STAR Protocols

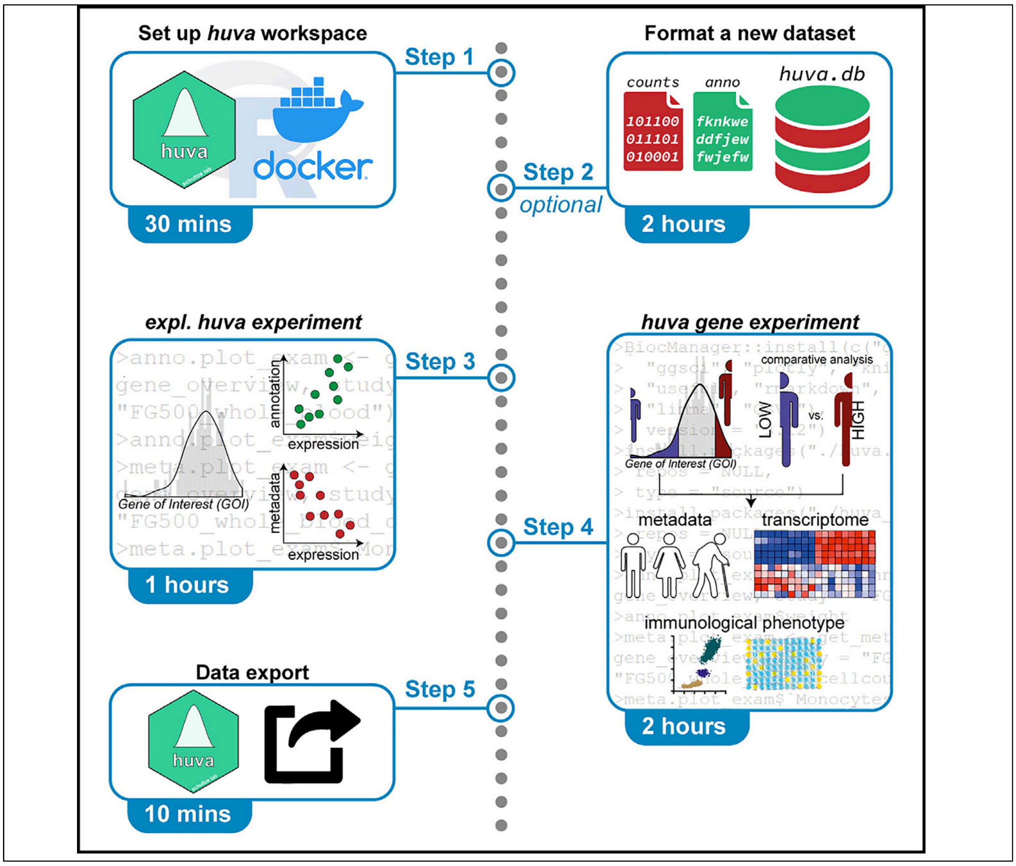
CellPress
OPEN ACCESS

Protocol

huva: A human variation analysis framework to predict gene perturbation from population-scale multi-omics data

Anna C. Aschenbrenner, Lorenzo Bonaguro

lorenzobonaguro@uni-bonn.de



Variance of gene expression is intrinsic to any given natural population. Here, we present a protocol to analyze this variance using a conditional quasi loss- and gain-of-function approach. The *huva* (*human variation*) package takes advantage of population-scale multi-omics data to infer gene function and the relationship between phenotype and gene expression. We describe the steps for setting up the *huva* workspace, formatting datasets, performing *huva* experiments, and exporting data.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Highlights

huva uses variation in gene expression to predict gene perturbation phenotype

huva leverages multi-layered dataset to integrate transcriptome and phenotype

huva provides an R-based framework for analysis of large human cohort studies

huva generates publication-ready graphics and summary statistics

Aschenbrenner & Bonaguro, STAR Protocols *4*, 102193
June 16, 2023 © 2023 The Author(s).
https://doi.org/10.1016/j.xpro.2023.102193

STAR Protocols

CellPress OPEN ACCESS

Protocol

huva: A human variation analysis framework to predict gene perturbation from population-scale multi-omics data

Anna C. Aschenbrenner[1] and Lorenzo Bonaguro[1,2,3,4,*]

[1]Systems Medicine, Deutsches Zentrum für Neurodegenerative Erkrankungen (DZNE), 53127 Bonn, Germany

[2]Genomics and Immunoregulation, Life & Medical Sciences (LIMES) Institute, University of Bonn, 53113 Bonn, Germany

[3]Technical contact

[4]Lead contact

*Correspondence: lorenzobonaguro@uni-bonn.de

https://doi.org/10.1016/j.xpro.2023.102193

SUMMARY

Variance of gene expression is intrinsic to any given natural population. Here, we present a protocol to analyze this variance using a conditional quasi loss- and gain-of-function approach. The *huva* (*human variation*) package takes advantage of population-scale multi-omics data to infer gene function and the relationship between phenotype and gene expression. We describe the steps for setting up the *huva* workspace, formatting datasets, performing *huva* experiments, and exporting data.

For complete details on the use and execution of this protocol, please refer to Bonaguro et al. (2022).[1]

BEFORE YOU BEGIN

Variation is an intrinsic characteristic of any biological parameter that is assessed at the population level. In humans, this is an essential variable when studying diseases in larger cohorts, and is increasingly recognized when describing healthy populations. Inclusion of omics data, such as transcriptomics, in large cohort studies, allows the use of the variance in gene expression to set up quasi-loss- and gain-of-function *in silico* experiments.

We have previously shown how variation in gene expression can be linked to gene function when both variables are measured in the same cohort.[2]

We developed the *huva* R package to enable the use of variance in gene expression to predict the functional role of a gene of interest (GOI). *huva* takes the extremes of the expression variance to model a quasi-loss- and gain-of-function experiment. We provide *huva* with a standalone database, *huva.db*, including several cohort-wide transcriptomic studies on healthy participants designed to study the inter individual variability of the immune system (500FG,[3,4] ImmVar,[5] and CEDAR[6]). In addition to the provided datasets, *huva* accepts user-provided datasets.

