Documentation: SAQT-GWAS v1.0

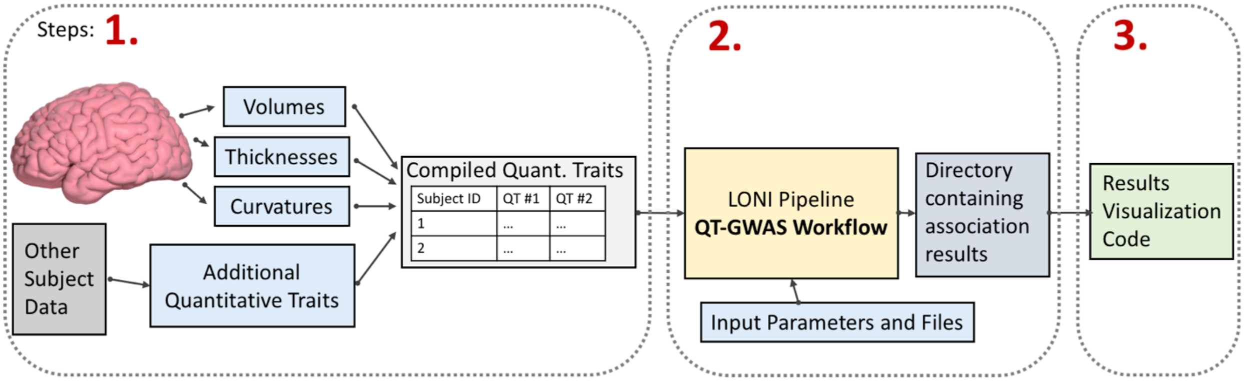
A Semi-Automated Quantitative Trait Genome-Wide Association Study Approach

William Matloff [a], Arthur Toga [a]

[a] *Laboratory of Neuro Imaging, USC Stevens Neuroimaging and Informatics Institute, Keck School of Medicine, University of Southern California, Los Angeles, CA 90032, USA*

Overview

The SAQT-GWAS approach consists of a LONI Pipeline workflow and a collection of R functions that together allow a user to quickly and easily conduct GWAS on hundreds of different quantitative traits simultaneously and subsequently visualize the results. This documentation details how to use this approach, which simplifies the process of conducting genome-wide association studies on hundreds of different quantitative traits to three steps.


Steps: 1.
Volumes
Thicknesses
Curvatures
Other Subject Data
Additional Quantitative Traits
Compiled Quant. Traits
Subject ID
QT #1
QT #2
1
...
...
2
...
...
2.
LONI Pipeline QT-GWAS Workflow
Directory containing association results
Input Parameters and Files
3.
Results Visualization Code

Step 0: Downloads and Installation

The following software and files must be downloaded and/or installed to use the SAQT-GWAS approach. Optional software/files are indicated as such.

1. LONI Pipeline

The LONI Pipeline is required to run the workflow comprising the second step of the SAQT-GWAS approach. Instructions for getting started with the LONI Pipeline can be found at http://pipeline.loni.usc.edu/.

2. R

R is used for the visualization of the association results. R can be installed from https://www.r-project.org/.

The SAQT-GWAS code depends on the following libraries which can be installed using the R Package Installer:

- devtools (v1.12.0)
- plotly (v4.5.6)
- plyr (v1.8.4)
- dplyr (v0.5.0)
- withr (1.0.2)
- roxygen2 (5.0.1)

Manhattanly, a library used for producing interactive Manhattan plots, can be downloaded from https://github.com/sahirbhatnagar/manhattanly by pressing "Clone or download", downloading the zip file, and unzipping it to the directory of choice. In the provided R code, it is necessary to specify the path of this directory.

3. saqt_gwas_v1.pipe

The LONI Pipeline workflow that conducts the genome-wide association studies on the user-provided quantitative traits and genetics data.

4. saqt_gwas_v1.R

The R code for visualizing the association results. Before use, line 8 of the code,

```
load_all('/path/to/manhattanly')
```

must be changed to reflect the path of the Manhattanly directory that was previously downloaded. The entails changing the path name. Details on how to use the functions contained in this script are presented later in the documentation.

5. saqt_gwas_jupyter_v1.R

Version of R code for use in a Jupyter notebook. Line 8 must also be changed to reflect the path of the Manhattanly directory.

