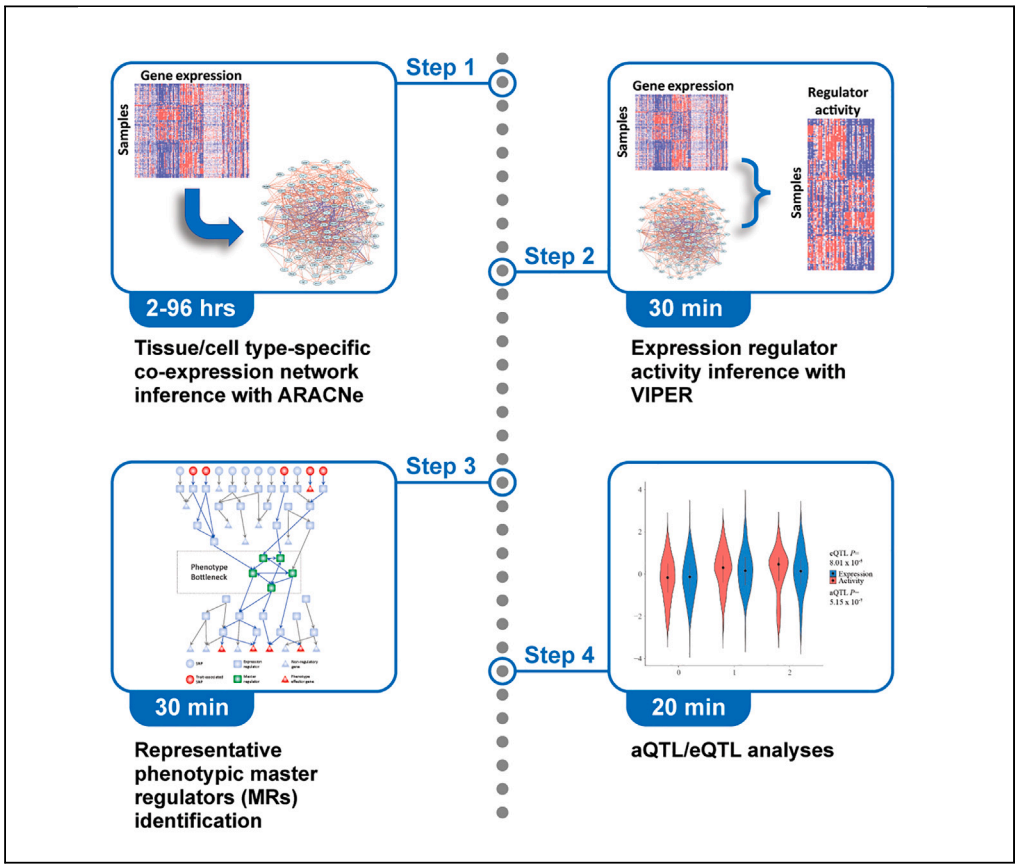
STAR Protocols

CellPress
OPEN ACCESS

Protocol

Master regulator activity QTL protocol to implicate regulatory pathways potentially mediating GWAS signals using eQTL data



Here, we present a protocol to identify transcriptional regulators potentially mediating downstream biological effects of germline variants associated with complex traits of interest, which enables functional hypothesis generation independent of colocalizing expression quantitative trait loci (eQTLs). We describe steps for tissue-/cell-type-specific co-expression network modeling, expression regulator activity inference, and identification of representative phenotypic master regulators. Finally, we detail activity QTL and eQTL analyses. This protocol requires genotype, expression, and relevant covariables and phenotype data from existing eQTL datasets.

Jason W. Hoskins, Trevor A. Christensen, Laufey T. Amundadottir

jason.hoskins@nih.gov (J.W.H.)
amundadottirl@nih.gov (L.T.A.)

Highlights

Identifying genes potentially mediating GWAS signals remains an outstanding challenge

Master regulators (MRs) integrate genetic/environmental info to establish a cell state

MR *trans*-QTL analyses reduce multiple testing burden while enriching for relevant genes

MRaQTL R package streamlines the approach, empowering post-GWAS hypothesis generation

Hoskins et al., STAR Protocols 4, 102362
September 15, 2023
https://doi.org/10.1016/j.xpro.2023.102362

STAR Protocols

CellPress
OPEN ACCESS

Protocol

Master regulator activity QTL protocol to implicate regulatory pathways potentially mediating GWAS signals using eQTL data

Jason W. Hoskins,[1,2,3,*] Trevor A. Christensen,[1] and Laufey T. Amundadottir[1,*]

[1]Laboratory of Translational Genomics, Division of Cancer Epidemiology and Genetics, National Cancer Institute, National Institutes of Health, Bethesda, MA 20892, USA

[2]Technical contact

[3]Lead contact

*Correspondence: jason.hoskins@nih.gov (J.W.H.), amundadottirl@nih.gov (L.T.A.)
https://doi.org/10.1016/j.xpro.2023.102362

SUMMARY

Here, we present a protocol to identify transcriptional regulators potentially mediating downstream biological effects of germline variants associated with complex traits of interest, which enables functional hypothesis generation independent of colocalizing expression quantitative trait loci (eQTLs). We describe steps for tissue-/cell-type-specific co-expression network modeling, expression regulator activity inference, and identification of representative phenotypic master regulators. Finally, we detail activity QTL and eQTL analyses. This protocol requires genotype, expression, and relevant covariables and phenotype data from existing eQTL datasets.

For complete details on the use and execution of this protocol, please refer to Hoskins et al.[1]

BEFORE YOU BEGIN

This protocol will walk you through the process of inferring a tissue-specific co-expression network, inferring expression regulator activities, identifying representative phenotypic master regulators (MRs), and performing expression and activity QTL analyses. The co-expression network inference by ARACNe has been tested within a Linux operating system (each bootstrap run with one 2.4 Ghz Intel E5-2680v4 CPU and 8GB memory), though there are other implementation offered by the authors of that software for other operating systems. The rest of the analyses have been implemented in a new R package called MRaQTL that should work in an R (v4.2 or higher) environment on any popular operating system. This protocol will be demonstrated with two different datasets: (1) a toy dataset derived from the open-access GEUVADIS eQTL data[2] with a simulated phenotype, and (2) the TwinsUK adipose eQTL dataset as used in Hoskins et al. (2021).[1] The provided GEUVADIS toy dataset is open access and provides examples of expected input file formats but yields biologically meaningless results. Conversely, the TwinsUK adipose dataset demonstrates a polished analysis with biologically meaningful results but the required expression, phenotype and covariables data are controlled access. If the reader prefers, they may instead follow along with the protocol using their own data formatted as for the toy data files. The times for each step using the MRaQTL package are based on run times with the GEUVADIS toy dataset run on a local computer with an 1.80 GHz Intel i7-1265U CPU and 32 GB of memory, but these do not represent the minimum requirements. Running this protocol with the GEUVADIS dataset requires a total of ~500 MB of hard drive space. Begin by downloading deposited data and installing the required software as described below and summarized in the key resources table.

Gaining access to and downloading input data [optional]

Timing: ≥1 week

The demonstration of this protocol for a "real world" analysis uses controlled-access data. Therefore, to replicate this particular analysis, you must first apply for access to, and download, the input data. However, this section (steps 1 and 2) may be entirely skipped if the reader is using the GEUVADIS toy dataset or their own data.

1. Complete the data access application on the TwinsUK Study website and await approval.
   a. Follow the instructions at https://twinsuk.ac.uk/resources-for-researchers/access-our-data/.
   b. Request the filtered and formatted RNA-seq, genotypes, BMI measurements and covariables associated with this protocol.
   c. Unless you have need of this data for other research purposes, specify that your aims for the data are the replication of this protocol.
2. Download controlled access input data.

Note: Once your application is approved you will be given access to download the expression data (log_2TPMs for 13,776 expressed genes for the 699 subjects), genotype data (imputed alternate allele dosages filtered for 40,486 BMI GWAS significant variants [$P < 5 \times 10^{-8}$] for the 699 subjects), BMI data (measured at time of biopsy for the 699 subjects), and covariates data (age at time of biopsy and 5 eigenvalues for population substructure for the 699 subjects).

Downloading freely available data files

Timing: 5 min

This protocol may also be followed using input data derived from the GEUVADIS eQTL dataset[2] with unrestricted access that are available for download from a GitHub repository. Even if the reader is using their own data, it is recommended to download these input files as examples of the expected file formats.

3. Download freely available input data.
   a. Download the GEUVADIS toy data (expression, genotype, covariables, simulated phenotype, gene and SNP map files), the expression regulator list, and (optionally) the TwinsUK (aka Eurobats) gene and SNP map files and ARACNe co-expression network from GitHub: https://github.com/hoskinsjw/aQTL_STAR_protocol/ (https://doi.org/10.5281/zenodo.7929966).
   b. Alternatively, if using a Linux system with Git installed, simply clone the entire GitHub repository with the following command.

```
> git clone https://github.com/hoskinsjw/aQTL_STAR_protocol
```

Installing software and R packages

Timing: 15 min

Required software for this protocol must be installed.