Huva is an easy-to-use R package that is also available via a graphical user interface on FastGenomics (https://beta.fastgenomics.org/a/huva) enabling scientists with or without bioinformatics skills to inspect the predicted function of a GOI for further validation experiments in culture systems (e.g., CRISPR-Cas9 mutants, RNA interference or gene overexpression) or animal models (e.g., knock-out/knock-in mouse).

Prerequisites/system requirements

1. Windows, Linux or MAC OSX system able to run Docker environments.
2. 64-bit processor.
3. 4GB system RAM (8GB recommended).

Running *huva* on a Docker container (recommended solution) does not require any dependency to be pre-installed on the system. If installing *huva* locally, R and several other packages need to be installed before *huva* (see troubleshooting 1 for details).

Setting up a *huva* workspace

⏱ Timing: 30 min

huva is an R package running on all operating systems compatible with base R (e.g., Windows, Linux, MAC OSX). To simplify the set-up of a workspace for *huva* analysis and to ensure the reproducibility of the analysis, we provide *huva* as a Docker image. We describe here the process to set up a Docker container running the most up-to-date version of *huva* while also exemplifying in the troubleshooting section how to install *huva* and *huva.db* on a local workstation as an alternative. As discussed in the original manuscript,[1] the *huva* framework runs on standard hardware (e.g., 1+ cores and 8 GB of RAM memory) within a few seconds for each GOI, analyzing human variation from over 2,400 transcriptomic profiles (included in the default *huva.db*).

4. Install Docker or another tool to deploy Docker containers (e.g., Singularity).

   Note: depending on the operating system follow the instructions at https://www.docker.com/, on Windows also the Windows Subsystem for Linux needs to be installed:

   a. Set up the Windows Subsystem for Linux 2 (WSL2) and any of the available Linux distributions:
      i. Install WSL2 first following the instructions reported here: https://learn.microsoft.com/en-us/windows/wsl/install.
      ii. Install Ubuntu or another Linux distribution as reported here: https://ubuntu.com/tutorials/install-ubuntu-on-wsl2-on-windows-10#1-overview.
   b. Download and install the latest version of Docker Desktop.
   c. Verify the installation of Docker Desktop.
      i. Open the Ubuntu terminal.
      ii. Run:

```
>docker info
```

   Note: If Docker is installed and running correctly this will output some basic information on the system, if no output is produced the installation was not successful. Since the cause of this malfunction could be extremely diverse, it is advisable to consult the troubleshooting section of Docker desktop (https://docs.docker.com/desktop/troubleshoot/overview/).

5. Start a Docker container for *huva* analysis.
   a. First, download the latest version of the *huva* docker image by typing in the terminal:

```
>docker pull lorenzobonaguro/huva_docker:015
```

b. Start a container with the downloaded image.

```
>docker run -dp [YOUR PORT]:8787 \ # define the port to use
>-e USER=[USER] -e PASSWORD=[PW] \ # username and password
>–name huva_analysis \ # name of the container
>-v [LOCAL DIRECTORY PATH]:/data/ \ # directory to mount
>lorenzobonaguro/huva_docker:015 #name of the docker image
```

c. Open the session on your browser by going to the address:

```
>http://localhost:[YOUR PORT]/
```

d. Log in with your username and password; in this environment, everything is ready to run a *huva* experiment

Note: The tag of the latest available *huva* docker image can be checked at https://hub.docker.com/. We discourage the use of the "latest" tag to ensure reproducibility of your script.

Note: This part is not intended as an exhaustive tutorial on how to use Docker, we encourage the reader to explore Docker functionalities on https://www.docker.com.

⚠ CRITICAL: If Docker cannot be installed on your system or you want to install the *huva* and *huva.db* packages with all the dependencies locally, we provide the respective script in the troubleshooting section.

Format a new dataset (optional)

⏱ Timing: 30 min, according to the dataset size

huva and *huva.db* are designed for the addition of new datasets to the original database. This is an optional step, which provides the user the opportunity to extend the analysis of human variation in other settings, i.e., tissues and organs. To exemplify this function, we showcase the implementation of the data from the GTEx consortium (v8).[7]

To generate a new *huva.db*, a normalized count table from transcriptomic data (e.g., RNAseq) is required with each sample as a separate column and HGNC gene symbols as row names, paired with a metadata table with samples as rows.

6. Data filtering and normalization: For RNAseq data, we recommend to filter low expressed genes as suggested by common RNAseq analysis pipelines (DEseq2, edgeR). For data normalization, we suggest using the *rlog* transformation from the DEseq2 package.[8]

   Note: This function transforms the count data to a log2 scale minimizing the differences for genes with low counts and normalizes the counts for the library size. Alternatively, logCPM with TMM normalization or the voom function (both part of the Limma package) are suitable transformation methods.

7. Data formatting: To be compatible with the *huva_dataset* format, format the new dataset as a separate list for counts, annotation and eventual additional metadata. Each list should have a matching substructure, here for example the structure of the GTEx dataset for whole blood and brain:

```
gtex_count
   |_whole_blood # Matrix with whole blood normalized counts
   |_brain # Matrix with brain normalized count
gtex_anno
   |_whole_blood # Metadata data frame for whole blood
   |_brain # Metadata data frame for brain
```

Note: The count table, annotation and metadata need to have a specific structure to be compatible with the *huva* framework. The count table should be a `matrix` with the sample names as column names and the gene names as row names. The metadata and annotation should be a `data.fame` with the parameters as column names and the sample names as row names. Those datasets can derive from different file format (e.g., csv, tsv, xlsx), R provides basic functions to read those format and convert them to `matrix` or `data.frame`.

8. Generate new *huva_dataset*: use the *generate_huva_dataset* function to combine the individual elements in an *huva_dataset* object that can be used for *huva* analysis:

```
>library(huva) # load huva package
>library(huva.db) # load huva.db package
>
>gtex.db <- generate_huva_dataset(dataset_name = "GTEx_v8",
>               data = "gtex_data",
>               annotation = "gtex_anno",
>               metadata = NULL)
```

CRUCIAL: The row names of the annotation table need to match the column names of the normalized count table otherwise the script will not be able to perform the analysis correctly and will lead to unexpected errors.

