1 to 60;
  select(i);
    when(2,4,5,6,7,10,11,13,17,19,20,21,22,25,26,28,32,34,35,36,37,40,41,43,47,49,50,51,
        52,53,56,58,60)
        score{i} = dv{i} - 1;
    otherwise
        score{i} = 5 - dv{i};
  end;
```

```
end;
_nraw =
sum(score1,score6,score11,score16,score21,score26,score31,score36,score41,score46,score51,sc
ore56);
_craw =
sum(score5,score10,score15,score20,score25,score30,score35,score40,score45,score50,score55,s
core60);
drop i score1-score60 ppl_around--manipulate;
```

Coded Medications (CODEDMEDS) Dataset

This dataset contains the four yes/no variables Anne Newman and her staff at the University of Pittsburgh developed from the Medications data. Specifically, they are HTNRX, LIPIDRX, NITRORX, DMRX. These represent whether or not the Participant is currently taking a medication for Hypertension, Lipid Lowering, Angina, or Diabetes Mellitus.
This dataset has no derived variables.

Cognitive Assessments (COGASSESS) Dataset

There are three forms that compose this battery, the NACC UDS, the Telephone Interview for Cognitive Status (TICS, see below), and the Informant-Based Date of Onset Interview. This dataset has no derived variables.

Consent Tracking and Interview Feasibility (CTIF) Dataset

This dataset contains the answers to the consent questions. There were different questions at the different Field Centers, based upon what their individual Internal Review Boards required. This dataset has no derived variables.

Lung Function (LUNGFUNC) Dataset

This dataset contains the results of the Spirometry test, sent to the Data Coordinating Center from the Reading Center. This dataset has no derived variables.

Medication Inventory (MEDCHK) Dataset

This dataset contains the response of the first question of the Medication Inventory; if any medication was taken in the past 2 weeks. This dataset has no derived variables.

Medication Inventory (MEDS) Dataset

This dataset contains the responses of the rest of the questions from the Medication Inventory; the medication name, strength, units, formulation code, whether or not the container was seen, and other notes. This data set has 1 record per medication; therefore, multiple records per Participant. It includes the person's ID to link this dataset to the others. This dataset has no derived variables.

Medical History (MEDHX) Dataset

This dataset contains information about the Medical History of the Participants, including all diseases the person has/had. This dataset has no derived variables.

Personal History (PERSHX) Dataset

These variables cover the smoking and alcohol intake histories of the Participants.

_SMOKENOW, _PIPENOW, and _PACKYRS

_SMOKENOW is current smoker. _PIPENOW is current pipe user. _PACKYRS is the number of packs smoked per day over the number of years smoked. (# Packs/day * yrs smoke(d)).

```
length _Smokenow _Pipenow 3;
```

```
merge age(in=a) smoke(in=b);
by id; if b=1;
if smoke100=0 or smokequitage>0 or smokequityr>0 then _Smokenow=0;
  else _Smokenow=smokenow;
label _Smokenow="Current Smoker?";
if pipe=0 or pipequitage>0 or pipequityr>0 then _Pipenow=0;
  else _Pipenow=pipenow;
label _Smokenow="Current Smoker?" _Pipenow="Current Pipe User?";
format _Smokenow _Pipenow YND12.;
if smokenow not in (0,1) then _PACKYRS=.;
if cigday=. then _PACKYRS=.;
if _age =. then _PACKYRS=.;
if smokenow=1
  then _PACKYRS=(_age - smoke1stage)*(cigday/20);
  else if (smokenow=0 and smoke1styr ne . and smokequityr ne . and cigday ne .)
       then _PACKYRS=(smokequityr-smoke1styr)*(cigday/20);
  else if (smokenow=0 and smoke1stage ne . and smokequitage ne . and cigday ne .)
       then _PACKYRS=(smokequitage-smoke1stage)*(cigday/20);
_PACKYRS=round(_PACKYRS,.01);
label _PACKYRS="# Packs/day * yrs smoke(d)";
```

_SUMPACKYRS

_SUMPACKYRS is similar to _PACKYRS, above; however, non-smokers are now coded as 0 instead of missing, and the number of years smoked is calculated from the start and quit dates or ages smoked.

```
_datayr=year(date);
  if ((_smokenow=0) and (smoke100 ne 1)) then _sumpackyrs=0; /* never smoker */
  if ((_smokenow=1) and (smoke1stage^=.)) then _sumpackyrs=(_age -
smoke1stage)*(cigday/20); /* current smoker, using age */
  else if ((_smokenow=1) and (smoke1styr^=.)) then _sumpackyrs=(_datayr -
smoke1styr)*(cigday/20); /* current smoker, using year */

  if ((_smokenow=0) and (smokenow=0) and (smoke100=1) and (smoke1stage^=.) and
(smokequitage^=.) and (cigday^=.))
  then _sumpackyrs=(smokequitage - smoke1stage)*(cigday/20); /* former smoker, using age
start and quit */
  else if ((_smokenow=0) and (smokenow=0) and (smoke100=1) and (smoke1styr^=.) and
(smokequityr^=.) and (cigday^=.))
  then _sumpackyrs=(smokequityr - smoke1styr)*(cigday/20); /* former smoker, using year start
and quit */

  _yrstart=dobyr+smoke1stage; /* create year start smoking variable to use for missing former
smokers */
```

```
  _yrend=dobyr+smokequitage; /* create year end smoking variable to use for missing former
smokers */

  if ((_smokenow=0) and (smokenow=0) and (smoke100=1) and (_yrstart^=.) and
(smokequityr^=.) and (cigday^=.))
  then _sumpackyrs=(smokequityr - _yrstart)*(cigday/20); /* former smoker, using year start and
quit with extrapolated year start*/
  else if ((_smokenow=0) and (smokenow=0) and (smoke100=1) and (smoke1styr^=.) and
(_yrend^=.) and (cigday^=.))
  then _sumpackyrs=(_yrend - smoke1styr)*(cigday/20); /* former smoker, using year start and
quit with extrapolated year end*/
LABEL _sumpackyrs="Derived # packs/day * years smoke(d), for current smokers, former
smokers, and non-smokers";
  if _sumpackyrs=-0.2 then _sumpackyrs=.;
  if _sumpackyrs=-0.6 then _sumpackyrs=.;
```

_SMOKE_CIG, _SMOKE_PIPE, and _SMOKE_CAT

_SMOKE_CIG is current, former, or never cigarette smoker. _SMOKE_PIPE is current, former, or never pipe smoker. _SMOKE_CAT is current, former, or never cigarette or pipe smoker.