4. Download the ARACNe Java executable jarfile or build it from GitHub.

a. ARACNe Java executable jarfile is available for download from http://califano.c2b2.columbia.edu/aracne. With the jarfile, no additional installation is required, though it does require Java 1.8 (or higher) to run.
b. Alternatively, the Linux jarfile may be built from the GitHub repository within a Linux shell as follows.

```
> git clone https://github.com/califano-lab/ARACNe-AP

> cd ARACNe-AP

> ant main
```

i. Note that ARACNe requires Apache ANT to build and Java 1.8 (or higher) to both build and run.
ii. The resulting executable arcane.jar file can be found in the /ARACNe-AP/dist/ directory and may be moved/copied to the working directory with your data files or added into your PATH.
iii. For installation troubleshooting, go to https://github.com/califano-lab/ARACNe-AP.

5. Download and install R (version 4.2.0 or current version).

Note: R is available for all common operating systems (with documentation) from https://www.r-project.org/. RStudio is an optional GUI/IDE that, among other things, makes it easier to interact with R, and find and install R packages. It is available for all popular operating systems with documentation from https://posit.co/download/rstudio-desktop/.

6. Start an R session (either standalone or in RStudio). Install and load the devtools package, and then install the MRaQTL package (https://doi.org/10.5281/zenodo.7930026) with the following commands.

```
> install.packages("devtools")

> library(devtools)

> install_github(''hoskinsjw/MRaQTL'')
```

a. Confirm if asked for permission to update any package dependencies.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Expression_regulator_list_from_Hoskins_et_al_2021.txt (gene symbols for TFs, co-TFs, and signal transduction factors)	This paper; Hoskins et al.[1]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_filtered_samples_WHR_significant_bi-allelic_SNPs.dosage	This paper; Lappalainen et al.[2]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_WHR_significant_bi-allelic_SNPs_locations_in_GRCh37.map	This paper; Lappalainen et al.[2]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_filtered_samples_expressed_genes_logRPKMs_with_symbols.txt	This paper; Lappalainen et al.[2]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_expressed_genes_locations_in_GRCh37.map	This paper	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_filtered_samples_select_covars.txt	This paper; Lappalainen et al.[2]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_simulated_phenotype.txt	This paper	https://github.com/hoskinsjw/aQTL_STAR_protocol/
GEUVADIS_100boots_ARACNe_network.txt	This paper	https://github.com/hoskinsjw/aQTL_STAR_protocol/

(*Continued on next page*)

Continued

REAGENT or RESOURCE	SOURCE	IDENTIFIER
[Optional] Eurobats_adipose_expressed_genes_logTPM_from_Hoskins_et_al_2021.txt (log_2TPM values)	Hoskins et al.[1]; Buil et al.[3]	https://twinsuk.ac.uk/resources-for-researchers/access-our-data/
[Optional] BMI_significant_SNPs_QCd_filtered.dosage (imputed alternate allele dosage)	Hoskins et al.[1]; Buil et al.[3]; Yengo et al.[4]	https://twinsuk.ac.uk/resources-for-researchers/access-our-data/
[Optional] Eurobats_filtered_BMI_data_from_Hoskins_et_al_2021.txt (BMI)	Hoskins et al.[1]; Buil et al.[3]	https://twinsuk.ac.uk/resources-for-researchers/access-our-data/
[Optional] Eurobats_filtered_and_formatted_covariables_from_Hoskins_et_al_2021.txt (Age, and 5 population substructure eigenvalues)	Hoskins et al.[1]; Buil et al.[3]	https://twinsuk.ac.uk/resources-for-researchers/access-our-data/
[Optional] Eurobats_adipose_expressed_genes_locations_in_GRCh37.map (Gene mapping file based on GRCh37)	This paper; Hoskins et al.[1]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
[Optional] BMI_significant_SNPs_locations_in_GRCh37.map (SNP mapping file based on GRCh37)	This paper; Hoskins et al.[1]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
[Optional] Eurobats_adipose_300bootstraps_ARACNe_network.txt (pre-made co-expression network)	This paper; Hoskins et al.[1]	https://github.com/hoskinsjw/aQTL_STAR_protocol/
Software and algorithms		
ARACNe Java executable jarfile	Margolin et al.[5]; Lachmann et al.[6]	http://califano.c2b2.columbia.edu/aracne-license or https://github.com/califano-lab/ARACNe-AP
R (version 4.2.0 or higher)	The R Foundation	https://www.r-project.org/
RStudio (current version)	RStudio	https://posit.co/download/rstudio-desktop/
devtools (R package)	RStudio	https://devtools.r-lib.org/
MRaQTL (R package)	This paper	https://github.com/hoskinsjw/MRaQTL

STEP-BY-STEP METHOD DETAILS

Here we describe step-by-step how to infer the tissue-specific ARACNe co-expression network, infer expression regulator activities with VIPER, identify representative phenotypic master regulators (MRs), and run eQTL and aQTL analyses. To demonstrate this protocol, we will use lymphoblastoid cell line expression and genotype data from GEUVADIS with a simulated phenotype. Results from controlled access adipose expression, genotype, and BMI data from the TwinsUK Study as used in Hoskins et al. (2021)[1] are also shown as examples from a more polished and biologically meaningful analysis. Alternatively, the reader may use their own data while following the protocol. Required formats for input files are noted throughout the protocol and demonstrated by the provided GEUVADIS data files.

Inferring ARACNe co-expression network

Timing: 2–96 h (for 462 samples in GEUVADIS toy dataset with 100 bootstraps run all in parallel or serially, respectively)

Gene regulatory networks dynamically evolve in response to the cell's microenvironment and importantly determine the cell state.[7–10] Therefore, when leveraging a co-expression network to infer expression regulator activities, it is important to use a network relevant to the cell type or tissue under study. ARACNe (Algorithm for the Reconstruction of Accurate Cellular Networks) is a tool designed to infer gene regulatory networks from transcriptomic data sets based on the mutual information between the genes' expression.[5,6] In this step we will infer a lymphoblastoid cell line-specific ARACNe network from log_2 transformed, RPKM normalized RNA-seq data from the GEUVADIS Study using the Java executable for ARACNe-AP within a Linux shell.

1. Navigate to your working directory containing the ARACNe executable jarfile, the expression data, and expression regulator list file.
2. Calculate the mutual information threshold for the expression data.

```
> java -Xmx16G -jar aracne.jar -e GEUVADIS_filtered_samples_expressed_genes_logRPKMs_
with_symbols.txt -o GEUVADIS_100boots --tfs Expression_regulator_list_from_Hoskins_
et_al_2021.txt --pvalue 1E-8 --seed 1 --calculateThreshold
```

⚠ CRITICAL: We have provided an expression regulator list as gene symbols that includes transcription factors, transcription co-factors, and signal transduction factors in humans. However, the user may use their own defined list of expression regulators using whatever gene identifiers they prefer (e.g., Entrez IDs, Ensembl Gene IDs, etc.). Indeed, if the user is working with non-human data, they must generate their own list of expression regulators for their species of study using gene identifiers matching their transcriptomic data to use as the input for the --tfs argument. The regulator list file is a simple text file with one gene identifier per line.

Note: The arguments used in this command are as follows: -e is the expression data file, -o is the name of the desired output directory, --tfs is the file listing expression regulator names, --pvalue is the *P*-value threshold for network edge inclusion (1E-8 is standard), --seed is the random number generator seed number that allows for reproducibility of inferences, and --calculateThreshold sets the mode to calculate the mutual information threshold for network edge inclusion, which is required for subsequent bootstrap network inferences. Please see the ARACNe-AP documentation at https://github.com/califano-lab/ARACNe-AP for further details regarding its arguments.

Note: ARACNe network inference should be run with expression data for at least 100 samples of the same tissue or cell type, though higher sample numbers would improve statistical power for detecting significant mutual information between genes, thereby yielding a denser network.[6] While homogeneity between samples will allow for the inference of a more context-specific network, there is a balance to be struck between homogeneity of samples and sufficient variance in the expression of regulators and target genes to enable detection of mutual information. Therefore, using samples of the same type but with diversity among their genetics and/or environmental exposures would likely yield a denser and more broadly applicable tissue or cell type-specific co-expression network. However, heterogeneous cellularity within samples is not ideal as it results in a network representing a blending of multiple cell types that may not yield accurate activity inferences with VIPER when applied to samples that greatly deviate from the average cellular distribution among samples used to infer the ARACNe network.

Note: The RNA-seq expression values should be adequately normalized to account for library size and gene length (e.g., RPKM, FPKM, or TPM). If using a sample type likely to have significant RNA composition bias (i.e., a large overrepresentation of reads mapping to a small subset of genes), as commonly occurs for example with glandular epithelial tissue, then other normalizations may be warranted (e.g., GeTMM).[11] Log_2 transformation may also be desirable to reduce heteroskedasticity within the data. The expression data file should be formatted as a tab-delimited text file with genes by rows and samples by columns with gene IDs in the first column and a header as the first row that includes sample IDs.