Note: By default, *huva* expects the genes to be annotated as HGNC gene symbols. If you prefer to keep the genes annotated with ENSEMBL IDs or Entrez Gene IDs, see the troubleshooting section.

Note: If you start with a new dataset, we encourage some preliminary exploratory data analysis (EDA) to check the data quality and the eventual presence of unwanted batch effects. If technical batches are present, this can be corrected with a batch correction tool such as the Limma batch correction (`removebatchEffect` function) or `ComBat` (sva package[9]). For a comprehensive review of the methods for EDA, we recommend the following resource.[10]

Note: When including a new dataset to *huva.db*, we suggest using datasets including transcriptomes from a minimum of 100 donors and at least one additional phenotypic or functional data layer.

Note: The *huva.db* format accepts two types of sample annotation. The `annotation` is a required part of the *huva.db*, this variable is meant to store basic information on the experimental cohort (e.g., sex, age) and accepts both numeric and non-numeric entries (does not accept missing values). The metadata variable, on the other hand, accepts only numeric values and is meant to store additional experimental layers (e.g., cell counts or cytokines secretion levels), here missing values are allowed.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Huva.db v. 0.1.5	Bonaguro et al.[1]	https://github.com/lorenzobonaguro/huva.db; https://doi.org/10.5281/zenodo.7071267
GTEx v8	GTEx Consortium[11]	https://www.gtexportal.org/home/datasets
Software and algorithms		
Huva v 0.1.5	Bonaguro et al.[1]	https://github.com/lorenzobonaguro/huva; https://doi.org/10.5281/zenodo.7071267
DEseq2 v. 1.30.1	Love et al.[8]	https://bioconductor.org/packages/release/bioc/html/DESeq2.html
Limma v 3.46.0	Ritchie et al.[12]	https://bioconductor.org/packages/release/bioc/html/limma.html
Fgsea v. 1.16.0	Korotkevich et al.[13]	https://bioconductor.org/packages/release/bioc/html/fgsea.html
ggplot2 v.3.3.3	R Tidyverse	https://ggplot2.tidyverse.org/
Docker Desktop v 20.10.16	https://www.docker.com/products/docker-desktop	RRID: SCR_016445
R v. 4.0.3	http://www.r-project.org/	RRID: SCR_001905
Other		
Code for reproducibility of the analysis	This paper	https://github.com/lorenzobonaguro/STAR_protocol_human_variation
Huva web portal	Bonaguro et al.[1]	https://beta.fastgenomics.org/a/huva

STEP-BY-STEP METHOD DETAILS

We will exemplify here the main steps required to run a *huva in silico* experiment (graphically summarized in Figure 1). We will show how to run the experiment and how to explore the results generating both, summary statistics (e.g., differentially expressed genes, fold-change and p-value of metadata and cell annotation, GSEA results) and graphical visualizations (e.g., GOI expression histogram, boxplot of the expression of selected genes, metadata and annotation boxplot and correlation plot with the GOI, heatmap of differentially expressed genes, dotplot of GSEA results).

All code shown here is also available in the GitHub repository of this manuscript (https://github.com/lorenzobonaguro/STAR_protocol_human_variation).

Furthermore, the *huva* R package includes a vignette which can be visualized in R after loading the package (vignette("huva_workflow")). Each *huva* function also includes detailed documentation according to the standard R structure (e.g., help(run_huva_experiment)).

Exploratory huva experiment

Timing: 30 min

As a first exploratory step of the *huva* framework perform an *exploratory huva experiment*.

Note: In this step, the expression of the selected GOI is compared across selected datasets and correlated with available metadata to evaluate the relationship between gene expression and the available parameters. This approach provides a quick overview of GOI expression across datasets and cell types, helping to shape downstream analyses.

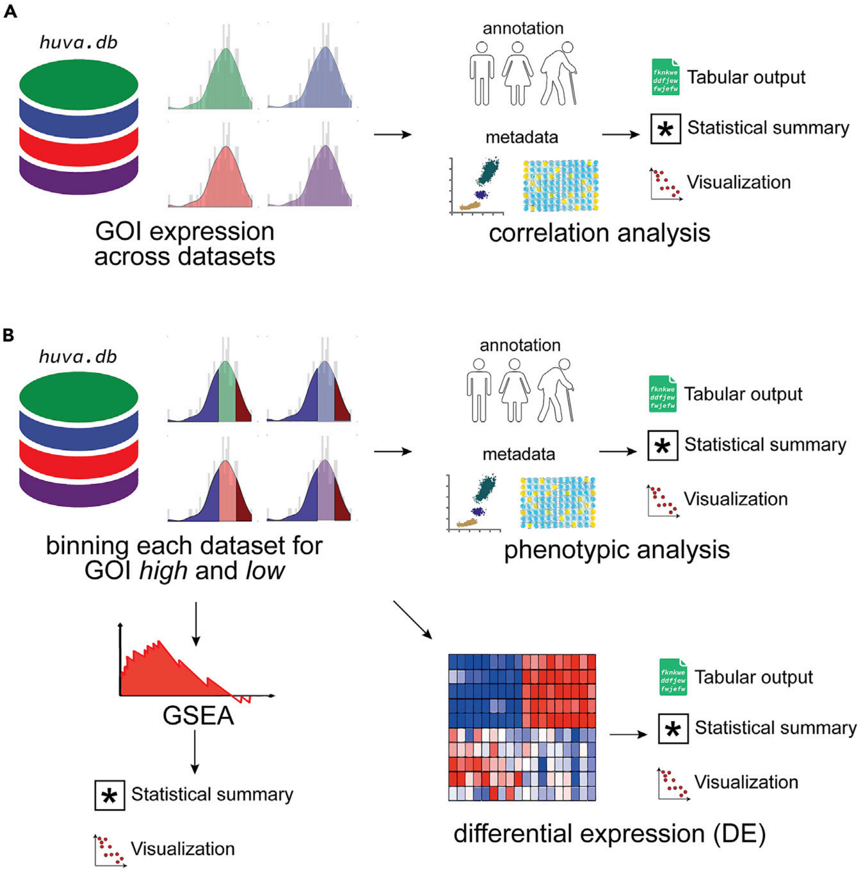


Figure 1. Overview of the *huva* framework workflow
(A) Graphical visualization of the *exploratory huva experiment* workflow.
(B) Graphical visualization of the *huva gene experiment* workflow.