```
Proc format;
Value smkcat
1 = “Never smoked”
2 = “Former smoker”
3 = “Current smoker”;
Run;
```

```
**Cigarette smoking only**;

if smoke100=0 then _smoke_cig=1;
   else if smoke100=1 and _smokenow=0 then _smoke_cig=2;
   else if smoke100=1 and _smokenow=1 then _smoke_cig=3;

**Pipe smoking only**;

if pipe=0 then _smoke_pipe=1;
   else if pipe=1 and _pipenow=0 then _smoke_pipe=2;
   else if pipe=1 and _pipenow=1 then _smoke_pipe=3;

**Cigarette + Pipe smoking**;

if _smoke_cig=1 and _smoke_pipe=1 then _smoke_cat=1;
   else if _smoke_cig=1 and _smoke_pipe=2 then _smoke_cat=2;
   else if _smoke_cig=1 and _smoke_pipe=3 then _smoke_cat=3;
   else if _smoke_cig=2 and _smoke_pipe=1 then _smoke_cat=2;
   else if _smoke_cig=2 and _smoke_pipe=2 then _smoke_cat=2;
   else if _smoke_cig=2 and _smoke_pipe=3 then _smoke_cat=3;
```

```
   else if _smoke_cig=3 and _smoke_pipe=1 then _smoke_cat=3;
   else if _smoke_cig=3 and _smoke_pipe=2 then _smoke_cat=3;
   else if _smoke_cig=3 and _smoke_pipe=3 then _smoke_cat=3;
```

Physical Function and Activity (PHYSICAL) Dataset

This dataset covers the Physical Exercise form. Variables in the dataset include, activity level, duration, and frequency of exercise. This dataset has no derived variables.

Performance Measures (PM) Dataset

This dataset includes the results of the Short Physical Performance Battery (SPPB), and the Grip Strength Test.

_TOTSCORE

This derived variable provides one score for the entire SPPB.

WALKSCORE: gives ratings for various values of the walking test.

```
if length eq 1 then do;
   if 0 < shorter < 4.82 then walkscore = 4;
   else if 4.82 <= shorter <= 6.20 then walkscore = 3;
   else if 6.21 <= shorter <= 8.70 then walkscore = 2;
   else if 8.70 < shorter < 60.00 then walkscore = 1;
 end;
 if length eq 2 then do;
   if 0 < shorter < 3.62 then _walkscore = 4;
   else if 3.62 <= shorter <= 4.65 then walkscore = 3;
   else if 4.66 <= shorter <= 6.53 then walkscore = 2;
   else if 6.53 < shorter < 45.00 then walkscore = 1;
 end;
```

```
_totscore = sum (sidescore,semiscore,tdmscore,walkscore,chairscore);
```

Prevalence of Disease (PREVDISEASE) Dataset

This dataset was created to derive 7 variables that combine information from the Medical History, Blood Pressure, Blood, and Coded Medications data sets.

_HTDIS, _STRK, _LUNGDIS, _HTN, _DIABETES, _PAD, and _CANCER

```
proc sort data=clinic.medhx out=medhx;
by id;run;
proc sort data=clinic.bphr out=bphr;
```

```
by id;run;
proc sort data=blood.blood out=blood;
by id;run;
proc sort data=codemeds.codedmeds (keep=id htnrx lipidrx dmrx) out=cmeds;
by id;run;
proc sort data=clinic.sdi (keep=id sex) out=sdi;
by id;run;

data medhxvars(keep= id  _htdis  _strk _lungdis);
*data medhxvars(keep= id midx cabg _htdis stroke tia _strk asth bronch copd _lungdis);
  set medhx;
*create prevalent disease heart disease*;
if (midx=1 or cabg=1) then _htdis=1;
   else _htdis=0;
*create prevalent disease stroke*;

if (stroke=1 or tia=1) then _strk=1;
   else _strk=0;
*create prevalent disease lung diseaase*******;
if (asth=1 or bronch=1 or copd=1) then _lungdis=1;
   else _lungdis=0;
run;
*title '_htdis _strk _lungdis derivation';
*proc print;
*run;
proc sort; by id; run;

data htn(keep=id htdx);
  set medhx;
  run;
proc sort; by id; run;
data htn1 (keep=id htdx htnrx);
  merge htn(in=a) cmeds(in=b);
  by id;
  if a;
run;
proc sort; by id; run;
data bp(keep=id _htn);
*data bp(keep=id htdx htnrx sys1-sys3 avgsys dia1-dia3 avgdia _htn);
  merge htn1(in=a) bphr(in=b);
  by id;
  if a;
  avgsys=mean(sys1,sys2,sys3);
  avgdia=mean(dia1,dia2,dia3);
  if  ((htdx=1 and htnrx=1)
      or (avgsys >=140)
```

```
      or (avgdia >=90))
      then _htn=1;
      else _htn=0;
run;
*title '_htn derivation';
*proc print;
*run;
proc sort; by id; run;

data diab(keep=id diab);
  set medhx;
  run;
proc sort; by id; run;
data diab1(keep=id _diabetes);
*data diab1(keep=id diab dmrx glur _diabetes);
```

```
merge diab(in=a) cmeds(in=b) blood(in=c);
  by id;
  if a;
  if (diab=1 and dmrx=1 or glur >=126)
     then _diabetes=1;
     else _diabetes=0;
run;
*title '_diabetes derivation';
*proc print;
*run;
proc sort; by id; run;

data pad(keep=id _pad);
*data pad(keep=id aai aabprl aabprr _pad);
  set bphr;
  aai = min(aabprl,aabprr);
  if (aai <= 0.9 and aai > 0) then _pad = 1;
  else if aai > 0.9 then _pad=0;
  run;
*title '_pad derivation';
*proc print;
*run;
proc sort; by id; run;

data cancer(keep=id _cancer);
*data cancer(keep=id sex breast leuk colon lung melan skin esophgl pancr ocancer prost
_cancer);
  merge medhx(in=a) sdi(in=b);
  by id;
  if a;
```

```
  if (breast=1 or leuk=1 or colon=1 or lung=1 or melan=1 or skin=1 or esophgl=1
    or ocancer=1 or prost=1) and sex=1 then _cancer=1;
  if (breast=1 or leuk=1 or colon=1 or lung=1 or melan=1 or skin=1 or esophgl=1
    or ocancer=1) and sex=2 then _cancer=1;
  if _cancer ne 1 then _cancer=0;
  run;
*title '_cancer derivation';
*proc print;
*run;
proc sort; by id; run;

data codemeds.prevdisease;
  merge medhxvars(in=a) bp(in=b) diab1(in=c) pad(in=d) cancer(in=e);
  by id;
  if a;
```

```
if id='' then delete;
 if _htdis=.    then _htdis=0;
 if _strk=.     then _strk=0;
 if _htn=.      then _htn=0;
 if _diabetes=. then _diabetes=0;
 if _cancer=.   then _cancer=0;
 if _lungdis=.  then _lungdis=0;
 if _pad=.      then _pad=0;
 run;
```

Socio-Demographic Information (SDI) Dataset

This dataset contains the information collected on the Socio-Demographic form.

AGE

_AGE is the age when the form was filled out.