Note: It is recommended that the expression data be filtered to exclude genes with very low to no expression across all samples prior to ARACNe network inference. The user may use their discretion for expression filtering, but one approach is to keep only genes with >0.1 TPM in $\geq$20% of samples AND $\geq$6 read counts (unnormalized) in $\geq$20% of samples, as described for GTEx v8 expression data analysis (https://gtexportal.org/home/datasets).

3. From the same directory, run 100 bootstrap network inferences (each with a different --seed value) using commands of the following form.

```
> java -Xmx16G -jar aracne.jar -e GEUVADIS_filtered_samples_expressed_genes_logRPKMs_with_symbols.txt -o GEUVADIS_100boots --tfs Expression_regulator_list_from_Hoskins_et_al_2021.txt --pvalue 1E-8 --seed 1
```

⚠ CRITICAL: This step is the most computationally intensive of the protocol. It would be most efficient to use a computer cluster to which the 100 individual bootstrap ARACNe commands (each with the seed flag set to a different number from 1 to 100) may be submitted and run in parallel. Please consult the documentation for your computer cluster for details on submitting parallel jobs. Trying to run this many bootstrap networks serially would require ~96 h in total. Therefore, for any readers that do not currently have access to a computer cluster, or who simply wish to conserve computational resources and time, we have made the resulting consolidated 100 bootstrap GEUVADIS lymphoblastoid (and 300 bootstrap TwinsUK adipose) ARACNe network available in the GitHub repository as described in the key resources table.

Note: The authors of ARACNe recommend a minimum of 100 bootstraps.[6] However, our testing indicated that with sufficient sample size, increasing the number of bootstraps increases the density of the final consolidated ARACNe network (in so far as the sampling space has not been saturated). The significance of network edges added by increasing bootstraps will be lower than those present with fewer bootstraps, but the contribution to VIPER's inference of regulator activities for each target gene is weighted by the significance of the corresponding edge. Consequently, adding more edges with diminishing significance does not greatly alter inferred activities. However, regulator activities will only be inferred for regulators with at least 25 target genes in the ARACNe network to ensure robustness of the inference, which means denser networks do enable activity inferences for more total regulators. For the TwinsUK adipose expression data we found that a 100 bootstraps network yielded activity inferences for 2,739 regulators, 300 bootstraps yielded inferences for 3,928 regulators, and 900 bootstraps yielded inferences for 4,213 regulators. Despite these differences, among the regulators with activities inferred from all networks, the correlations among the activities based on networks with differing bootstraps were all very high (Pearson $r > 0.98$), which is consistent with VIPER's previously demonstrated robustness.[12]

4. Consolidate the bootstrap networks into a single, final ARACNe network, which we rename and copy without its header (for a subsequent step) using the following commands.

```
> java -Xmx32G -jar aracne.jar -o GEUVADIS_100boots --consolidate

> mv ./GEUVADIS_100boots/network.txt ./GEUVADIS_100boots/GEUVADIS_100boots_ARACNe_network.txt

>awk 'BEGIN{OFS="\t"}NR>1' ./GEUVADIS_100boots/GEUVADIS_100boots_ARACNe_network.txt > ./GEUVADIS_100boots/GEUVADIS_100boots_ARACNe_network_no_header.txt
```

Inferring regulator activities with VIPER

Timing: 30 min

The VIPER algorithm integrates enrichment of expected expression changes among a regulator's target genes with confidence in the regulator-target network edges and target overlap between different regulators (i.e., pleiotropy) to infer regulator activities from the expression of downstream target genes.[12] In this step we use the VIPER algorithm as implemented in the MRaQTL package to infer the activities of 4,046 expression regulators based on the GEUVADIS expression data and the

100 bootstraps GEUVADIS lymphoblastoid cell line ARACNe network inferred in the previous step. However, we will also show figures generated from the TwinsUK adipose dataset as an example of a more polished analysis. If the reader would like to replicate the TwinsUK results or use their own data, they may simply replace the corresponding file names throughout the following steps.

5. Start an R/RStudio session and, using the following commands, load the MRaQTL package, and set your working directory to that containing the expression data, covariables data, and the tissue-specific ARACNe network without a header (as generated in **Step 4**).

```
> library(MRaQTL)

> setwd("/path/to/data/")
```

6. Use the prepActExp() function to run the regulator activity inference with VIPER, check the distribution of gene-wise correlations between regulator expression and activity, and prepare and output the expression and activity data for subsequent eQTL/aQTL analyses.

```
> prepActExp(

"GEUVADIS_filtered_samples_expressed_genes_logRPKMs_with_symbols.txt","GEUVADIS_
filtered_samples_select_covars.txt",

"GEUVADIS_100boots_ARACNe_network_no_header.txt",

"Toy_data")
```

Note: The four arguments used in this command correspond to the expression data file, the covariables file, the ARACNe network file (without a header), and a prefix string for labeling the output files, in that order. For full details on the function and its arguments, enter "?prepActExp" in the R console or search for the function in the RStudio Help tab.

Note: Depending on the OpenBLAS library version currently installed on the system, the user might receive the error "return code from pthread_create() is 22." In this case, the reader should manually install the preprocessCore package with threading disabled using the following command in their R session:

```
> BiocManager::install("preprocessCore", configure.args="--disable-threading", force = TRUE)
```

Note: If the reader opted to use a pre-made ARACNe network, they must first save a copy of it without a header to use it as the network input for the prepActExp() function. An awk command for doing this was provided in **Step 4**, but the file is likely small enough that the header could be removed manually using a standard text editor.

Note: In this step the covariables file is only used by the R script to ensure the output activity and normalized expression data files have the same samples in the same order, which is important for the eQTL and aQTL analysis steps later in the protocol. The covariables file should be formatted as a tab-delimited text file with covariables by rows and samples by columns with covariable names in the first column and a header as the first row that includes sample IDs.

Note: In addition to the normalized expression and activity data, the script outputs a density plot for all paired correlations between each regulator's expression and activity (see Figure 1).

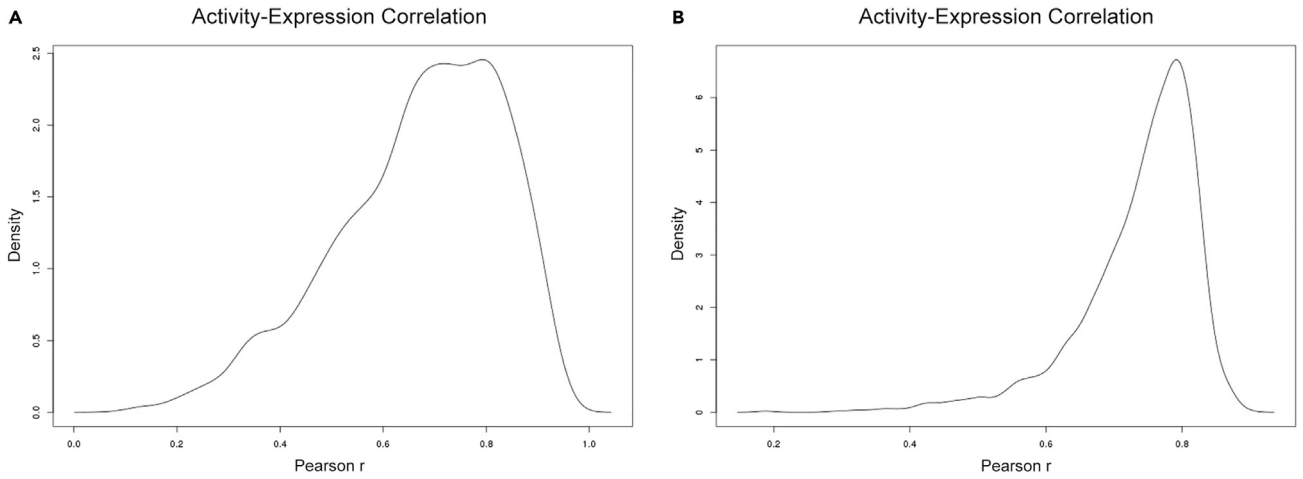


Figure 1. Density plots of paired regulator expression-activity correlations

(A and B) Density plots of Pearson correlations between paired expression and activity values across samples for each expression regulator for which VIPER could infer activity using the (A) GEUVADIS toy dataset, or (B) TwinsUK adipose dataset.

As transcript expression level is an imperfect proxy for gene product activity given various post-transcriptional and post-translation regulatory phenomena, we do expect the correlation between regulator expression and activity to be less than perfect. However, such correlations should still tend to be high for most regulators, as they are for this demonstrated analysis (Figure 1). See problem 3 in the troubleshooting section if this plot indicates most correlations are weak (i.e., Pearson r < 0.7).