6. Templates (optional)

Template files that have the first necessary lines of code for exploring and visualizing the association results using the provided R functions.

saqt_gwas_template.R is an R file that shows the first lines of code necessary for using the functions in an R console.

saqt_gwas_template.ipynb is a Jupyter notebook that serves as a starting point for using Jupyter to visualize the association results. Before use, ensure that the Jupyter kernel is set to R.

7. Jupyter (optional)

Jupyter is a web application that allows a user to create documents that contain both code and the output of that code. These "notebooks" are an excellent way for exploring and sharing association results. Installation instructions can be found at: https://jupyter.org/install.html. We recommend installing Jupyter using Anaconda and conda.

IRKernel (https://github.com/IRkernel/IRkernel) must be installed to run R code in a Jupyter notebook. The r-essentials package should also be downloaded to provide access to the necessary R packages (https://anaconda.org/r/r-essentials). This blog post describes the process in detail: https://www.continuum.io/blog/developer/jupyter-and-conda-r.

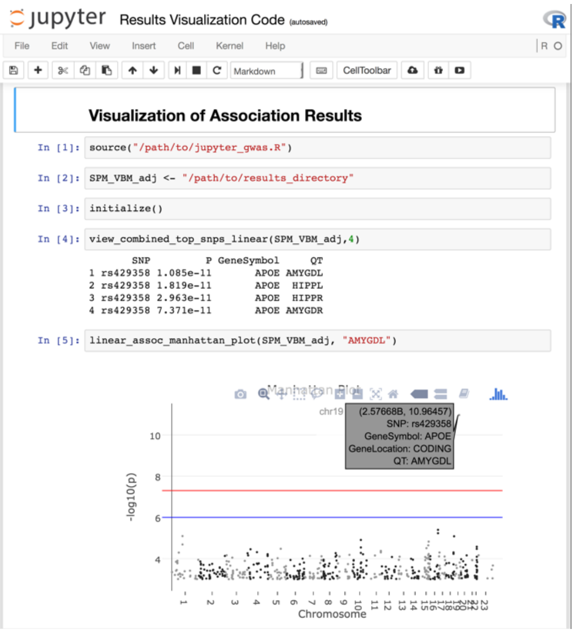
Next, the Plotly library must be installed. To do this, in a Jupyter notebook, run the following lines of code:

```
library(devtools)
devtools::install_github("ropensci/plotly")
```

On a Mac, it may be necessary to first set the environmental variable TAR to where tar is located (found by typing in "which tar" in Terminal). This can be done with:

```
Sys.setenv(TAR='/usr/bin/tar')
```

An example Jupyter notebook is shown below:



Visualization of Association Results

```
In [1]: source("/path/to/jupyter_gwas.R")

In [2]: SPM_VBM_adj <- "/path/to/results_directory"

In [3]: initialize()

In [4]: view_combined_top_snps_linear(SPM_VBM_adj,4)

          SNP         P GeneSymbol     QT
1 rs429358 1.085e-11       APOE AMYGDL
2 rs429358 1.819e-11       APOE  HIPPL
3 rs429358 2.963e-11       APOE  HIPPR
4 rs429358 7.371e-11       APOE AMYGDR

In [5]: linear_assoc_manhattan_plot(SPM_VBM_adj, "AMYGDL")
```