```
 _AGE=floor((date-dob)/365.25);
```

Spirometry Safety Questionnaire (SPIRO) Dataset

The Spirometry Safety Questionnaire asks background questions that would preclude taking the pulmonary function test, such as major surgery, heart attack, or stroke in the past three months. There are no derived variables in this dataset.

Spirometry Safety Questionnaire (SPIROMEDS) Dataset

This provides the medication data from question 9b of the Spirometry Safety Questionnaire. This data set has 1 record per medication; therefore, multiple records per Participant. It includes the person's ID to link this dataset to the others. This dataset has no derived variables.

Survival Indices (survl_indices) Dataset

This dataset contains derived variables that are indicators of survival and healthy aging.

Healthy Aging Index

Variables related to the healthy aging index (A. Newman, R. Minster, J. Sanders, et al, Pittsburgh) are ***HAI, HAI_m, HAI_rg, HAI_m_rg, HAI_rl,*** and ***HAI_m_rl***.

HAI

HAI is the evenly weighted healthy aging index. It is calculated from 5 other variables and thus is only calculated when a LLFS participant has a measurement for all 5 component variables. The 5 component variables are: systolic blood pressure, forced vital capacity, mini-mental state exam, serum creatinine, and serum fasting glucose.
To create the ***HAI***, each component variable receives a score of 0 (healthiest tertile), 1 (middle tertile), or 2 (unhealthiest tertile)—with the exception of fasting glucose, for which clinical cutoffs were applied. For systolic blood pressure, if a participant has a physician diagnosis of hypertension or if a participant was using medication for hypertension, they were coded in the unhealthiest tertile (score=2). Similarly for fasting glucose, if a participant has a physician diagnosis of diabetes or if a participant was using medication for diabetes, they were coded in the unhealthiest tertile (score=2). For forced expiratory volume and serum creatinine, separate tertiles were applied to men and women. The scores of the five component variables were then summed for each participant to create the HAI which has a range of 0 (healthiest) to 10 (unhealthiest).

HAI Tertile Thresholds

		0	1	2
Systolic blood pressure, mmHg*		< 126	≥ 126 and <143	≥ 143
Forced vital capacity, L	(*women*)	≥ 2.61	< 2.61 and ≥ 2.14	< 2.14
	(*men*)	≥ 3.84	< 3.84 and ≥ 3.19	< 3.19
MMSE, points		> 26	> 23 and ≤ 26	≤ 23
Serum creatinine, mg/dL	(*women*)	< 0.8	0.8–1.0	> 1.0
	(*men*)	< 1.1	1.1–1.3	> 1.3
Serum fasting glucose, mg/dL†		< 100	100–125	≥ 126

*Physician diagnosis of hypertension or taking anti-hypertensive medication led to score=2

† Physician diagnosis of diabetes or taking medication for diabetes led to score=2

HAI_rg

HAI_rg is the age, sex, and PC1-10 adjusted residuals of the HAI used for the GWAS.

HAI_m

HAI_m is the mortality weighted healthy aging index, where each component variable of the HAI is given a mortality optimized weight. Each component score was multiplied by the weight for that component, divided by the sum of all 5 component weights, and multiplied by 5 to obtain the weight for that component. Each component weight was then multiplied to the score for that component and all 5 weighted component scores were summed to calculate the HAI_m. When we calculated the cox models in CHS for each one point increase in the index we had the betas for mortality. We used the betas as modifiers for the component heritability weights (which before were all equal at 0.20) in the index. That way the weights were optimized for mortality prediction, with the most strongly associated components receiving a greater weight.

HAI_m Weights

	Weight
Systolic blood pressure, mmHg*	0.17085
Forced vital capacity, L	0.38386
MMSE, points	0.42873
Serum creatinine, mg/dL	0.13397
Serum fasting glucose, mg/dL†	0.23880

HAI_m_rg

HAI_m_rg is the age, sex, and PC1-10 adjusted residuals of the mortality weighted HAI, HAI_m, used for the GWAS.

HAI_rl

HAI_rl is the age and sex adjusted residuals of the HAI used for the linkage analysis.

HAI_m_rl

HAI_m_rl is the age and sex adjusted residuals of the mortality weighted HAI, HAI_m, used for the linkage analysis.

R code (from R. Minster)

```
# Objective: Calculate HAI residuals in LLFS.

# Clean up workspace

rm(list = ls(all = TRUE))

  # Set working directory

setwd("/Users/rminster/Documents/Professional/bz-llfs-hai/")

  # Load libraries.

  # Read in data

a <- read.csv("../03-llfs-data/llfsdata_csv_20120413/lungfunc.csv")
b <- read.csv("../03-llfs-data/llfsdata_csv_20120413/gtriplet_v2.csv")
c <- read.csv("../03-llfs-data/llfsdata_csv_20120413/codedmeds.csv")
d <- read.csv("../03-llfs-data/llfsdata_csv_20120413/cogassess.csv")
e <- read.csv("../03-llfs-data/llfsdata_csv_20120413/blood.csv")
f <- read.csv("../03-llfs-data/llfsdata_csv_20120413/bphr.csv")
g <- read.csv("../03-llfs-data/llfsdata_csv_20120413/pm.csv")
h <- read.csv("../03-llfs-data/llfsdata_csv_20120413/sdi.csv")
i <- read.csv("../03-llfs-data/llfsdata_csv_20120413/venip.csv")
j <- read.csv("../03-llfs-data/llfsdata_csv_20120413/pershx.csv")
k <- read.csv("../03-llfs-data/llfsdata_csv_20120413/medhx.csv")
```

```
  # Merge data into a single file

data <- merge(a, b, by = "subject")
data <- merge(data, c, by = "subject", all.x = TRUE)
data <- merge(data, d, by = "subject")
data <- merge(data, e, by = "subject")
data <- merge(data, f, by = "subject")
data <- merge(data, g, by = "subject")
data <- merge(data, h, by = "subject")
data <- merge(data, i, by = "subject")
data <- merge(data, j, by = "subject")
data <- merge(data, k, by = "subject")

rm(a, b, c, d, e, f, g, h, i, j, k)