Identifying representative putative phenotypic master regulators (MRs)

Timing: 30 min

Phenotypic cell states can be identified by, and are importantly determined by, their transcriptional state.[7–10] Gene regulatory networks are key to the progression from one cell state to another and to the maintenance of cellular homeostasis by canalizing genetic and environmental information in the establishment of stable transcriptional states.[7–10] The canalization of genetic information by gene regulatory networks is dramatically demonstrated in cancers and other genetically induced disease states wherein many distinct mutational profiles may converge on the same transcriptional and phenotypic state.[8,13] Furthermore, a growing body of regulatory network-based analyses have indicated that the canalization of genetic and environmental information occurs through master regulators (MRs) that are organized by positive feedback relationships into tightly connected regulatory modules that importantly determine the transcriptional state.[8,13] Based on this conceptual framework, we use a master regulator (MR) analysis to focus our *trans*-eQTL and aQTL analyses on a subset of regulators that best represent these MR modules, thereby greatly reducing the multiple testing burden while simultaneously enriching for regulators that are important for establishing the transcriptional state associated with the phenotype of interest. Specifically, we select representative phenotypic MRs based on ranked importance in a random forest classification or regression model predicting phenotype from inferred regulator activities. The MR analysis is split into two steps: (1) random forest cross-validation analysis to determine a suitable number of representative phenotypic MRs, and (2) final random forest model training for the selection of representative phenotypic MRs.

7. If still open, you may continue working in the same R session used above and move on to **Step 8**. Otherwise, start an R/RStudio session, load the MRaQTL package, and set your working directory to that containing the inferred activity data (as generated in **Step 6**), and phenotype data as in **Step 5**.

8. Use the rfCrossVal() function to perform random forest cross-validation analysis to estimate how many regulators are needed to effectively minimize the mean cross-validation error. For this demonstration with the GEUVADIS dataset, a simulated phenotype labeled "Sim_pheno" is used as the phenotype of interest.

```
> rfCrossVal("Toy_data_regulators_activities.txt",
"GEUVADIS_simulated_phenotype.txt","Sim_pheno","Toy_data")
```

⚠ CRITICAL: It is essential that the phenotype data come from the same cohort as the transcriptomic data. For the demonstration of this protocol, we use a phenotype data file that only includes the "Sim_pheno" simulated phenotype as a continuous variable. However, users may input a phenotype file with multiple different phenotypes (one per column) since one of the arguments passed to the rfCrossVal() function specifies for which phenotype you wish to identify representative phenotypic MRs. Indeed, the user may wish to run MR and QTL analyses for each relevant phenotype. The phenotype file should be formatted as a tab-delimited text file with samples by rows and phenotypes by columns with sample IDs in the first column and a header as the first row that includes the phenotype names. Both continuous and binary (e.g., case-control status) phenotypes are supported, though their outputs differ as a reflection of the use of random forest regression versus classification, respectively.

Note: The four arguments used in this command correspond to the regulator activities data file generated in **Step 6**, the phenotype file, a string indicating the phenotype column name, and a prefix string for labeling the output files, in that order. For full details on the function and its arguments, enter "?rfCrossVal" in the R console or search for the function in the RStudio Help tab. Note that there are six additional arguments that provide greater control over this analysis but for this example we use their defaults.

Note: One of the outputs of this step is a plot overlaying the distributions of the phenotype of interest for the training and test sets (Figure 2). This can indicate potential biases in the phenotype between the two sets. For this demonstration analysis, we see that for the GEUVADIS and TwinsUK datasets, the "Sim_pheno" and BMI distributions are well matched between the

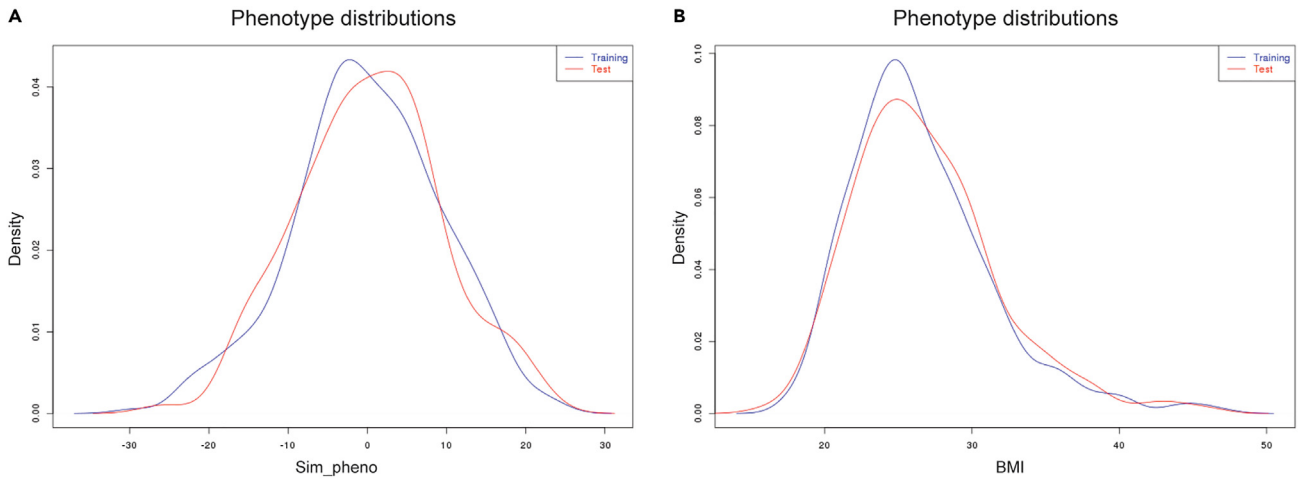


Figure 2. Phenotype distributions in the training and test sets

(A and B) Overlayed density plots for the (A) "Sim_pheno" values among GEUVADIS samples, or (B) BMI measurements among TwinsUK samples for the training (blue line) and test (red line) sets. The phenotype distributions are suitably similar between the training and test sets for both datasets.

training and test sets, respectively (Figure 2). If your phenotype distributions are uneven, see problem 4 in the troubleshooting section for possible remedies.

Note: This analysis begins by testing linear models associating each regulator's activity to the phenotype of interest as a first step to focus the analysis on the most relevant regulators and reduce the computational burden. By default, only the most significant regulators with a Bonferroni-adjusted $P \leq 0.05$ are used in the cross-validation random forest modeling up to a maximum of 500, but with a minimum of 100. For this demonstrated analysis, out of 1,574 significantly associated with "Sim_pheno", the top 500 regulators are used. If there are few regulators with activities significantly associated to your phenotype according to the linear models (e.g., less than ~100), which you can check in the function's standard output log file, see problem 5 in the troubleshooting section.

⚠ CRITICAL: In addition to the phenotype distribution plot previously mentioned, the rfCrossVal() function outputs a log file with the standard output generated by the function, the R workspace (to be used in the next step), and a plot of mean cross-validation error as a function of feature count. The cross-validation plot is the key result that indicates how many regulators are needed to effectively minimize the phenotype prediction error (Figure 3). A feature count at which the prediction error stabilizes indicates the number of representative phenotypic MRs we will select (typically ~100) in the next step when training and testing the final phenotypic MR random forest model. If the prediction error fails to stabilize, see problem 6 in the troubleshooting section.

9. Use the finalRF() function to train the final phenotypic MR random forest model and identify the representative putative phenotypic MRs. Based on the previous step, we will identify 100 representative "Sim_pheno" MRs.

```
>finalRF("Toy_data_rfcv_workspace.RData",
"Toy_data_regulators_QNorm_INT_expression.txt",
"Sim_pheno","Toy_data",100)
```

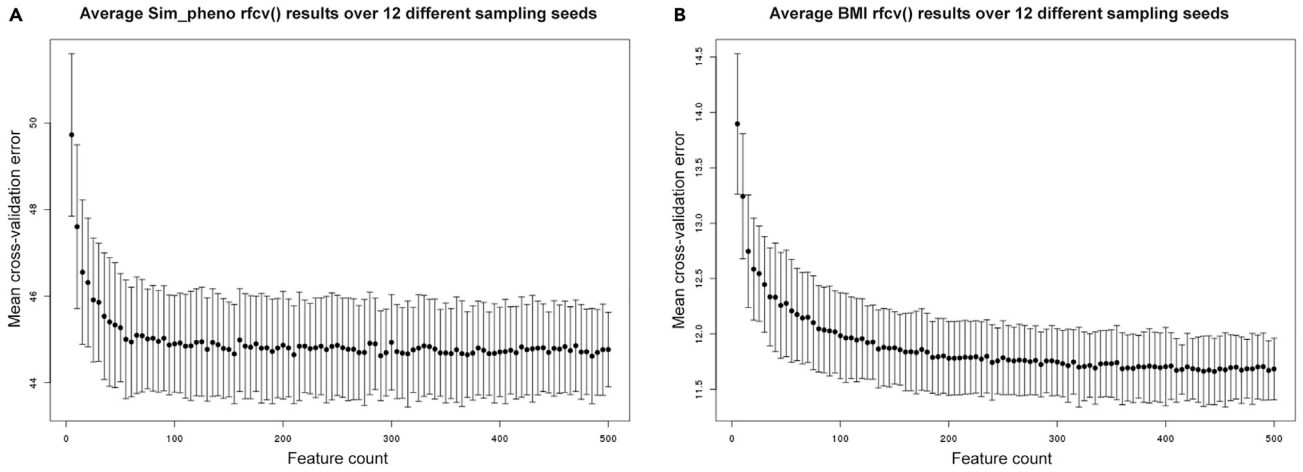


Figure 3. Cross-validation random forest plots

(A and B) Cross-validated prediction performance of random forest regression models with the number of predictors sequentially reduced by five starting with 500. Models were trained to predict (A) "Sim_pheno" for the GEUVADIS dataset, or (B) BMI for the TwinsUK dataset from regulator activities 12 independent times, each with a unique seed. The plot compares the number of predictors included in the model versus the mean cross-validation error, and error bars indicate the standard deviation of the 12 analyses. Users should choose a feature count shortly after where the curve begins to flatten. For this protocol we chose 100, which determines the number of representative "Sim_pheno" or BMI MRs selected in the next step.