8. RStudio/RStudio Notebooks (optional)

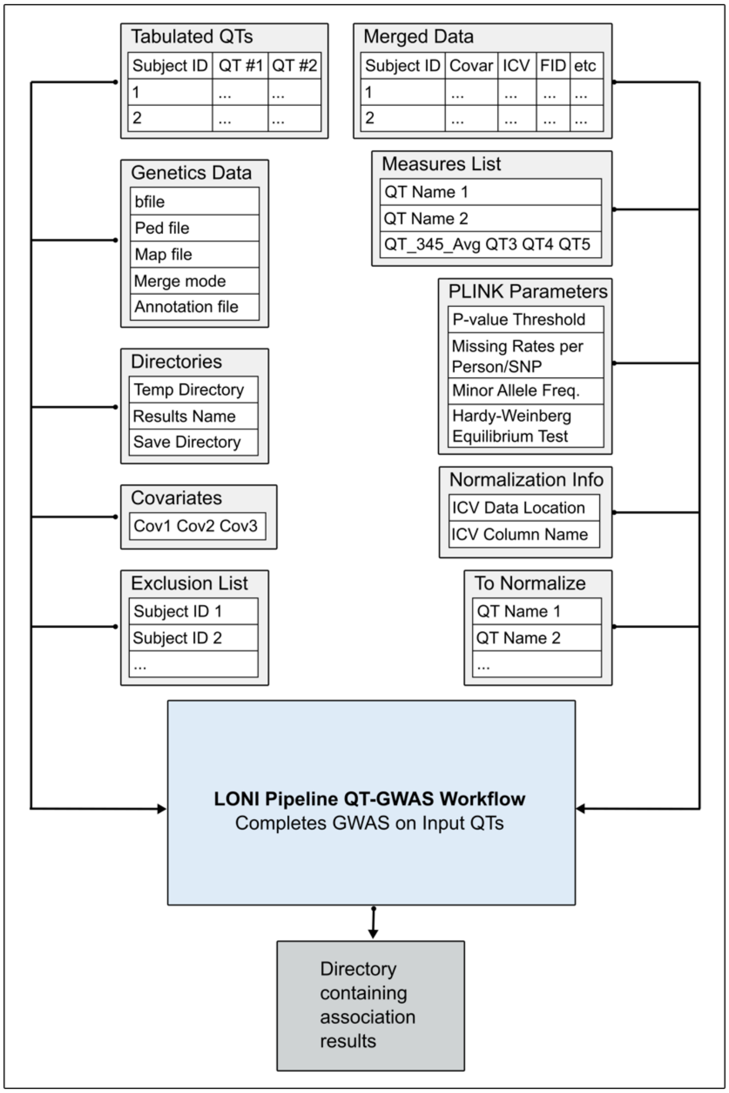
RStudio is an excellent, free, and open-source IDE that can be used to run R:

https://www.rstudio.com/products/rstudio/download/.

RStudio Notebooks provide similar functionality to Jupyter notebooks. It may be easier to get started with RStudio Notebooks than Jupyter.

Step 1: Preparing Workflow Inputs and Parameters

The Inputs:



There are 22 adjustable inputs to the workflow. These inputs are changed from within the LONI Pipeline. Several of the inputs are files that must be created.

1. Covariates
    - String input
    - Covariate names on 1 line separated by spaces
    - Names should match a column name in the Merged Data input file
    - Example: AGE ICV HAND
    - The workflow does not currently support having no covariates

2. Temp Directory
    - String input
    - Path to directory for storing intermediate workflow output
    - If using the LONI cluster, this should be {$tempdir}/{$username}
    - At the beginning and end of the workflow, the directory {Temp Directory}/{Results Name} gets removed. Be cautious when specifying these two string variables so that this directory is not an existing one you do not want to be deleted

3. Results Name
    - String input
    - Specifies name of directory that should be created to store results
    - Value should be descriptive of the data set/results
    - Should not have the same name as a folder in the specified temp directory or server save directory (otherwise the existing folder will be deleted)

4. Merged Data
    - File input – input is a file path specifying the location of the file
    - **CSV file** – first row is column name, other rows are values
    - Contains various data for each subject, such as covariate values, FID (Family ID) values, etc.
    - Leftmost column is: subject ID. This should be labeled as PTID
    - Required columns to have:
        - PTID (Subject ID)
        - Covariates
        - ICV (if using an external ICV value for normalization or as a covariate)
        - FID (family ID)
        - Casecontrol (specifies if subject is a case or control)
        - nonorm (column of 1's if planning on not normalizing by ICV)

PTID	Covar #1	ICV	FID	Casecontrol	nonorm

5. Stats File
    - String input specifying path to stats file with quantitative traits
    - **Text file** – tab separated text file
    - First row = column names

- First column = subject ID’s
- Remaining columns = QT values for each subject with row 1 being the measure name (avoid spaces if possible)

Subject ID	QT #1	QT #2	…
Pt1	#	#	#
Pt2	#	#	#

- Quantitative trait data can be obtained from any source (images, clinical data, biospecimen data, etc.)