  # Convert data to correct data type

data$sys1 <- as.integer(as.character(data$sys1))
data$sys2 <- as.integer(as.character(data$sys2))

data$creatr <- as.numeric(as.character(data$creatr))

data$glur <- as.integer(as.character(data$glur))

n <- nrow(data)

data$sbp <- (data$sys1 + data$sys2) / 2
data$fvcl <- data$fvc / 1000
data$proband <- as.integer(data$gen == 2 & data$control == 0)
data$offspring <- as.integer(data$gen == 3 & data$control == 0)
data$control <- as.integer(data$gen == 3 & data$control == 1)
data$gender <- data$sex.x
levels(data$gender) <- c(NA, "F", "M")

  # Code from healthiest (0) to least healthy (2)

sbp_t <- rep(NA, n)
sbp_t[data$sbp < 126] <- 0
sbp_t[data$sbp >= 126 & data$sbp < 143] <- 1
sbp_t[data$sbp >= 143] <- 2
sbp_t[data$htnrx == 1] <- 2
sbp_t[data$htdx == 1] <- 2
data <- data.frame(data, sbp_t)

creat_t <- rep(NA, n)
creat_t[data$gender == "F" & data$creatr < 0.8] <- 0
creat_t[data$gender == "F" & data$creatr >= 0.8 & data$creatr <= 1.0] <- 1
creat_t[data$gender == "F" & data$creatr > 1.0] <- 2
creat_t[data$gender == "M" & data$creatr < 1.1] <- 0
creat_t[data$gender == "M" & data$creatr >= 1.1 & data$creatr <= 1.3] <- 1
creat_t[data$gender == "M" & data$creatr > 1.3] <- 2
data <- data.frame(data, creat_t)

fvc_t <- rep(NA, n)
fvc_t[data$gender == "F" & data$fvcl >= 2.61] <- 0
fvc_t[data$gender == "F" & data$fvcl < 2.61 & data$fvcl >= 2.14] <- 1
```

```
fvc_t[data$gender == "F" & data$fvcl < 2.14] <- 2
fvc_t[data$gender == "M" & data$fvcl >= 3.84] <- 0
fvc_t[data$gender == "M" & data$fvcl < 3.84 & data$fvcl >= 3.19] <- 1
fvc_t[data$gender == "M" & data$fvcl < 3.19] <- 2
data <- data.frame(data, fvc_t)

gluc_t <- rep(NA, n)
gluc_t[data$glur < 100] <- 0
gluc_t[data$glur >= 100 & data$glur < 126] <- 1
gluc_t[data$glur >= 126] <- 2
gluc_t[data$X_FASTTIME <= 6] <- NA
gluc_t[data$dmrx > 0] <- 2
gluc_t[data$diab == 1] <- 2
data <- data.frame(data, gluc_t)

 # From Mikeâ€™s code -- from comparison of old data, Amy Matteiniâ€™s codings and CHS

mmse_t <- rep(NA, n)
mmse_t[data$mmsetot > 26] <- 0
mmse_t[data$mmsetot > 23 & data$mmsetot <= 26] <- 1
mmse_t[data$mmsetot <= 23] <- 2
table(mmse_t)
data <- data.frame(data, mmse_t)

 # Subset data to what is needed

data <-data[, c("subject", "offspring", "proband", "control", "gen", "fc.x",
       "X_AGE", "mmse_t", "sbp_t", "creat_t", "fvc_t", "gluc_t", "gender")]
names(data) <- c("id", "o", "p", "c", "generation", "center",
              "age", "mmse", "sbp", "creat", "fvc", "gluc", "sex")
data$center <- as.integer(data$center)
data$center2[data$center == 2] <- 1
data$center2[is.na(data$center2)] <- 0
data$center3[data$center == 3] <- 1
data$center3[is.na(data$center3)] <- 0
data$center4[data$center == 4] <- 1
data$center4[is.na(data$center4)] <- 0
#data[data$c == 1, 3:ncol(data)] <- NA
data <- data[!duplicated(data$id), ]

 # Equal weighting

w1 <- 0.2
w2 <- 0.2
w3 <- 0.2
w4 <- 0.2
w5 <- 0.2

data$hai <- 5 * (w1 * data$mmse +
              w2 * data$sbp +
              w3 * data$creat +
              w4 * data$gluc +
              w5 * data$fvc)

 # Mortality-optimized weighting
```

```
w1 <- 0.42873
w2 <- 0.17085
w3 <- 0.13397
w4 <- 0.23880
w5 <- 0.38386

w <- sum(w1, w2, w3, w4, w5)

w1 <- w1 / w * 5
w2 <- w2 / w * 5
w3 <- w3 / w * 5
w4 <- w4 / w * 5
w5 <- w5 / w * 5

data$hai_m <- w1 * data$mmse +
            w2 * data$sbp +
            w3 * data$creat +
            w4 * data$gluc +
            w5 * data$fvc

table(data$o, data$p, data$c, exclude = NULL, deparse.level = 2)

names(data)[1] <- "subject"
data <- data[complete.cases(data$hai), ]

m <- lm(hai ~ age + sex, data = data, na.action = na.exclude)
data$hai_rl <- residuals(m)

m <- lm(hai_m ~ age + sex, data = data,
      na.action = na.exclude)
data$hai_m_rl <- residuals(m)

write.table(data[data$c == 0, c("subject", "hai_rl", "hai_m_rl")], "3d5c-hai-linkage.csv",
  sep = ",", row = FALSE, quote = FALSE)

pcs <- read.csv("../bl-llfs-genetic-data/2vc-ancestry-pc's/llfseignvec.csv")
i <- read.table("2ve2-genetic-data-ids.txt", header = TRUE)

data <- merge(data[data$subject %in% i$subject, ], pcs[pcs$outlier == 0, ])

m <- lm(hai ~ age + sex + pc1 + pc2 + pc3 + pc4 + pc5 + pc6 + pc7 + pc8 + pc9 +
   pc10, data = data, na.action = na.exclude)
m <- step(m, scope=list(upper = ~ age + sex + pc1 + pc2 + pc3 + pc4 + pc5 + pc6 + pc7 + pc8 + pc9 +
      pc10, lower = ~ age + sex))
summary(m)
data$hai_rg <- residuals(m)

m <- lm(hai_m ~ age + sex + pc1 + pc2 + pc3 + pc4 + pc5 + pc6 + pc7 +
      pc8 + pc9 + pc10, data = data, na.action = na.exclude)
m <- step(m, scope=list(upper = ~ age + sex + pc1 + pc2 + pc3 + pc4 + pc5 + pc6 + pc7 + pc8 + pc9 +
      pc10, lower = ~ age + sex))
summary(m)
data$hai_m_rg <- residuals(m)

write.table(data[, c("subject", "hai_rg", "hai_m_rg")], "3d5d-hai-gwas.csv",
  sep = ",", row = FALSE, quote = FALSE)
```

Telephone Interview for Cognitive Status (TICS) Dataset

This data set is part of the Cognitive Assessments. It was used when an in-person visit was not feasible. This dataset has no derived variables.

Venipuncture (VENIP) Dataset

The Venipuncture dataset contains information about the blood collection, any bleeding disorders, and the shipment information about the tubes. Derived variables were added as well.

_FASTTIME and _FAST
_FASTTIME is the fasting hours. _FAST is the fasting status.