1. First, run the experiment; the results are stored in the R environment:
   a. Load the packages.

```
>library(huva)

>library(huva.db)
```

   b. Define your gene of interest (GOI) as a variable, exemplified here with *MYD88*.

```
>GOI <- ‘‘MYD88’’
```

   c. Run the *exploratory huva experiment* with a simple function performing the analysis across all datasets provided in the database.

```
>gene_overview <- gene_exam(huva_dataset = huva.db,

gene = gene_name)
```

Note: If a gene is not expressed in a dataset, a message will be displayed (e.g., "MYD88 is not present in PLA").

2. Explore the results of the *exploratory huva experiment* using several built-in functions not only to extract the data but also to plot and visualize the results.

a. At first, examine the expression of the GOI across the provided datasets.
   i. Export the expression as data frame.

```
>expr_exam <- get_expr_exam(huva_expression = gene_overview, > study = "ImmVar",
>              dataset = "ImmVar_CD4T")
```

   ii. Or plot the expression for each dataset (Figure 2A).

```
>expr_exam.plot <- get_expr.plot_exam(huva_expression = gene_overview, bins = 50, alpha = 1)
>expr_exam.plot
```

Note: The single plots are stored within the results of the *exploratory huva experiment*, which can be accessed directly by exploring the object (e.g., `gene_overview$plot$FG500_whole_blood`).

b. In the next step, investigate the correlation between the GOI and the provided annotations and metadata with the outputs produced with the following commands.
   i. A data frame with the annotation (`get_anno_exam`) or metadata (`get_metadata_exam`) of each dataset.

```
>anno_exam <- get_anno_exam(huva_expression = gene_overview, > study = "CEDAR",
>              dataset = "CEDAR_CD4T")
>meta.table_exam <- get_meta_exam(huva_expression = gene_overview, study = "FG500")
```

   ii. A table with statistics (t-test for discrete variables and Pearson's correlation for continuous variables) for annotation (`get_anno.stat_exam`) and metadata (`get_meta.stat_exam`).

```
>anno.stat_exam <- get_anno.stat_exam(huva_expression = gene_overview, study = "FG500",
dataset = "ALL")
>meta.stat_exam <- get_meta.stat_exam(huva_expression = gene_overview, study = "FG500",
dataset = "FG500_whole_blood_cellcount")
```

   iii. Plots for single parameters (see examples in Figures 2B–2D).

```
>anno.plot_exam <- get_anno.plot_exam(huva_expression = gene_overview, study = "FG500",
dataset = "FG500_whole_blood")
>anno.plot_exam$height
>meta.plot_exam <- get_meta.plot_exam(huva_expression = gene_overview, study = "FG500",
dataset = "FG500_whole_blood_cellcount")
>meta.plot_exam$‘Monocytes (CD14+)‘
```

Note: Some of the functions require the user to set a "study" and "dataset" argument to export the results of the *exploratory huva experiment*. If the default *huva.db* is used, all available datasets/studies can be explored with the function `huva_overview()`.

Note: The distinction between annotation and metadata is due to the structure of the *huva* functions. The main formal distinction is that annotations are mandatory and do not allow for missing values but can contain both numeric and non-numeric variables. On the contrary, metadata are optional and can include missing data points without resulting in an error when running a *huva experiment* but are limited to numeric variables.