6. External ICV
   - Number input
   - 1 if want to use ICV as specified in merged data file
   - 0 if want to use ICV as specified in stats file

7. ICV Column Name
   - String input
   - Name of the merged data csv column that you want to use for normalizing specified measures
   - If don’t want to normalize, can use nonorm column of all ones in merged data input

8. Measures
   - File input – input is path
   - **Text file** – each line contains measure name to perform a GWAS on
   - To get an average of multiple measure names – place multiple on a single line separated by a space
     - First is the name of the average and the remaining are those measures you want to be averaged together
   - Measure names (not the average names) must be included in the stats file
   - Example:
     - lh_MeanSensMotor lh_precentral_thickness lh_postcentral_thickness

9. To Normalize
   - File input – input is path
   - **Text file** – each line is a measure that you want to normalize by the specified ICV Column name
   - If don’t want to normalize by ICV, ensure that ICV column name is a column consisting of all 1’s and that the to normalize text file consists of at least 1 measure

10. Subjects to Exclude
    - String input – path to text file
    - Two column **text file** – tab-separated

- First column is FID, second column is subject ID
- This is a useful way of excluding subjects as part of quality control
- To exclude none: include in the text file a FID/ID record that doesn't exist

11. bfile

- String input – path to binary files containing genetics data for the subjects
- Binary files consist of 3 files: .bed, .fam, and .bim files
- Input is just the basename of the file (not the file extension)
- Example: /path/to/bfile_basename
- http://pngu.mgh.harvard.edu/~purcell/plink/data.shtml#bed

12. Ped file

- File input – input is the path to PED file
- PLINK PED file for external genetics data (SNPs not included in the genotyping chip)
    - External genetics data is a Ped/Map file combination
- http://pngu.mgh.harvard.edu/~purcell/plink/data.shtml#ped
- The workflow does not currently support having no external genetic data

13. Map file

- File input – input is the path to MAP file
- PLINK MAP file for external genetics data (SNPs not included in the genotyping chip)
    - External genetics data is a Ped/Map file combination
- http://pngu.mgh.harvard.edu/~purcell/plink/data.shtml#map
- The workflow does not currently support having no external genetic data

14. Merge Mode

- Number input
- Specifies how PLINK handles the merging between the chip genotyping bfile and the additional PED and MAP files
- Merge mode 2 is a commonly used value
- http://pngu.mgh.harvard.edu/~purcell/plink/dataman.shtml#merge

15. Annotation File

- File input – input is the path to annotation text file
- Often available from manufacturer of genotyping chip that was used to produce the genetics data
- Tab-delimited **text file**
- Columns include:

SNP	Chr	Coordinate	Gene Symbol	Gene Location	Exon Location	Coding Status	Amino Acid 1 | Amino Acid 2
…	…	…	…	…	…	…	…

16. Server Save Directory

- String input
- Input should be path for directory on the server where you would like to save the directory containing the results of the QT-GWAS studies
- Before copying the association results to the Server Save Directory, the workflow first deletes any existing files in the directory {Server Save Directory}/{Results Name} if it already exists, so be sure to copy association results to a different directory (such as a local computer) before running the workflow a second time so that you don't overwrite your results

17. Pfilter

- Number input – N (for example .001 for to only get p-values < 1e-3)
- Report only statistics with p-values less than N
- We recommend a Pfilter of .001 of less to decrease time and computer memory needed to create Manhattan plots

18. Missing Rate per Person

- Number input, part of GWAS quality control
- http://pngu.mgh.harvard.edu/~purcell/plink/thresh.shtml

19. Missing Rate per SNP

- Number input, part of GWAS quality control
- http://pngu.mgh.harvard.edu/~purcell/plink/thresh.shtml

20. Minor Allele Frequency

- Number input, part of GWAS quality control
- http://pngu.mgh.harvard.edu/~purcell/plink/thresh.shtml

21. Hardy-Weinberg Equilibrium Test

- Number input, part of GWAS quality control
- http://pngu.mgh.harvard.edu/~purcell/plink/thresh.shtml

22. noweb

- String input
- --noweb to prevent web-based check of PLINK version

The Workflow:

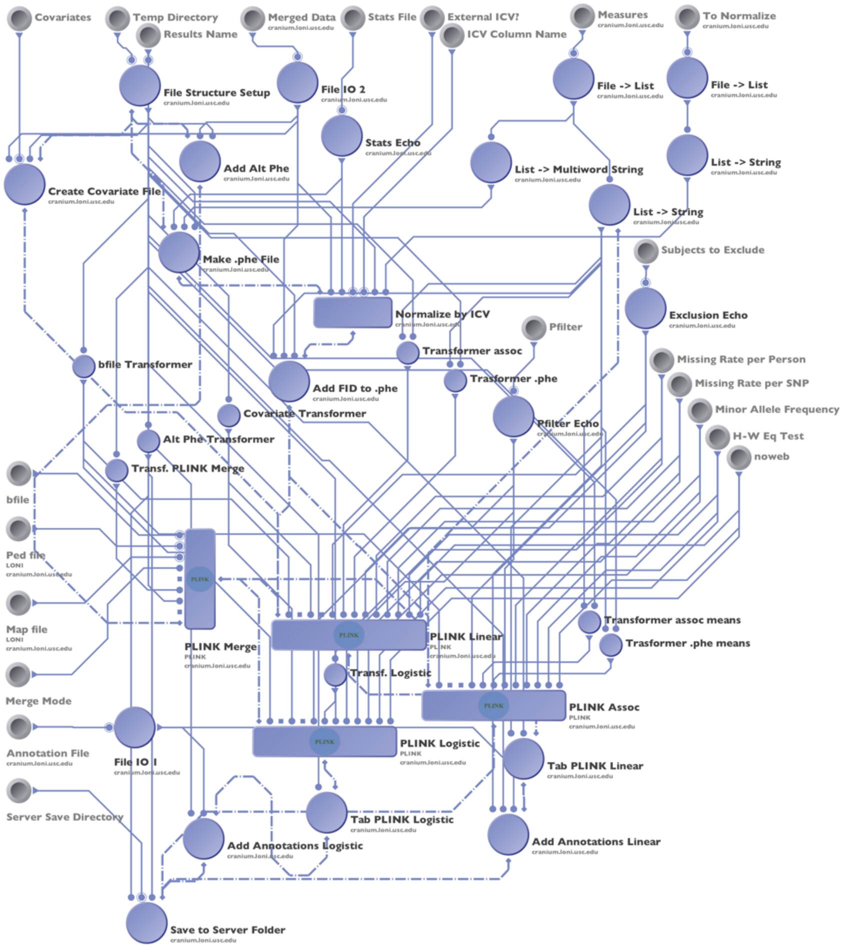
With these inputs, the workflow automatically performs the necessary data transformation and processing steps to run the PLINK Linear and Logistic models for each phenotype, detailed here:

http://pngu.mgh.harvard.edu/~purcell/plink/anal.shtml#glm

These models use an additive model for coding the genotype. The annotated results of these association studies are saved in the specified save directory.

Step 2: Running the Workflow – QT-GWAS_v1.pipe

To run the workflow, first ensure that all the inputs are specified by double clicking on each input and entering the desired value. See above for details about each of these inputs. Next, connect to the LONI Cranium Cluster and run the workflow.



Upon completion, a directory containing the results is created at {Server Save Directory}/{Results Name}. Be sure to copy/move this directory to a separate location for later visualization. Often, users copy the results to a local computer so that they can run the visualization code locally on the results.

Step 3: Visualization

Visualizing the association results is accomplished by using the provided R functions. The R code is designed to function perfectly with the unaltered output of the Pipeline workflow. The code can be run in the R console, in RStudio, in an RStudio Notebook, or a Jupyter notebook.

R console:

Example session of running the code in the R console:

```
> source("/path/to/saqt_gwas_v1.R", print.eval = TRUE)        # load the code
> results_name <- "/path/to/results_directory"    # specify results directory
> initialize()                                         # initialize libraries
```

The template saqt_gwas_template.R contains these lines of code. Next, run various functions as desired, such as:

```
> linear_assoc_manhattan_plot(results_name, “Measure-Name”)
```

Plots will appear in the web browser, as they are created using the interactive Plotly library. Be sure to change the path names to appropriate values.

RStudio console:

The same process as above is used to run this code (saqt_gwas_v1.R) in the RStudio console. With RStudio, the plots appear in the RStudio viewer instead of a web browser.