```
if daylast in("1","T") then daylast2=0;
        else if daylast in("2","Y") then daylast2=1;
if daylast in("2","Y") and lasttime=. and drawtime^=. then do;
      lasttime=.; end;
        else lasttime2=lasttime;

     datelast=DHMS(date-daylast2,hour(lasttime2),minute(lasttime),0);
     visittime=DHMS(date,hour(drawtime),minute(drawtime),0);
     _FASTTIME=round(((visittime-datelast)/3600),.01);

     if _FASTTIME < 8 then _FAST=0;
     if 8<=_FASTTIME<24 then _FAST=1;
     if _FASTTIME >=24 then _FAST=2;
     drop daylast2 lasttime2 datelast visittime;
     label _FASTTIME="Fasting hours"
           _FAST="Fasting status 0:<8hrs; 1:8-24hrs; 2:>=24hrs";
```

What Data Collected per Participant (WHATDATA) Dataset

This data set is an inventory of all the forms and reading center data that were collected for each Participant. This dataset has no derived variables.

PHASE II

Follow Up (FOLLOWUP) Dataset

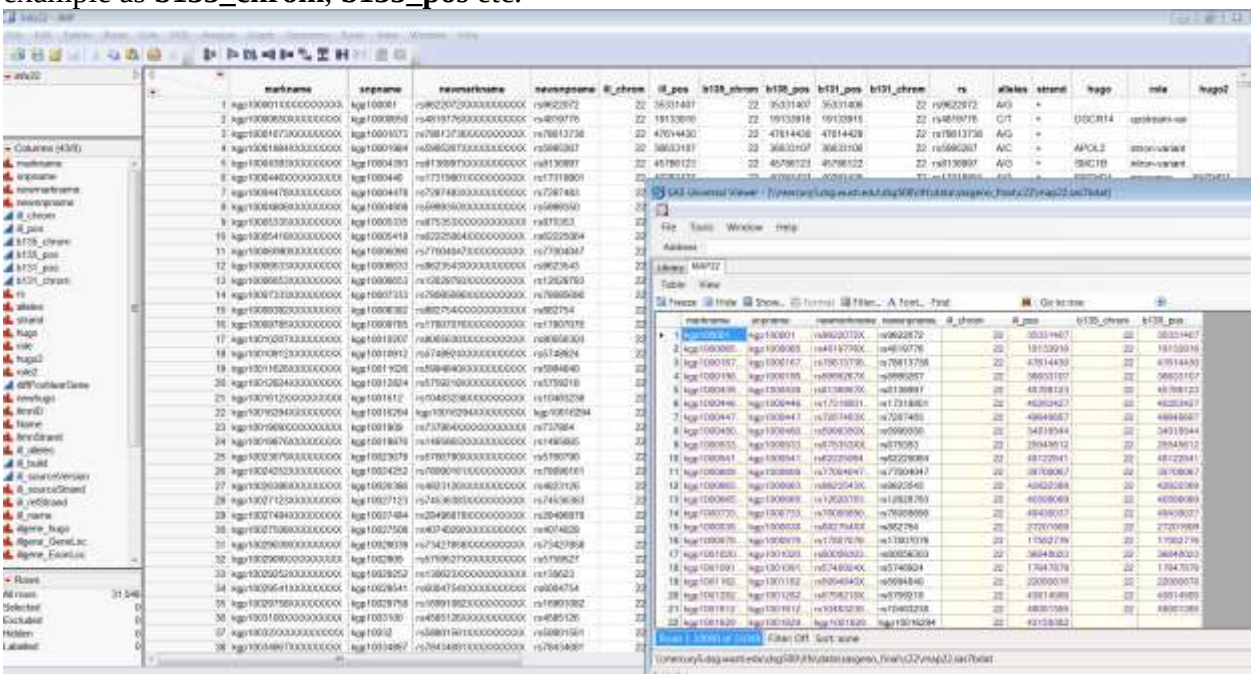
This dataset contains the information from the Follow Up form. It is completed at every follow up occasion. This dataset has no derived variables.

GENOTYPES

Annotation (Info and Map) Datasets

The first version of the annotation file was created based on the gene annotation file Illumina provided. The file corresponding to the chip used in genotyping, namely Human Omni 2_5-8v1 version was used. A large number of markers corresponded with a "kgp" id which was annotating that a particular marker was originated from 1000 Human Genome project. At a later time, an annotation file was provided by CIDR that delivered a correspondence among the "kgp" snps that lacked rs names with now matching rs-names. Therefore, a new annotation is created by merging these new rs names to the existing data. Two file type data are distributed: a) a map file per chromosome; and b) an info file, which have been merged with the NCBI b135 version of dbSNP database to find the most recent version of gene and position annotation. To have a continuation and match with the genotype markers names provided originally by Illumina, the snp names have been kept as the snp identifiers in our database, the improved annotation information by SNP can all be found in the info files for each chromosome.

NOTE: The most important connectors in the data are: **markname** in the map, which is a 20 character long identifier for the SNPs and matched with the genotype, genefreq and info files. The corresponding variable with no padding "XXXXXX" is **snpname**. The next important variable is **newmarkname**, which represent a variable with the most rs-names known in this chip. The corresponding variable with no padding "XXXXXX" is **newsnpname**. The rest of the variables are annotations, such as chromosome, position, gene name, role of the SNP. It will be of interest that one uses the information that comes from the latest NCBI dbSNP annotated for example as **b135_chrom**, **b135_pos** etc.



The above is a picture combination of a view of map22.sas7bdat and info22.sas7bdat.
Following are two tables that summarize the work on markers for the maps and annotation files.

No	ilrs	ilkgp	other	total	new_rs	still_kgp	other	total	b135	b131	illonly	total	diff
1	57,135	126,907	30	**184,072**	168,169	15,873	30	**184,072**	167,408	109	16,162	**183,679**	393
2	55,980	138,146		**194,126**	178,112	16,014		**194,126**	177,299	118	16,249	**193,666**	460