⚠ CRITICAL: One of the outputs of this script is a text file with summary statistics for the comparison of predicted (based on the final MR random forest model) and actual phenotype values for both the training and test sets (linear model statistics generated by stats::lm() function for continuous phenotypes and confusion matrix statistics generated by the caret::confusionMatrix() function for binary phenotypes). For this protocol demonstration with the simulated phenotype, the associations should both be quite significant ($P_{training}$ = 2.26 × 10^{-50} and P_{test} = 7.03 × 10^{-17}). However, if the *P*-values are poor for either the training or test set with your own data, the MR analysis should not be trusted. See problem 7 in the troubleshooting section.

Note: The finalRF() function is run with five arguments corresponding to the R workspace file from the previous step, the file for the regulators normalized expression data generated in **Step 6**, a string indicating the phenotype column name, a prefix string for labeling the output files, and the number of representative phenotypic MRs desired (based on the random forest cross-validation plot), in that order. For full details on the function and its arguments, enter "?finalRF" in the R console or search for the function in the RStudio Help tab.

Note: While having good prediction within the training and test sets is a minimum requirement for trusting the putative representative phenotypic MRs, when possible, it is ideal to validate their generalized predictive value in an independent expression data set. Toward this end, the script will also output the R workspace that includes the final trained MR random forest model that may be used for such validation.

Performing activity QTL (aQTL) and eQTL analyses

Timing: 20 min

The ultimate aim of this protocol is the identification of master regulators (MRs) that may mediate phenotypic influences of germline genetic variants identified through genome-wide association studies (GWAS). As discussed above, MRs represent theoretical bottlenecks in gene regulatory networks through which genetic information may be canalized in the establishment of the phenotype-associated transcriptional state. Consequently, such genetic influences may be detectable in *trans* on the expression and/or activity of the representative phenotypic MRs. Indeed, we have previously shown not only that MR *trans*-eQTLs and aQTLs can colocalize with GWAS signals even in the absence of any colocalizing *cis*-eQTL, but that aQTLs are advantageous over eQTLs in such *trans* analyses.[1] Therefore, in this step we will perform *cis*-eQTL and aQTL analyses on all regulators and *trans*-eQTL and aQTL analyses on the representative phenotypic MRs identified in the previous step. Once again, we will provide the commands for analyzing the GEUVADIS dataset, but we must re-emphasize that in this context it serves as a toy dataset which is not expected to generate clean, biologically meaningful results. For this reason, we also display the results from the TwinsUK dataset comparing BMI GWAS-significant variants[4] to the inferred adipose BMI MRs as an example of a well-polished analysis that is meaningful.

10. If still open, you may continue working in the same R session used above and move on to **Step 11.** Otherwise, start an R/RStudio session, load the MRaQTL package, and set your working directory to that containing all the normalized expression and activity data generated in previous steps as well as the phenotype and covariables data as in **Step 5.**
11. Use the matrixQTL() function to run *cis*-eQTL and aQTL analyses on the regulators with the following commands, respectively.

```
> matrixQTL(
"GEUVADIS_filtered_samples_WHR_significant_bi-allelic_SNPs.dosage",
"GEUVADIS_WHR_significant_bi-allelic_SNPs_locations_in_GRCh37.map",
"Toy_data_regulators_QNorm_INT_expression.txt",
"GEUVADIS_expressed_genes_locations_in_GRCh37.map",
"GEUVADIS_filtered_samples_select_covars.txt",
"Toy_data_all_regs_eQTL")
> matrixQTL(
"GEUVADIS_filtered_samples_WHR_significant_bi-allelic_SNPs.dosage",
"GEUVADIS_WHR_significant_bi-allelic_SNPs_locations_in_GRCh37.map",
"Toy_data_regulators_activities.txt",
"GEUVADIS_expressed_genes_locations_in_GRCh37.map",
"GEUVADIS_filtered_samples_select_covars.txt",
"Toy_data_all_regs_aQTL")
```

Note: The matrixQTL() function is run here with six arguments corresponding to the imputed genotype dosage file, the SNP map file, the expression or activity data file, the gene map file, the covariables data file, and a prefix string for labeling the output files, in that order. A covariables file is recommended but optional, as NULL is the default value. For full details on the function and its arguments, enter "?matrixQTL" in the R console or search for the function in the RStudio Help tab.

Note: For this demonstration we have arbitrarily restricted the genotype data to significant waist-hip ratio (WHR) GWAS variants. For a real analysis, the variants analyzed should be chosen based on the GWAS results of a complex trait of interest for which the cell/tissue context being studied is likely relevant.

Note: The SNP map file should be formatted as a tab-delimited text file with 3 columns labeled in a header as "snp", "chr", and "pos", with the first column containing SNP IDs, the second containing chromosome IDs (e.g., chr1, chrX, etc.), and the third column containing chromosomal coordinates (based on the same reference genome as the gene map file). The gene map file should be formatted as a tab-delimited text file with 4 columns labeled in a header as "Gene", "chr", "s1", and "s2", with the first column containing gene IDs, the second containing chromosome IDs (e.g., chr1, chrX, etc.), and the third and fourth column containing chromosomal coordinates corresponding to the start and end sites of the gene (based on the same reference genome as the SNP map file). For this demonstration of the protocol, both map files use GRCh37 coordinates. See the GEUVADIS files for examples of the required formatting.

Note: For the GEUVADIS dataset we are only adjusting for sex and ancestry categories (Finnish, Tuscan, Utah, Yoruba). Population heterogeneity can strongly confound QTL analyses and adjusting for categories is a crude approach. For a real analysis it would preferable to increase the population homogeneity by sample filtering and then further adjust for residual population admixture effects by including the top 3–5 genotype-based principal components as covariables, which was the approach used for the TwinsUK adipose data analyses.[1] Alternative or additional covariables may be added for this step as the user deems appropriate, but we would strongly recommend excluding any covariables that are well correlated with the phenotype of interest as it may diminish any associations that are relevant to the mediation of the phenotype.