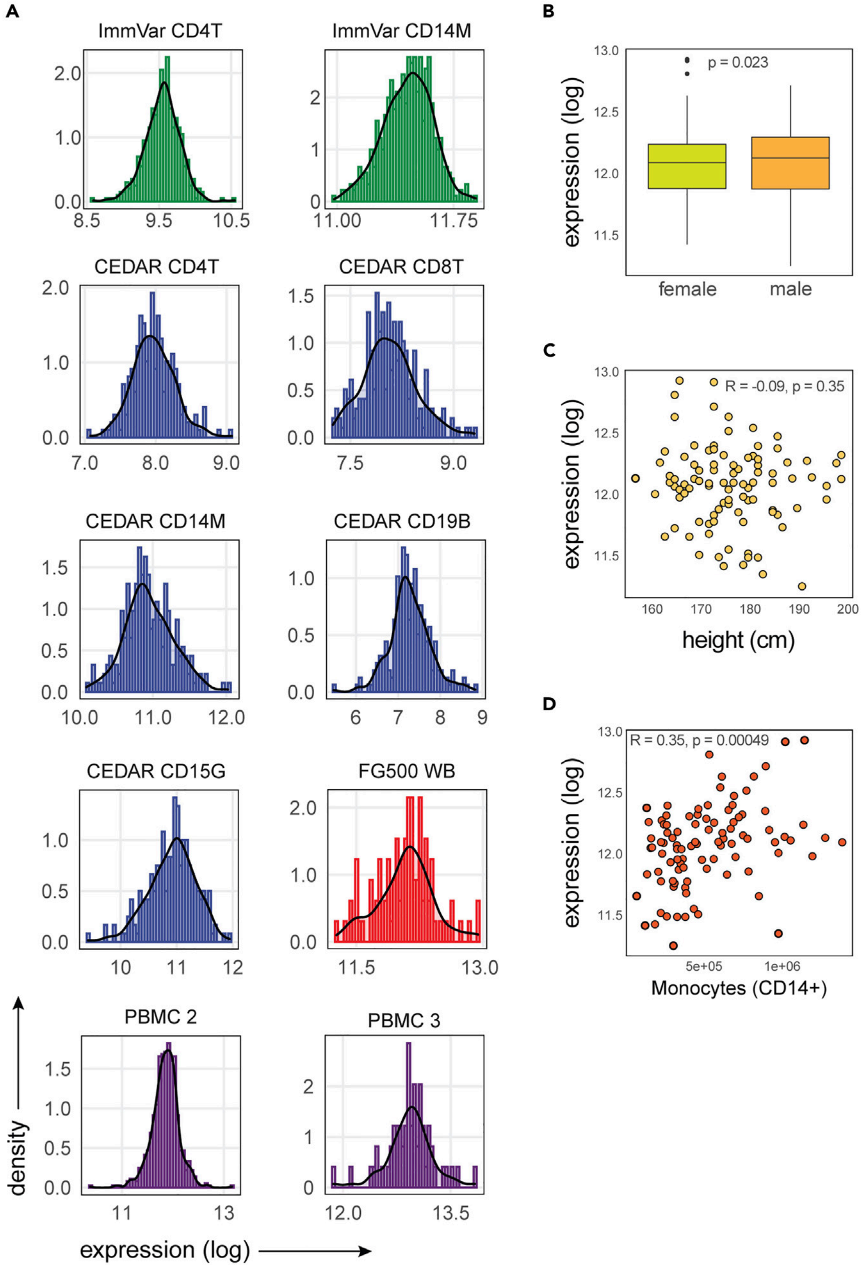


Figure 2. Exemplary result of an *exploratory huva experiment*

(A) Histogram of the expression of the gene of interest (GOI) in each dataset included in the *huva.db*. (CD4T - CD4+ T cells, CD8T - CD8+ T cells, CD14M - CD14+ monocytes, CD19B - CD19+ B cells, CD15G - CD15+ granulocytes, WB - whole blood, PBMC - peripheral blood mononucleated cells).

(B) Correlation between gene expression and sex in the FG500 whole blood dataset.

(C) Correlation between gene expression and height in the FG500 whole blood dataset. Pearson's correlation statistics are shown.

(D) Correlation between gene expression and number of CD14+ monocytes in the FG500 whole blood dataset. Pearson's correlation statistics are shown. Box plots were constructed in the style of Tukey, showing median, 25th and 75th percentiles.

huva gene experiment

Timing: 30 min

The *huva gene experiment* performs a *huva* analysis consecutively on all datasets included in the *huva.db*.

For optimization of the analysis, the user can set several parameters. First, the *huva.db* and the GOI need to be defined. Next, the quantile to be used for the analysis needs to be set together with the gene set enrichment reference and the p-value adjustment method for the differential expression (DE) analysis. For a detailed overview of the available parameters see the function documentation (`help(run_huva_experiment)`). The results of the *huva gene experiment* are collected in a *huva_experiment* R object for further exploration in each dataset.

3. First, execute the *huva gene experiment*, the results are stored in the R environment:

```
>binned_dataset <- run_huva_experiment(data = huva.db,
>                  gene = GOI,
>                  quantiles = 0.10,
>                  gs_list = hallmarks_V7.2,
>                  summ = T,
>                  datasets_list = NULL,
>                  adjust.method = "BH")
```

Note: The GOI has been set previously, see step: 1a.

Note: We provide here also a more detailed description of the parameters of the *huva gene experiment*:

data: huva database to the used for the experiment (`huva_dataset` class object).

gene: Name of the gene of interest (GOI).

quantiles: Definition of the quantile of segregation of the samples to define the *low* and *high* groups, quantiles are always symmetrical. If not differently stated, a quantile of 0.1 (10%) is used as default (quantile 0.1 will use the 10th and 90th percentiles). The selection of the quantiles determines the number of samples assigned to the *low* or *high* experimental groups. We recommend starting with 0.10 for datasets of < 100 samples and 0.05 for datasets > 100 samples. As shown in the original publication,[1] this setting does not substantially affect the biological interpretation of the results.

gs_list: List of gene sets to be used for Gene Set Enrichment Analysis, the gene list needs to be provided using the same nomenclature as the *huva.db* (e.g., HGNC symbols).

summ: TRUE or FALSE, defines if the summary of the results in the different dataset should be calculated. Default as TRUE as some of the downstream functions need the summary results. Setting to FALSE can speed up the analysis.

dataset_list: Vector used to filter the *huva.db* for the analysis for only selected datasets, if NULL (default) all datasets will be used.

adjust.method: p-value adjustment method used to correct the DE genes analysis. It can be set to "holm" (Holm), "hochberg" (Hochberg), "hommel" (Hommel), "bonferroni" (Bonferroni), "BH" (Benjamini & Hochberg), "BY" (Benjamini & Yekutieli), "fdr" (False Dircovery Rate), "none" (no correction for multiple testing).

Note: If a gene is not expressed in a dataset a message will be displayed (e.g., "MYD88 is not present in PLA")

4. Explore the results of the *huva gene experiment* using several built-in functions not only to extract the data, but also to plot, and visualize the results.
   a. Investigate the expression of a list of selected genes, including our GOI. This function computes boxplots for all selected genes in all available datasets and outputs it as a list. The code below exemplifies the visualization of the expression of three selected genes in the FG500 whole blood dataset (Figure 3A).

```
>plot_binned <- plot_binned_gene(goi = c("MYD88", "CRELD1", "STAT1" "RCAN3"), huva_experi-
ment = binned_dataset)

>plot_binned$FG500_whole_blood
```

   b. Export from the *huva gene experiment* the gene expression count table filtered for the dataset used in the *huva gene experiment*, encompassing expression values for the binned groups for all present genes expressed in the dataset of choice.

```
>expr_huva <- get_expr_huva(huva_exp = binned_dataset,

>              study = "FG500",

>              dataset = "FG500_whole_blood")
```

   c. Further, investigate the relationship between the GOI and the provided annotations and metadata with the outputs produced with the following commands.
      i. The filtered annotation or metadata table.

```
>anno_huva <- get_anno_huva(huva_exp = binned_dataset,

>              study = "FG500")

>meta_huva <- get_meta_huva(huva_exp = binned_dataset,

>              study = "FG500")
```