3	45,960	117,712		**163,672**	150,012	13,660		**163,672**	149,541	97	13,872	**163,510**	162
4	39,198	113,647	1	**152,846**	139,835	13,010	1	**152,846**	139,260	86	13,200	**152,546**	300
5	40,847	104,603	3	**145,453**	133,109	12,341	3	**145,453**	132,748	90	12,547	**145,385**	68
6	54,996	99,673	17	**154,686**	142,373	12,296	17	**154,686**	141,543	102	12,725	**154,370**	316
7	36,782	92,253	37	**129,072**	118,479	10,556	37	**129,072**	117,926	72	10,813	**128,811**	261
8	35,929	89,586		**125,515**	115,631	9,884		**125,515**	115,345	90	10,026	**125,461**	54
9	31,739	71,268	4	**103,011**	95,458	7,549	4	**103,011**	95,078	57	7,676	**102,811**	200
10	37,889	81,519		**119,408**	109,694	9,714		**119,408**	109,364	65	9,901	**119,330**	78
11	35,123	80,970	2	**116,095**	106,430	9,663	2	**116,095**	106,099	91	9,816	**116,006**	89
12	34,478	78,219	25	**112,722**	103,177	9,520	25	**112,722**	102,687	92	9,732	**112,511**	211
13	27,034	56,447	2	**83,483**	76,633	6,848	2	**83,483**	76,327	43	7,077	**83,447**	36
14	22,510	54,000		**76,510**	70,360	6,150		**76,510**	70,166	50	6,253	**76,469**	41
15	21,078	51,211	5	**72,294**	66,467	5,822	5	**72,294**	66,291	39	5,916	**72,246**	48
16	21,821	54,787	2	**76,610**	70,874	5,734	2	**76,610**	70,669	35	5,838	**76,542**	68
17	19,260	47,125	2	**66,387**	61,111	5,274	2	**66,387**	60,915	39	5,387	**66,341**	46
18	21,117	47,435		**68,552**	63,465	5,087		**68,552**	63,325	40	5,163	**68,528**	24
19	13,853	33,880		**47,733**	43,767	3,966		**47,733**	43,557	40	4,088	**47,685**	48
20	17,802	38,739	1	**56,542**	52,731	3,810		**56,541**	52,581	25	3,885	**56,491**	51
21	9,940	22,135		**32,075**	29,470	2,605		**32,075**	29,389	16	2,644	**32,049**	26
22	9,837	23,473		**33,310**	31,035	2,275		**33,310**	30,899	13	2,323	**33,235**	75
	690,308	**1,623,735**	**131**	**2,314,174**	**2,126,392**	**187,651**	**130**	**2,314,173**	**2,118,417**	**1,409**	**191,293**	**2,311,119**	**3,055**

Final count of markers included in the data follows:

chrom	LIBNAME	NAME	remobs	NOBS
1	C1	markname	379	**176,754**
2	C2	markname	444	**187,627**
3	C3	markname	154	**158,475**
4	C4	markname	291	**148,058**
5	C5	markname	66	**140,918**
6	C6	markname	288	**148,707**
7	C7	markname	251	**123,975**
8	C8	markname	50	**121,422**
9	C9	markname	188	**98,900**
10	C10	markname	75	**114,826**
11	C11	markname	85	**111,619**
12	C12	markname	198	**108,540**
13	C13	markname	36	**80,896**
14	C14	markname	39	**73,798**
15	C15	markname	44	**69,671**
16	C16	markname	60	**72,736**
17	C17	markname	46	**62,495**
18	C18	markname	24	**66,377**
19	C19	markname	46	**43,783**
20	C20	markname	46	**54,055**
21	C21	markname	23	**30,798**
22	C22	markname	70	**31,048**
Total			**2,903**	**2,225,478**

Info Datasets Variables

markname: Illumina provided locus name (rs number, if available) padded with "XXX" to 20 characters long

snpname: Illumina provided locus name (rs number, if available)

newmarkname: CIDR provided locus name (rs number, if available) padded with "XXX" to 20 characters long

newsnpname: CIDR provided locus name (rs number, if available)

ill_chrom: Illumina provided chromosome number

ill_pos: Illumina provided base pair position

b135_chrom: dbSNP Build 135 chromosome number

b135_pos: dbSNP Build 135 base pair position

b131_pos: dbSNP Build 131 base pair position

b131_chrom: dbSNP Build 131 chromosome number

rs: SNP rs number if found

alleles: dbSNP Build 135 alleles

strand: dbSNP Build 135 strand