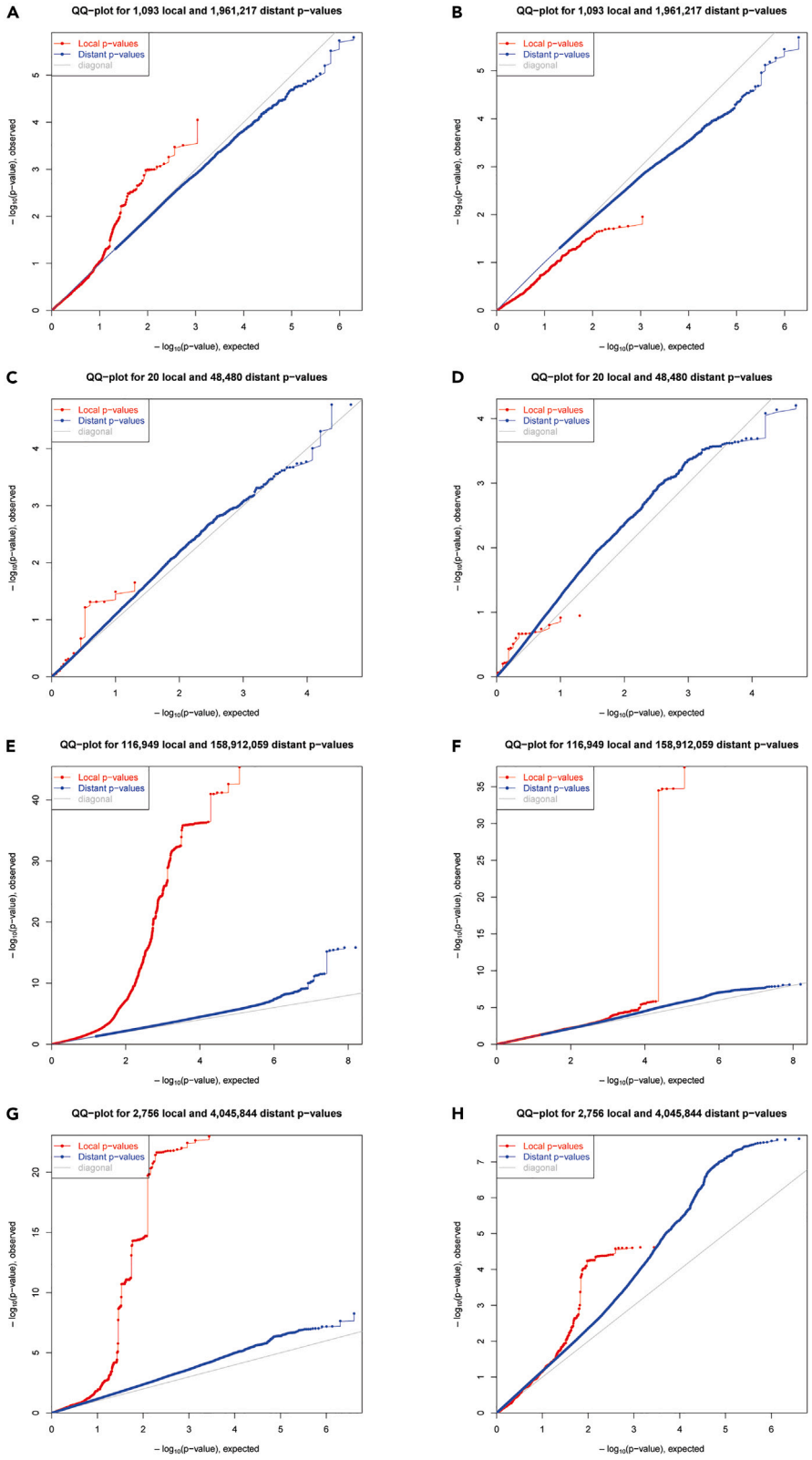

A
QQ−plot for 1,093 local and 1,961,217 distant p−values
B
QQ−plot for 1,093 local and 1,961,217 distant p−values
C
QQ−plot for 20 local and 48,480 distant p−values
D
QQ−plot for 20 local and 48,480 distant p−values
E
QQ−plot for 116,949 local and 158,912,059 distant p−values
F
QQ−plot for 116,949 local and 158,912,059 distant p−values
G
QQ−plot for 2,756 local and 4,045,844 distant p−values
H
QQ−plot for 2,756 local and 4,045,844 distant p−values
Local p−values
Distant p−values
diagonal
− log10(p−value), observed
− log10(p−value), expected

Figure 4. QQ-plots for the all-regulator and MR eQTL and aQTL analyses

(A–H) QQ-plots comparing the observed versus expected distributions of -$\log_{10}P$ for all tested *cis* (red) and *trans* (blue) QTLs from the following analyses: (A and E) eQTL analysis of all regulators and significant GWAS variants, (B and F) aQTL analysis of all regulators and significant GWAS variants, (C and G) eQTL analysis of representative phenotypic MRs and significant GWAS variants, (D and H) aQTL analysis of representative phenotypic MRs and significant GWAS variants. Panels (A–D) represent results with the GEUVADIS toy dataset, while panels (E–H) represent results from the TwinsUK adipose dataset.

Therefore, if PEER factors are being included as covariables to adjust for unknown batch effects, it would be wise to include the phenotype of interest as a covariable in the PEER factor training to ensure all inferred factors are uncorrelated with it.[14] Considerations of sample QC and covariable inclusion are key to the success of any QTL analyses.

Note: These eQTL and aQTL analyses with all regulators are focusing on *cis* rather than *trans* associations, which is why the *P*-value thresholds are being set to 1.0 and 0.05, respectively. This will still save the nominally significant *trans*-QTL results, but the *trans P*-value threshold may be adjusted according to the user's interest in saving the *trans*-QTL results for all regulators.

Note: The matrixQTL() function outputs QQ-plots from the Matrix eQTL package that indicates the observed versus expected distributions of -$\log_{10}$(*P*-values) for the *cis* and *trans* QTLs tested (Figure 4). If these plots show little to no -$\log_{10}P$ above expectation (i.e., above the diagonal), see problem 9 in the troubleshooting section. Note that the GEUVADIS dataset runs into this problem, which is probably due at least in part to insufficient adjustment for population admixture. This is not a problem for the TwinsUK adipose analysis.

Alternatives: The matrixQTL() function uses functions imported from the MatrixEQTL R package[15] for rapid QTL analyses, but users may use any eQTL analysis tool or package they prefer. The only difference between aQTL and traditional eQTL analyses is the use of inferred regulator activities in place of normalized expression values. If an alternative eQTL analysis tool or package is used in place of the matrixQTL() function, the user will need to format the input data accordingly.

12. Use the matrixQTL() function to run *trans*-eQTL and aQTL analyses restricted to the representative phenotypic MRs.

```
> matrixQTL(
"GEUVADIS_filtered_samples_WHR_significant_bi-allelic_SNPs.dosage",
"GEUVADIS_WHR_significant_bi-allelic_SNPs_locations_in_GRCh37.map",
"Toy_data_MRs_only_QNorm_INT_expression.txt",
"GEUVADIS_expressed_genes_locations_in_GRCh37.map",
"GEUVADIS_filtered_samples_select_covars.txt",
"Toy_data_MR_eQTL",cisP=1,transP=1)
> matrixQTL(
"GEUVADIS_filtered_samples_WHR_significant_bi-allelic_SNPs.dosage",
"GEUVADIS_WHR_significant_bi-allelic_SNPs_locations_in_GRCh37.map",
"Toy_data_MRs_only_activities.txt",
"GEUVADIS_expressed_genes_locations_in_GRCh37.map",
"GEUVADIS_filtered_samples_select_covars.txt",
"Toy_data_MR_aQTL",cisP=1,transP=1)
```

Note: These analyses are run as for the *cis* analyses above, but with the MR filtered expression or activity data and with the *trans P* threshold fully relaxed to 1. Here the *cis P* threshold is still set to 1 (the default), which will again save all the *cis*-QTL results, though they should be identical to the corresponding QTLs generate in the previous *cis*-QTL analyses with all regulators.

Note: The eQTL and aQTL summary statistics files include a column for FDR. However, the MatrixEQTL package at the heart of the matrixQTL() function does not account for the LD between SNPs nor for correlations between gene expression or activities, which in general leads to overly stringent multiple testing correction. As mentioned above, one of the primary advantages to restricting the *trans*-QTL analyses to representative phenotypic MRs is the dramatic reduction of the multiple testing burden, which provides greater power in detecting relevant associations. Therefore, when applying multiple testing corrections, use only the MR *trans*-QTL results rather than the all-regulators *trans*-QTL results that may have been saved from the *cis*-focused analyses above. For examples of multiple testing correction of QTL results while appropriately accounting for both LD of variants and gene correlations, see Hoskins et al. (2021) and Huang et al. (2018).[1,16]

EXPECTED OUTCOMES

The "/Toy_data_outputs/" directory of the https://github.com/hoskinsjw/aQTL_STAR_protocol/ GitHub repository includes example outputs from the analyses described here using the GEUVADIS dataset, with the exclusion of the RData workspace files. Large text files (>25 MB) have been truncated but still indicate the expected form of such output files.

Inferring ARACNe co-expression network

As described above, the steps for inference of the ARACNe co-expression network will generate N+3 files, where N is the number of bootstraps performed. In addition to the N individual bootstrap networks, these steps will output a mutual information threshold file and two final, consolidated co-expression network files that differ only in the inclusion or exclusion of a header.

Inferring regulator activities with VIPER

The prepActExp() function generates six files: 1) a log file containing the standard output generated by the function, 2) a tab-delimited file for the interactome, which differs from the original ARACNe network by the replacement of *P*-values with likelihood and replacement of mutual information with a "Mode of Action" (MoA) score that indicates the direction and magnitude of the expression regulatory relationship,[12] 3) a png image file for the activity-expression correlation density plot, 3) a file with the upper quantile normalized and inverse normal transformed expression data for all expressed genes, 4) a file with the upper quantile normalized and inverse normal transformed expression data for regulators only, and 5) a file with the expression regulator activities inferred by VIPER.

Identifying representative putative phenotypic master regulators

The master regulator (MR) analysis utilizes two functions sequentially: the rfCrossVal() function is used to determine a reasonable number of predictors for minimizing phenotype prediction error, and then the finalRF() function trains the final random forest model using the number (selected from the random forest cross-validation) of regulators chosen by importance for phenotype prediction. The first function generates four files: 1) a log file containing the standard output from the function, 2) a png image file for the phenotype distributions in the training and test sets, 3) a png image file for the random forest cross-validation plot, and 4) an RData workspace file that is used as input for the subsequent finalRF() function. The finalRF() function generates seven files: 1) a log file containing the standard output from the function, 2) a pdf file for the plot of type 1 importance (mean decrease in accuracy upon predictor permutation) for the representative putative phenotypic MRs in the final random forest model, 3) a text file containing the summary statistics from testing the associations between the final MR random forest model's predicted phenotypes and actual phenotypes in the training and test sets, 4) a tab-delimited text file listing the representative putative

phenotypic MRs, 5) A tab-delimited text file containing the normalized expression data filtered to include only the representative phenotypic MRs, 6) a tab-delimited text file containing the activity data filtered to include only the representative phenotypic MRs, and 7) an RData workspace file that includes the final MR random forest model.