      ii. The statistical comparison between the *low* and *high* groups derived from the *huva gene experiment*.

```
>anno.stat <- get_anno.stat_huva(huva_exp = exper,

>                study = "FG500")

>meta.stat <- get_meta.stat_huva(huva_exp = exper,

>                study = "FG500",

> dataset = "FG500_whole_blood_cellcount")
```

      iii. Plots of single parameters for the two groups, *high* and *low* (Figures 3B and 3C).

```
>anno.plot <- get_anno.plot_huva(huva_exp = exper,

>                study = "FG500")
```

```
>anno.plot$FG500_whole_blood$age

>meta.plot <- get_meta.plot_huva(huva_exp = exper,

>                study = "FG500")

>meta.plot$FG500_whole_blood_cellcount$‘Monocytes (CD14+)‘
```

d. Export the results from differential gene expression (DE) analysis of the comparison *low* vs. *high* from the *huva gene experiment* and their visualization.

Note: *Huva* uses the Limma workflow[12] for differential expression analysis, the output includes logFC (log fold-change), average gene expression, t-value (moderated t-statistic), p-value, p-value adjusted for multiple testing (according to the setting of the `run_huva_experiment` function) and B-statistics value (log-odds that that gene is differentially expressed).

i. Export of the DE results from the *huva gene experiment* as a table. Settings can be chosen manually in the function. Defaults for filtering are logFC > 1 and corrected p-value < 0.05.

```
>DE_huva <- get_DE_huva(huva_exp = exper,

>            study = "FG500",

>            dataset = "FG500_whole_blood",

>            pval = 0.001,

>            logFC = 1)
```

ii. Using the result table (*get_DE_huva*), compute a PCA plot to visualize the separation of the *low* and *high* groups in the latent space (Figure 3D).

```
>DE_huva$PCA_FG500_whole_blood
```

iii. Generate a bar plot showing the number of DE genes (Figure 3E).

```
>DE_huva$plot_FG500_whole_blood
```

iv. Plot a heatmap of the differentially expressed genes (Figure 3F).

```
>plot_HM(DE_huva$HM_FG500_whole_blood)
```

e. Within the *huva gene experiment*, a functional enrichment on the ranked gene list is performed (GSEA[14]). This result uses the gene sets provided during the *huva gene experiment* as a reference (see step 3, `gs_list` parameter in the `run_huva_experiment` function). Explore the results with the help of built-in functions.

i. Export the ranked gene list into a separate object.

```
>rank_huva <- get_rank_huva(huva_exp = exper,

>              study = "ImmVar",

>              dataset = NULL,

>              n_top_genes = 5)
```

ii. From this result export the ranked gene list as a table for a chosen sample subset of the chosen dataset, e.g., the CD4$^+$ T cells from the ImmVar study.

```
>rank_huva$ImmVar_CD4T
```

iii. Plot the top up- and down-regulated genes (*n_top_genes*) for a chosen sample subset of the chosen dataset, e.g., the CD4^{+} T cells from the ImmVar study (Figure 3G).

```
>rank_huva$plot_ImmVar_CD4T
```

iv. Export and plot, also interactively, the result of the GSEA for a chosen sample subset of the chosen dataset, e.g., the CD4^{+} T cells from the ImmVar study. Interactive visualization was implemented to facilitate the interpretation of the plot (Figure 3H).

```
# Export the results
>gsea_huva <- get_gsea_huva(huva_exp = exper,
>              study = "FG500")
# Static visualization
>gsea_huva$plot_FG500_whole_blood
# Interactive visualization
>gsea_huva$int_plot_FG500_whole_blood
```

Note: A convenient way to perform a *huva gene experiment* is using our GUI available on FastGenomics (https://beta.fastgenomics.org/a/huva). This provides fast access to *huva* without the need for any coding skills. The main drawback of this alternative is the limited capability to export plots and tables for secondary analyses (see point 5).

Note: Similar to the *huva gene experiment*, the R implementation of the *huva* approach can be used for a *huva phenotype experiment* or *huva signature experiment*. A phenotype of interest or a single-sample signature enrichment is used to stratify the samples into *high* and *low* groups. More details on how to run these experiments can be found in the package vignette.

Note: Some of the functions require the user to set a “study” and “dataset” argument to export the results of the *huva gene experiment*. If the default *huva.db* is used, all available datasets/studies can be explored with the function `huva_overview()`.

⚠ CRITICAL: Within the *huva gene experiment*, some steps of randomization are performed for statistical testing to ensure the reproducibility of the results, see troubleshooting section.

Export *huva* results for secondary analysis

⏲ Timing: 10 min

Outputs of *huva experiments* can be exported as standard R objects (e.g., data frames or lists) to be used in other analytical pipelines. For example, the Differential Expression (DE) lists for *low* and *high* groups from a *huva gene experiment* may be used for comparison to transcriptome results from a genetic loss-of-function experiment in a model system (e.g., knock-out mouse) (i.e., input for GSEA).

The *huva gene experiment* output is similar to a standard list in R with the following structure:

```
huva_experiment
|_[study_name] #Each study provided in the huva.db
|  |_anno
|    |_[dataset_name] #Applies for all elements at this level
|  |_data
|  |_DE_genes
|  |_Rank_genelist
|  |_gsea
|  |_metadata
|
  |_summary # Provides summary statistics for some of the results
  |_Rank
    |_[study_dataset_name] #Applies for all elements at this level
  |_gsea
  |_anno
  |_metadata
```

Note: The structure of the *huva gene experiment* can be visualized with: `str(exper)`.