Jupyter notebook:

With Jupyter successfully installed, running the Jupyter web application is accomplished by opening the terminal and typing in “jupyter notebook”. The web application then starts in the browser and allows a user to find a desired directory and either open an existing Jupyter notebook or create new one. If starting a new notebook, be sure that the notebook is using the R kernel. The template, saqt_gwas_template.ipynb is an easy notebook to start with. To use this, simply open and change it to reflect the locations of the code and results files. Cells of code and text can be run by pressing Shift-Enter. Follow instructions in the terminal to shut down the notebooks, stop the server, and shut down all kernels. On a Mac, this is accomplished by typing Control-C. More information about using Jupyter notebooks can be found at:

https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/

RStudio notebook:

RStudio also has an excellent notebook format that is very easy-to-use. To create a notebook in RStudio, go to File → New File → R Notebook. Add a code chunk with the Add Chunk command. Run the code chunk with the Run Chunk command. More detailed information about creating and using RStudio notebooks can be found at:

http://rmarkdown.rstudio.com/r_notebooks.html

After a code chunk with the following lines is run, additional code chunks can be created to run any of the visualization functions provided in saqt_gwas_v1.R and see the output in the same document. Be sure the change the below pathnames to reflect to correct path.

~~~~
```{r}
source("/path/to/saqt_gwas_v1.R")                   # load the code
results_name <- "/path/to/results_directory"        # specify results directory
initialize()                                        # initialize libraries
```
~~~~

Functions:

```
> initialize()
```

- Load the required libraries for all subsequent code

```
> view_phenotypes(location)
```

- “location” argument is the variable specifying the path of the results directory
- Example: `view_phenotypes(results_name)`
- Output is a list of the phenotype measures that were used in the analysis

```
> linear_assoc_manhattan_plot(location, phenotype)
```

- “location’ argument is the variable specifying the path of the results directory
- “phenotype” is a string indicating one of the phenotype measures
- Example: `linear_assoc_manhattan_plot(results_name, “Left-Amygdala”)`
- Output is a Manhattan plot in the web browser displaying the results of the association study for the specified phenotype

```
> combined_linear_assoc_manhattan_plot(location)
```

- “location” argument is the variable specifying the path of the results directory
- Output is a Manhattan plot displayed in the web browser displaying the association results for all tested phenotypes

```
> view_combined_top_snps_linear(location, number)
```

- “location” argument is the variable specifying the path of the results directory
- “number” argument is the total number of SNPs to display

- Outputs SNPs in order of increasing p-value (smallest p-value is first)
- Considers SNP-Phenotype p-values for each phenotype tested

```
> view_top_snps_linear(location, phenotype, number)
```

- "location" argument is the variable specifying the path of the results directory
- "phenotype" argument is a string variable indicating one of the phenotype measures
- "number" argument is the total number of SNPs to display
- Outputs the top results for the GWAS of the specified phenotype

```
> logistic_assoc_manhattan_plot(location)
```

- "location" argument is the variable specifying the path of the results directory
- Outputs a Manhattan plot in a web browser displaying the results of the logistic association model (testing for SNPs associated with cases vs controls)

```
> view_top_snps_logistic(location, number)
```

- "location" argument is the variable specifying the path of the results directory
- "number" argument is the total number of SNPs to display
- Outputs a list of the most significant SNPs of the logistic association model

```
> phe_by_groups(location, phenotype)
```

- "location" argument is the variable specifying the path of the results directory
- "phenotype" argument is a string variable indicating one of the phenotype measures
- Output is a plot in the web browser showing the phenotype values for each group

```
> view_external_data(ext_data_loc)
```

- "ext_data_loc" is a string argument specifying the path of a csv document containing columns for SNP RS number, gene symbol, p-value, and quantitative trait
- Need to list columns for this:
    - Column names are important
- This csv file is meant for comparison with association results

```
> compare_two(ext_data_loc, location, p_value_name)
```

- "ext_data_loc" is a string argument specifying the path of a csv document containing columns for SNP RS number, gene symbol, p-value, and quantitative trait
- "location" argument is the variable specifying the path of the results directory
- "p_value_name" specifies the P-value column name in the ext_data_loc csv file. This name should be something other than "P" or "p"
- Output shows the external data in order of decreasing $-\log_{10}$(p-value) and the corresponding p-value from the association results specified in the "location" argument