Performing activity QTL (aQTL) and eQTL analyses

Four distinct QTL analyses are performed in this protocol using the matrixQTL() function, and each analysis produces results divided into *cis* and *trans* associations: eQTL and aQTL analyses of all regulators with inferred activities, and eQTL and aQTL analyses of only the representative putative phenotypic master regulators (MRs). Each of the four QTL analyses generates four files: 1) a log file containing the standard output from the function, 2) a pdf file for the QQ-plot of observed versus expected -$\log_{10}P$ for all tested *cis*- and *trans*-QTLs, 3) a tab-delimited text file containing the summary statistics for all *cis*-QTLs passing the user-defined *P*-value threshold, and 4) a tab-delimited text file containing the summary statistics for all *trans*-QTLs passing the user-defined *P*-value threshold.

LIMITATIONS

As with any complex analysis, this protocol is subject to some notable limitations. The first is that when analyzing a large dataset, running hundreds of bootstrap networks with ARACNe can take a long time without running them as parallel jobs on a computer cluster.

Since ARACNe reconstructs gene regulatory networks exclusively from transcriptomic data, the identification of a regulator's target genes requires sufficient variance in its expression. As such, expression regulators whose activities are determined primarily through post-translational regulation with little to no change in their own steady state transcript levels may not be effectively linked to their true downstream targets, which would thereby preclude inference of their activities via VIPER. Furthermore, while VIPER's activity inferences are remarkably robust to noise in the expression data and changes in the ARACNe network density,[12] they are still dependent on how adequately representative the supplied tissue/cell type-specific ARACNe network is, and therefore care should be taken to ensure the quality and contextual adequacy of the ARACNe network employed. Reliance on a single ARACNe network has been alleviated by an updated version of VIPER called metaVIPER that accepts multiple ARACNe networks that are adaptively applied regulon-by-regulon for the inference of each regulator.[17] Such flexibility could be advantageous when dealing with more heterogeneous samples, but we have not yet implemented that functionality in the MRaQTL package. This may be implemented in future versions.

We previously demonstrated that focusing *trans*-eQTL/aQTL analyses of significant GWAS variants on relevant phenotypic master regulators (MRs) enables the identification of gene regulatory pathways potentially mediating the phenotypic effects of GWAS signals even when colocalizing *cis*-eQTLs are unobserved.[1] However, the selection of relevant MRs is a non-trivial task given the correlations in gene expression and inferred activities among many expression regulators, some of which may be functionally key to establishing the phenotype-associated transcriptional state of the cell (i.e., are true phenotypic MRs) while others may be incidental and unnecessary for the phenotype of interest. Consequently, we utilized the non-linear method of random forest modeling to identify a set of representative putative phenotypic MRs based on their importance in predicting the phenotype of interest from their inferred activities, regardless of the correlations between them. However, despite such models being robust in their phenotype predictions, the regulator importance estimates can be sensitive to very small differences in inferred activity profiles (likely amplified by the bootstrapping and bagging used in training the decision trees) such that the ranking of regulators by importance may vary among similarly important regulators, yielding somewhat distinct final MR lists. This sensitivity should not reduce confidence in observed MR *trans*-eQTLs or aQTLs since the differences between possible final MR lists are among regulators of similar importance whose

activities are correlated with those of other, included MRs. We raise this point only to emphasize a nuance in interpreting the MR results: the putative phenotypic MR list should be regarded as a list of regulators *representing* the activities of various gene regulatory bottlenecks (i.e., highly correlated and cooperative MR modules) that are important in establishing the phenotype-associated transcriptional state, rather than an exhaustive list of the direct mediators of the genetic component of the phenotype. As such, any observed MR *trans*-eQTL or aQTL may be viewed as representative of a potential genetic influence on a hypothetical MR module in which the given representative putative phenotypic MR participates. Alternative approaches for more comprehensive identification of phenotypic MRs in these relevant gene regulatory modules are currently being explored.

TROUBLESHOOTING

Problem 1

Occasionally an ARACNe bootstrap will fail to complete in **Step 3**, yielding fewer bootstrap networks than intended for the final consolidated network.

Potential solution

When this occurs, it is consistent for a given bootstrap seed number. Therefore, if one is intent on reaching a particular number of bootstrap networks for final network consolidation, run another bootstrap network with an unused seed number.

Problem 2

The VIPER algorithm used by prepActExp() fails to infer activities for very many regulators (e.g., significantly less than 50% of the full expression regulator list) in **Step 6**.

Potential solution

By default, VIPER only infers activities for regulators with at least 25 expressed target genes based on the given ARACNe network. Therefore, this problem is likely caused by an overly sparse ARACNe network, which may result from using a small expression dataset (~100 or less samples) or too few bootstraps in the ARACNe network inference. If the expression dataset is small, one could try using a pre-made ARACNe network inferred from a matching sample type (inferred from more samples) or inferring a new ARACNe network from a larger expression dataset for the matching sample type. Alternatively, one could try inferring the ARACNe network from their dataset with more bootstraps, though increasing the number of bootstraps will probably have limited efficacy much above the total sample size of the expression dataset.

Problem 3

Pearson correlations between matched activities and expression values are poor (e.g., most regulators have $r < \sim 0.7$), as indicated in the correlation density plot generated by the prepActExp() function in **Step 6**.

Potential solution

This could indicate the ARACNe network used for VIPER inference poorly represents the gene regulatory architecture of the given dataset. Assuming the ARACNe network was inferred from the same expression dataset, then this would likely indicate significant heterogeneity among the samples, perhaps due to batch effects or highly variable cellularity. It may be possible to improve homogeneity of samples by adjusting for known batch effects in the expression data prior to ARACNe network inference and VIPER activity inference, but we would strongly warn against applying correction for unknown batch effects. Alternatively, if the dataset is sufficiently sized, it may be possible to split the samples into more homogeneous groups for separate analyses. If none of these options are viable, the dataset may not be suitable for this analysis.

Problem 4

Phenotype distributions in the training and test sets for the master regulator analysis are not well-matched in **Step 8**.

Potential solution

If the distributions are poorly matched because of one or two outliers, you may consider dropping such outliers from the master regulator analysis. Otherwise, you may provide a new random number generator seed to the "seed" argument of the rfCrossVal() function. This will lead to a new sampling for the training and test sets.

Problem 5

There are not very many (e.g., less than ~100) regulators whose activities are significantly associated with the phenotype of interest as indicated in the cross-validation random forest standard output log file generated by the rfCrossVal() function in **Step 8**.

Potential solution

This could indicate 1) that your data set has insufficient power, or 2) that the tissue type under investigation does not have any transcriptional state that is well correlated with the phenotype of interest. In either of these cases, you would likely need a new dataset with more samples or representing a more relevant tissue or cell type. If these explanations seem unlikely, it is possible the ARACNe network used by VIPER for activity inferences was poorly matched to the overall expression dataset (see problem 3 above).

Problem 6

The mean cross-validation error does not stabilize (i.e., remains notably sloped) with increasing feature counts according to the cross-validation random forest plot generated by the rfCrossVal() function in **Step 8**.

Potential solution

This could suggest a complex or inconsistent phenotype-associated pattern of regulator activities in the given sample set, perhaps due to problematic heterogeneity among the samples or because the phenotype of interest does not associate with a transcriptional state that is strongly canalized by the gene regulatory network in the tissue or cell type being investigated. While this may suggest the dataset is unsuitable for this analysis for the given phenotype, the reader could still continue to the next step (final random forest MR analysis) to assess how well the final random forest model predicts the phenotype. We recommend 100 as the default number of putative MRs for this analysis, though the reader may experiment with different numbers. If the final MR random forest model predictions significantly correlate with the actual phenotype values in both the training and test sets (as indicated in the actual vs. predicted output file), it may still be worth continuing with the MR *trans*-QTL analyses.

Problem 7

The summary statistics in the predicted vs. actual comparison output file generated in **Step 9** indicate poor phenotype prediction by the final MR random forest model for either the training or test dataset.

Potential solution

This problem is typically foreseeable from, and explainable by, the issues described in problems 4, 5, and 6 discussed above. If none those problems are observed, indicating the cross-validation random forest analysis was successful, then the phenotype prediction for at least the training set should be fairly accurate. If not, double-check that all inputs for **Step 9** are correct. If the predictions look good for the training set but not the test set, and the phenotype distributions look comparable between the training and test datasets as seen from the output of **Step 8**, then there may be more subtle

differences between the sample sets, perhaps due to unknown batch effects. In that case, you may need to perform some clustering analyses on the expression or activity data for all samples to check for such problematic sample difference that may be skewing the training or test set.

Problem 8

An eQTL/aQTL analysis fails to complete and generate its output files in **Step 11** or **Step 12**.