hugo: dbSNP Build 135 gene symbol

role: dbSNP Build 135 SNP function class

ABBREV	DESCRIPTION
cds-synon	synonymous change. ex. rs248, GAG->GAA, both produce amino acid: Glu
intron	intron. ex. rs249.
cds-reference	contig reference
synonymy unknown	coding: synonymy unknown
nearGene-3	within 3' 0.5kb to a gene. ex. rs3916027 is at NT_030737.9 pos7669796, within 500 bp of UTR starts 7669698 for NM_000237.2.
nearGene-5	within 5' 2kb to a gene. ex. rs7641128 is at NT_030737.9 pos7641128, with 2K bp of UTR starts 7641510 for NM_000237.2.
STOP-GAIN	changes to STOP codon. ex. rs328, TCA->TGA, Ser to terminator.
missense	alters codon to make an altered amino acid in protein product. ex. rs300, ACT->GCT, Thr->Ala.
STOP-LOSS	changes STOP codon to other non-stop codon
frameshift	indel snp causing frameshift.
cds-indel	indel snp with length of multiple of 3bp, not causing frameshift.
UTR-3	3 prime untranslated region. ex. rs3289.
UTR-5	5 prime untranslated region. ex. rs1800590.
splice-3	3 prime acceptor dinucleotide. The last two bases in the 3 prime end of an intron. Most intron ends with AG.ex.rs193227 is in acceptor site.

splice-5	5 prime donor dinucleotide. 1st two bases in the 5 prime end of the intron. Most intron starts is GU. ex.rs8424 is in donor site.

hugo2: dbSNP Build 135 other strand overlapping gene symbol

role2: dbSNP Build 135 other strand overlapping SNP function class

diffPosNearGene: Distance (bp) to the nearest gene. = 0 if SNP on the gene, < 0, if with lower position (on upstream), > 0 if with higher position (on downstream)

Newhugo: The nearest gene name in ( ) if hugo is missing. If distance to the nearest gene (i.e. diffPosNearGene) > 5 kbp, postfix "_beyond" to gene name.

IlmnStrand: Illumina provided strand

ill_alleles: Illumina provided alleles

ill_build: Illumina provided build version

ill_sourceVersion: Illumina provided source version

ill_sourceStrand: Illumina provided source strand

ill_refStrand: Illumina provided reference strand

illgene_hugo: Illumina provided gene symbol

illgene_GeneLoc: Illumina provided gene location

illgene_ExonLoc: Illumina provided exon location

illgene_CodingStatus: Illumina provided coding status

cidr_chrom: CIDR provided chromosome number

cidr_pos: CIDR provided base pair position

P_HWE: SNP Hardy Weinberg Equilibrium P Value

Callrate: SNP genotyping callrate

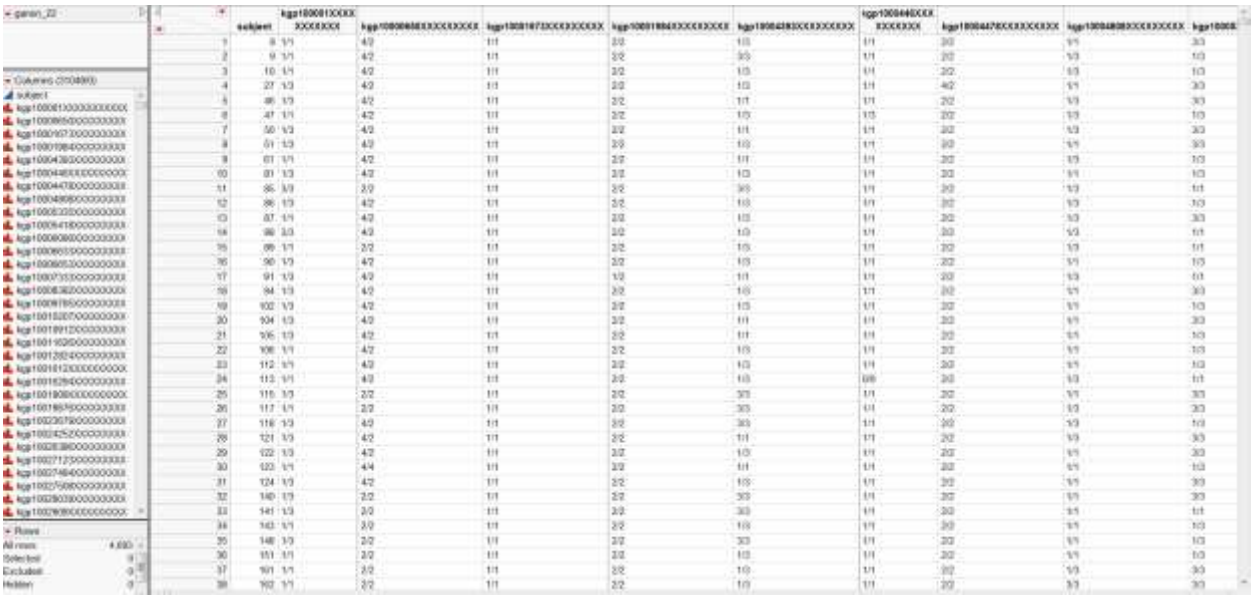
coded_all: GWAS coded allele

noncoded_all: GWAS other allele

coded_af: Allele frequency of the coded allele

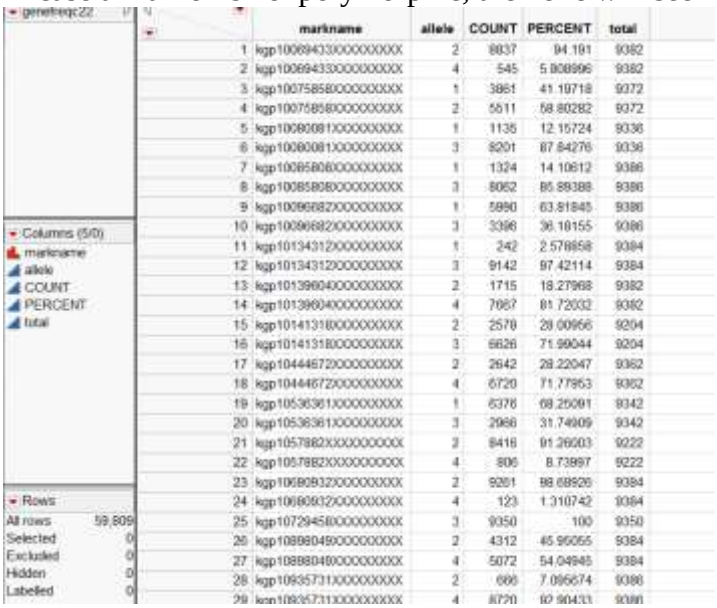
Anonymous Genotypes (GANON) Datasets

These represent the genotype datasets, organized by chromosome. There is one record per Subject and the columns represent the SNP markers. Each cell contains the genotype coded as allele1/allele2, in numerical representation where 1=A, 2=C, 3=G, and 4=T. These datasets are provided in two different formats, SAS and CSV. In addition, we provide CSV formatted subsets, split for conducting parallel programming.



Gene Frequency (GENEFREQ) Datasets

These sets of data (one per chromosome) contain the marker names (MARKNAME), numeric representations of the alleles (ALLELES), the frequency of each allele for each marker in the sample (PERCENT), expressed as a percent, and the number of subjects that carried that allele (COUNT) and total of counts. These data sets have 1 record per allele, usually two per marker. In case a marker is nonpolymorphic, then one will see 1 allele with 100 as its percentage.