5. Export of the output of a *huva gene experiment* to facilitate interoperability with other analytical pipelines. Export single outputs as exemplified below.

```
# Export the filtered expression table for the ImmVar CD4+ T cell dataset
>exper$ImmVar$data$ImmVar_CD4T
# Export the filtered annotation table for the CEDAR CD8+ T cell dataset
>exper$CEDAR$anno$CEDAR_CD8T
# Export DE gene table from the CEDAR monocyte dataset
>exper$CEDAR$DE_genes$CEDAR_CD14M
# Export the ranked gene list from the ImmVar monocyte dataset
>exper$ImmVar$Rank_genelist$ImmVar_CD14M
# Export GSEA results from the CEDAR granulocyte dataset
>exper$CEDAR$gsea$CEDAR_CD15G
# Export filtered cell count metadata table from the FG500 PBMC dataset
>exper$FG500$metadata$FG500_whole_blood_cellcount
# Export summary statistics from CEDAR CD8+ T cell sample annotation
>exper$summary$anno$CEDAR_CD8T
# Export summary statistics from FG500 PBMC cytokine secretion metadata
>exper$summary$metadata$FG500_whole_blood_cytokines
```

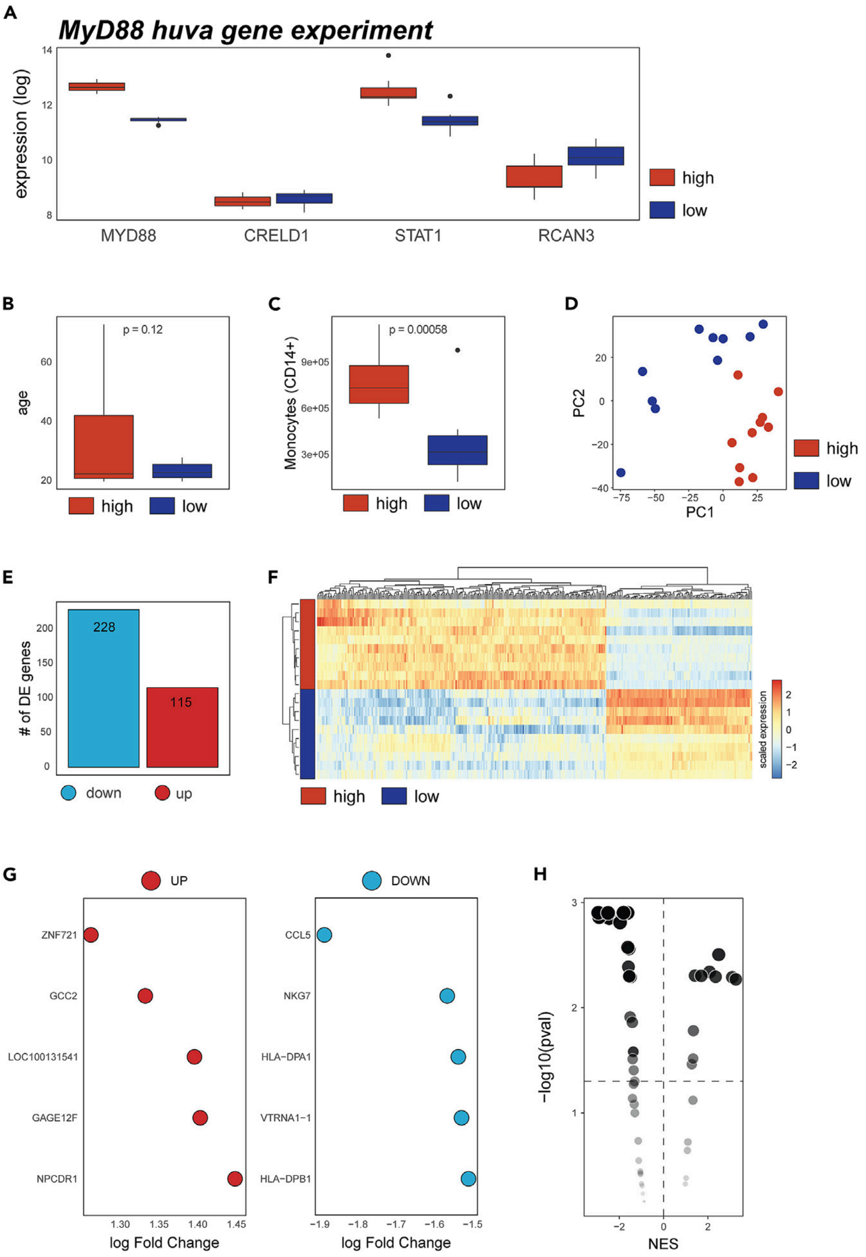


Figure 3. Exemplary result of a *huva gene experiment* for *MYD88*

(A) Boxplot showing the expression of selected genes (*MYD88*, *CRELD1*, *STAT1*, *RCAN3*) in the *low* and *high* groups for the FG500 whole blood dataset.

(B) Boxplot showing the age of the donors in the *high* and *low* groups of the FG500 whole blood dataset.

(C) Boxplot showing the number of CD14+ monocytes in the *high* and *low* groups of the FG500 whole blood dataset.

(D) Principal component analysis (PCA) plot for the transcriptome of the *low* and *high* groups in the FG500 whole blood dataset.

(E) Barplot showing the number of DE genes ($p < 0.001$, logFC > 1) in the comparison *low* vs. *high* in the FG500 whole blood dataset.

(F) Heatmap of the differentially expressed genes ($p < 0.001$, logFC > 1) in the comparison *low* vs. *high* in the FG500 whole blood dataset.

(G) Dotplot of the top 5 down- and up-regulated genes according to fold change in the comparison *low* vs. *high* in the ImmVar CD4 T^+ cell dataset.

(H) Gene Set Enrichment Analysis (GSEA) plot on the ranked gene list in the comparison *low* vs. *high* in the FG500 whole blood dataset. Box plots were constructed in the style of Tukey, showing median, 25th and 75th percentiles.