Potential solution

The MatrixEQTL R package used by the matrixQTL() function to conduct the eQTL/aQTL analyses can freeze when trying to write too many results to file, which typically happens when the number of *trans*-QTL tests is large and the *P*-value threshold for outputting the *trans*-QTL results was set too high. The simplest fix would be to lower the *trans*-QTL *P*-value threshold, which is set by the "transP" argument given to the matrixQTL() function. Alternatively, the user could split up the QTL analyses by chromosomes or sections of chromosome. If neither of those solutions are sufficient, the user may inspect the matrixQTL() function's standard output log file for the analysis and consult the Matrix eQTL website for more troubleshooting help. Finally, the user may use a different eQTL analysis tool or package, though this will likely require some reformatting of the input data.

Problem 9

The QQ-plot for an eQTL or aQTL analysis shows little to no $-\log_{10}P$ above expectation (i.e., above the diagonal) in **Step 11** or **Step 12**.

Potential solution

This problem could have a range of possible causes, which might include:

- Variant and gene map files were derived from different genome builds.
- The samples are in inconsistent orders between the expression/activity data, genotype data and covariates data.
- The dataset has insufficient sample size (i.e., a lack of statistical power).
- There are too few distinct loci being represented by the variants being analyzed, perhaps due to being selected from a relatively small GWAS.
- Some included covariable(s) might be adjusting for variance in expression/activity relevant to the genetic component of the relevant transcriptional state.
- The sample type being analyzed may not be very involved in mediating the genetic effects on the complex trait of interest.
- There are remaining confounding effects that have not been sufficiently adjusted for, such as population admixture.

If the problem is observed for all QTL analyses, then one or more of these general causes are likely since at the very least some true positive *cis*-eQTLs should be observable when examining many distinct loci. However, if it is only observed in the *trans*-eQTL/aQTL analyses, this could indicate the representative putative phenotypic MRs are not relevant to mediating the genetic component of the given complex trait in the tissue or cell type being studied. Finally, if it is only observed in the *cis*-aQTL analyses this may not indicate a problem at all. As previously reported, *cis*-aQTLs tend to be weaker than *cis*-eQTLs, though the opposite is true in *trans* when focusing on representative phenotypic MRs.[1]

RESOURCE AVAILABILITY

Lead contact

Further information and requests for relevant resources should be directed to, and will be fulfilled by, the lead contact, Jason W. Hoskins (jason.hoskins@nih.gov).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The MRaQTL R package introduced by this protocol is freely available from the https://github.com/hoskinsjw/MRaQTL GitHub repository (https://doi.org/10.5281/zenodo.7930026). All files used to demonstrate this protocol with the GEUVADIS toy data are freely available from the https://github.com/hoskinsjw/aQTL_STAR_protocol/ GitHub repository (https://doi.org/10.5281/zenodo.7929966). The TwinsUK adipose data files used to generate some example figures in this protocol are controlled access but may be provided pending approval of a Data Access Application with the TwinsUK Study: https://twinsuk.ac.uk/resources-for-researchers/access-our-data/.

ACKNOWLEDGMENTS

This project has been funded in whole or in part with federal funds from the National Cancer Institute, National Institutes of Health, under contract HHSN261200800001E. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government. This work also utilized the computational resources of the NIH HPC Biowulf cluster (http://hpc.nih.gov).

AUTHOR CONTRIBUTIONS

Conceptualization, J.W.H.; formal analysis, J.W.H.; software, J.W.H.; methodology, J.W.H.; validation, T.C.; funding acquisition, L.T.A.; supervision, L.T.A.; writing – original draft, J.W.H.; writing – review & editing, all authors.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

1. Hoskins, J.W., Chung, C.C., O'Brien, A., Zhong, J., Connelly, K., Collins, I., Shi, J., and Amundadottir, L.T. (2021). Inferred expression regulator activities suggest genes mediating cardiometabolic genetic signals. PLoS Comput. Biol. *17*, e1009563. https://doi.org/10.1371/journal.pcbi.1009563.

2. Lappalainen, T., Sammeth, M., Friedländer, M.R., 't Hoen, P.A.C., Monlong, J., Rivas, M.A., Gonzàlez-Porta, M., Kurbatova, N., Griebel, T., Ferreira, P.G., et al. (2013). Transcriptome and genome sequencing uncovers functional variation in humans. Nature *501*, 506–511. https://doi.org/10.1038/nature12531.

3. Buil, A., Brown, A.A., Lappalainen, T., Viñuela, A., Davies, M.N., Zheng, H.F., Richards, J.B., Glass, D., Small, K.S., Durbin, R., et al. (2015). Gene-gene and gene-environment interactions detected by transcriptome sequence analysis in twins. Nat. Genet. *47*, 88–91. https://doi.org/10.1038/ng.3162.

4. Yengo, L., Sidorenko, J., Kemper, K.E., Zheng, Z., Wood, A.R., Weedon, M.N., Frayling, T.M., Hirschhorn, J., Yang, J., and Visscher, P.M.; GIANT Consortium (2018). Meta-analysis of genome-wide association studies for height and body mass index in approximately 700000 individuals of European ancestry. Hum. Mol. Genet. *27*, 3641–3649. https://doi.org/10.1093/hmg/ddy271.

5. Margolin, A.A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., and Califano, A. (2006). ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. BMC Bioinf. *7*, S7. https://doi.org/10.1186/1471-2105-7-S1-S7.

6. Lachmann, A., Giorgi, F.M., Lopez, G., and Califano, A. (2016). ARACNe-AP: gene network reverse engineering through adaptive partitioning inference of mutual information. Bioinformatics *32*, 2233–2235. https://doi.org/10.1093/bioinformatics/btw216.

7. Arendt, D., Musser, J.M., Baker, C.V.H., Bergman, A., Cepko, C., Erwin, D.H., Pavlicev, M., Schlosser, G., Widder, S., Laubichler, M.D., and Wagner, G.P. (2016). The origin and evolution of cell types. Nat. Rev. Genet. *17*, 744–757. https://doi.org/10.1038/nrg.2016.127.

8. Califano, A., and Alvarez, M.J. (2017). The recurrent architecture of tumour initiation, progression and drug sensitivity. Nat. Rev. Cancer *17*, 116–130. https://doi.org/10.1038/nrc.2016.124.

9. Sonawane, A.R., Platig, J., Fagny, M., Chen, C.Y., Paulson, J.N., Lopes-Ramos, C.M., DeMeo, D.L., Quackenbush, J., Glass, K., and Kuijjer, M.L. (2017). Understanding tissue-specific gene regulation. Cell Rep. *21*, 1077–1088. https://doi.org/10.1016/j.celrep.2017.10.001.

10. Stadhouders, R., Filion, G.J., and Graf, T. (2019). Transcription factors and 3D genome conformation in cell-fate decisions. Nature *569*, 345–354. https://doi.org/10.1038/s41586-019-1182-7.

11. Smid, M., Coebergh van den Braak, R.R.J., van de Werken, H.J.G., van Riet, J., van Galen, A., de Weerd, V., van der Vlugt-Daane, M., Bril, S.I., Lalmahomed, Z.S., Kloosterman, W.P., et al. (2018). Gene length corrected trimmed mean of M-values (GeTMM) processing of RNA-seq data performs similarly in intersample analyses while improving intrasample comparisons. BMC Bioinf. *19*, 236. https://doi.org/10.1186/s12859-018-2246-7.

12. Alvarez, M.J., Shen, Y., Giorgi, F.M., Lachmann, A., Ding, B.B., Ye, B.H., and Califano, A. (2016). Functional characterization of somatic mutations in cancer using network-based inference of protein activity. Nat. Genet. *48*, 838–847. https://doi.org/10.1038/ng.3593.

13. Hahn, W.C., Bader, J.S., Braun, T.P., Califano, A., Clemons, P.A., Druker, B.J., Ewald, A.J., Fu, H., Jagu, S., Kemp, C.J., et al. (2021). An expanded universe of cancer targets. Cell *184*, 1142–1155. https://doi.org/10.1016/j.cell.2021.02.020.

14. Stegle, O., Parts, L., Piipari, M., Winn, J., and Durbin, R. (2012). Using probabilistic

estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. Nat. Protoc. *7*, 500–507. https://doi.org/10.1038/nprot.2011.457.

15. Shabalin, A.A. (2012). Matrix eQTL: ultra fast eQTL analysis via large matrix operations. Bioinformatics *28*, 1353–1358. https://doi.org/10.1093/bioinformatics/bts163.

16. Huang, Q.Q., Ritchie, S.C., Brozynska, M., and Inouye, M. (2018). Power, false discovery rate and Winner's Curse in eQTL studies. Nucleic Acids Res. *46*, e133. https://doi.org/10.1093/nar/gky780.

17. Ding, H., Douglass, E.F., Jr., Sonabend, A.M., Mela, A., Bose, S., Gonzalez, C., Canoll, P.D., Sims, P.A., Alvarez, M.J., and Califano, A. (2018). Quantitative assessment of protein activity in orphan tissues and single cells using the metaVIPER algorithm. Nat. Commun. *9*, 1471. https://doi.org/10.1038/s41467-018-03843-3.