	marknam	allele	COUNT	PERCENT	total
1	kgp1006943300000000XX	2	8837	94.191	9382
2	kgp1006943300000000XX	4	545	5.808896	9382
3	kgp1007585800000000XX	1	3861	41.19718	9372
4	kgp1007585800000000XX	2	5511	58.80282	9372
5	kgp1008008100000000XX	1	1135	12.15724	9336
6	kgp1008008100000000XX	3	8201	87.84276	9336
7	kgp1008580800000000XX	1	1324	14.10612	9386
8	kgp1008580800000000XX	3	8062	85.89388	9386
9	kgp1008668200000000XX	1	5990	63.81845	9386
10	kgp1008668200000000XX	3	3396	36.18155	9386
11	kgp1013431200000000XX	1	242	2.578858	9384
12	kgp1013431200000000XX	3	9142	97.42114	9384
13	kgp1013960400000000XX	2	1715	18.27968	9382
14	kgp1013960400000000XX	4	7667	81.72032	9382
15	kgp1014131800000000XX	2	2570	28.00956	9204
16	kgp1014131800000000XX	3	6626	71.99044	9204
17	kgp1044467200000000XX	2	2642	28.22047	9362
18	kgp1044467200000000XX	4	6720	71.77953	9362
19	kgp1053636100000000XX	1	6376	68.25091	9342
20	kgp1053636100000000XX	3	2966	31.74909	9342
21	kgp1057882XXXXXXXXXX	2	8416	91.26003	9222
22	kgp1057882XXXXXXXXXX	4	806	8.73997	9222
23	kgp1068093200000000XX	2	9261	98.68926	9384
24	kgp1068093200000000XX	4	123	1.310742	9384
25	kgp1072945800000000XX	3	9350	100	9350
26	kgp1089804900000000XX	2	4312	45.96055	9384
27	kgp1089804900000000XX	4	5072	54.04945	9384
28	kgp1093573100000000XX	2	666	7.095674	9386
29	kgp1093573100000000XX	4	8720	92.90433	9386

GTRIPLET Dataset

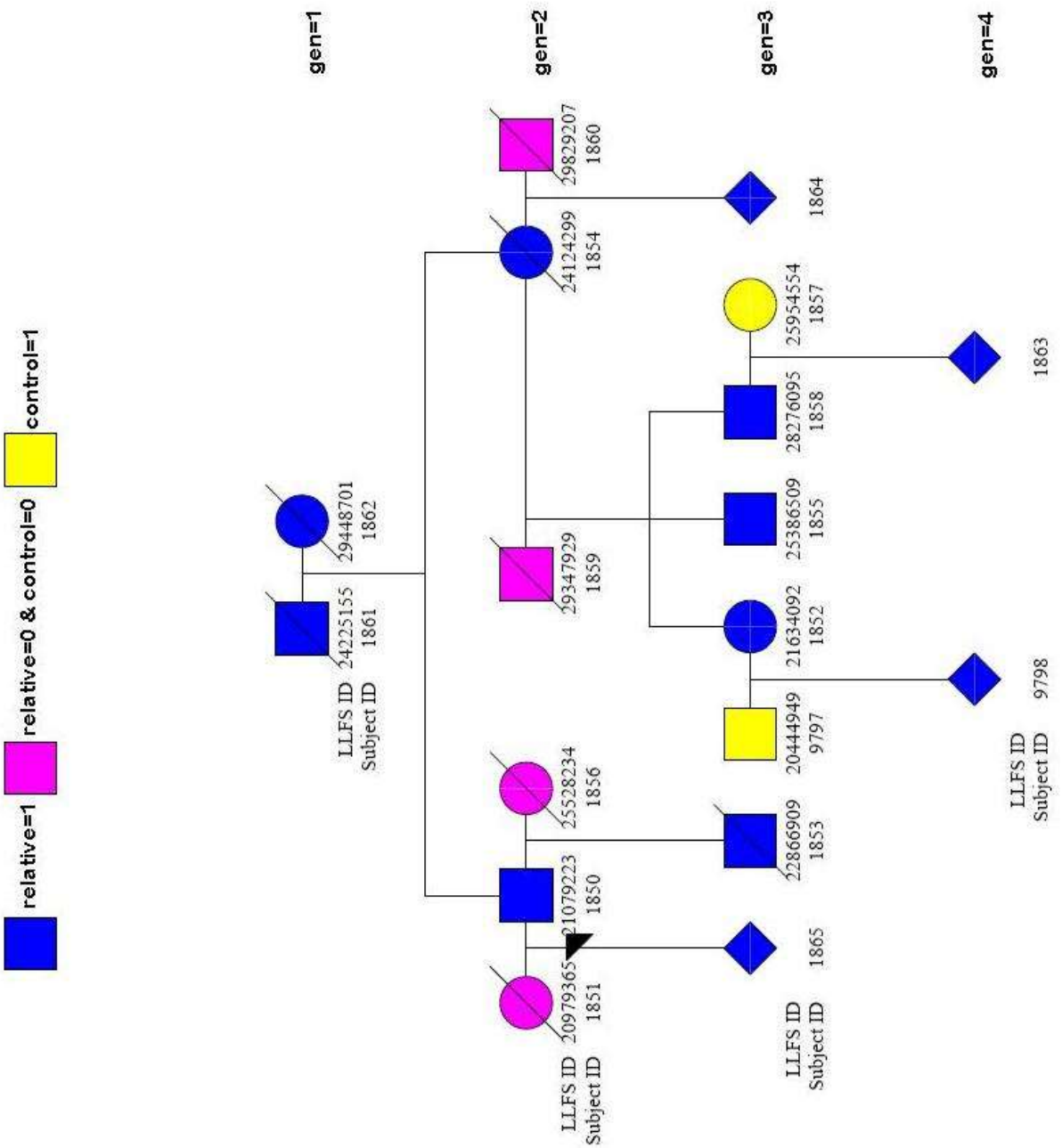
The GTRIPLET dataset reflects the pedigree structures corrected using genetic information and GRR (Graphical Representation of Relationships). Therefore, it is the preferred pedigree

structure for analysis. A triplet is the person, his/her mother, and his/her father. This is all the necessary information needed to determine relatedness.

SUBJECT: the de-identified, unique identifier for each Participant. It is a 5 digit number. Datasets with one obs/subject are uniquely identified by ID and can be merged/linked using this variable.
MOMSUBJ: the SUBJECT number of the person's mother
DADSUBJ: the SUBJECT number of the person's father
Proband_status: the index indicates a subject who is the proband in a pedigree.
gpedid: is the preferred indicator of family membership since it is derived using genetic information.
deceased: vital status.
twinrelatn: twin relationship, MZ or DZ.
relative: the Subject is genetically related to the proband.
control: the Subject is married into the Proband's offspring generation. This person married an offspring of the Proband.
gen: Generation Number.

1 = Proband's Parents Generation
2 = Proband's Generation
3 = Proband's Offspring Generation
4 = Proband's Grandchildren Generation

In the pedigree plot (in the figure on the next page), the color code reflects the values of relative and control. The diagonal line indicates a deceased Subject. The black arrow points to the Proband. Circles are females, squares are males, and diamonds are dummy, placeholder children, to indicate the relationship of a spouse pair without biological children between them.


relative=1
relative=0 & control=0
control=1
gen=1
gen=2
gen=3
gen=4
LLFS ID
Subject ID
2944701
1862
2422515
1861
2982907
1860
2412429
1854
2954554
1857
2827609
1858
1863
2538623
1856
2934792
1859
2538650
1855
2163409
1852
2044494
9797
9798
2107922
1850
2086690
1853
2097936
1851
1865
1864