EXPECTED OUTCOMES

Huva allows the use of variation in the human population, e.g., on the transcriptome level, to predict links between gene expression and functional phenotypes of any given cell type, organ, or tissue, for which large multi-layer datasets are existing (omics, phenotypic, or clinical and functional data) are existing. With the need of only a few lines of code, the *huva experiment* provides an overview of the predicted role of a GOI. The combination of the *exploratory huva experiment* and *huva gene experiment* provides the correlation between the expression of a selected GOI and phenotypic measurement together with a well-defined list of DE genes and pathways enrichment combined with phenotypic and functional changes. Computationally, this analysis requires only few seconds (3s on average) on standard hardware.[1]

The output of the *huva experiment* can be easily exported and integrated into many downstream pipelines. This approach opens new possibilities to interrogate the increasingly available population-scale multi-layered datasets to infer gene function and relationships between phenotypes and expression. This aids the hypothesis generation to prioritize experiments in humans or appropriate model systems.[1,2]

QUANTIFICATION AND STATISTICAL ANALYSIS

A detailed description of the *huva* method is reported in the original publication.[1] We further refer to the reader to the source code (see key resources table) for a detailed description of the statistical methods.

LIMITATIONS

The *huva* approach contrasts individuals with high and low expression of a GOI to predict the functional role of a given gene.[1] A limitation of the *huva* approach is its inability to infer causality between the GOI and the predicted phenotype, for which further supporting experimental data are required (e.g., assessment of human mutations, genetic model systems such as CRISPR-Cas KO). Furthermore, *huva* relies on the assumption that variation in gene expression is linked to a functional phenotype. While all tested examples so far have shown this assumption to be valid,[1] extension to all expressed genes depends on further experimental validation. *huva* provides a complete set of predicted functional phenotypes for all other genes that can serve as the basis for these tests. In addition, with the built-in *huva.db*, the *huva* experiment focuses on human circulating immune cells. If the user wants to study a GOI in the context of another cell type or organ, a new dataset needs to be provided (e.g., GTEx v8).

The *huva* R package offers a prime example of a tool to investigate variation in human cohorts. At the moment, the included datasets in the *huva* package can be used to address gene functions within the human immune system. Yet, the tool can be easily expanded to other tissues/cell types.

TROUBLESHOOTING

Problem 1

If technical limitations prohibit the installation of the Docker runtime in your computing environment (prerequisites/system requirements), it is possible to install *huva*, *huva.db*, and all dependencies locally on a workstation (step: Set up a *huva* workspace).

Potential solution

- Download and install R v. 4.0.1 from https://cran.r-project.org/. A newer version of R should be compatible with *huva* v 0.1.5 but was not formally tested.
- From R, install the Bioconductor package manager:

```
>if (!require("BiocManager", quietly = TRUE))

>  install.packages("BiocManager")

>BiocManager::install(version = "3.12")
```

- Install *huva* dependencies.

```
>BiocManager::install(c("ggplot2", "Rmisc", "ggpubr", "reshape2",
>          "ggsci", "plotly", "knitr", "pheatmap",
>          "useful", "rmarkdown", "fgsea",
>          "limma", "GSVA"),
>          version = "3.12")
```

- Download the *huva* and *huva.db* packages from Zenodo.

```
>download.file("https://zenodo.org/record/7088729/files/huva_0.1.5.tar.gz",         destfile         =
"./huva_0.1.5.tar.gz")
>download.file("https://zenodo.org/record/7088729/files/huva.db_0.1.5.tar.gz",        destfile         =
"./huva.db_0.1.5.tar.gz")
```

- Install *huva.db*.

```
>install.packages("./huva.db_0.1.5.tar.gz",
>       repos = NULL,
>       type = "source")
```

- Install *huva*.

```
>install.packages("./huv_0.1.5.tar.gz",
>       repos = NULL,
>       type = "source")
```

Problem 2

The provided normalized count table uses GeneIDs or ENSEMBL IDs instead of HGNC symbols leading to no GSEA results (Format a new dataset).

Potential solution

We recommend using rlog normalized count tables formatted with gene symbols. If another annotation is preferred, an updated reference for the GSEA needs to be provided. The structure of the default hallmark GSEA reference can be inspected with `View(huva.db::hallmarks_V7.2)` and used for preparation of a new reference with the desired gene annotation. The new reference will need to be defined in the *run_huva_experiment* function.

```
>binned_dataset_new <- run_huva_experiment(data = gtex.db,
>                    gene = "[GOI]",
>                    quantiles = 0.05,
>                    gs_list = [new_reference],
>                    summ = T,

>                    datasets_list = NULL,

>                    adjust.method = "none")
```

Problem 3

The results of the *huva gene experiment* slightly change at every run despite having the same settings.

Potential solution

To solve this problem a seed can be set to ensure reproducibility of the *huva gene experiment* (see `help(set.seed)`) by adding the following line of code before the *huva experiment*.

```
>set.seed(1234) # Any number can be used
```

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Lorenzo Bonaguro (lorenzobonaguro@uni-bonn.de).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The code used for the analysis shown in this manuscript was deposited on GitHub and Zenodo (see key resources table). The *huva* code was previously reported[1] and is available on GitHub and Zenodo (access number provided in the key resources table). The GTEx v8 dataset was downloaded on 13.10.2022 from the online portal (https://www.gtexportal.org/home/datasets), and raw counts were used for the analysis.

ACKNOWLEDGMENTS

We thank Joachim L. Schultze for the critical reading of the manuscript.

A.C.A. is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) via grant 458854699 (AS637/1-1), grant 466168337 (AS637/2-1), grant 466168626 (AS637/3-1), and SFB 1454 – project number 432325352; the BMBF-funded project IMMME (01EJ2204D); and the EU-funded project NEUROCOV. L.B. is supported by the BMBF-funded excellence project Diet-Body-Brain (DietBB) under grant number 01EA1809A. L.B. is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) via grant 513977171 (ImmuDiet BO 6228/2-1). The Genotype-Tissue Expression (GTEx) Project was supported by the Common Fund of the Office of the Director of the National Institutes of Health and by NCI, NHGRI, NHLBI, NIDA, NIMH, and NINDS. The data used for the analyses described in this manuscript were obtained from the GTEx Portal on 13.10.2022.

AUTHOR CONTRIBUTIONS

Conceptualization, L.B., A.C.A.; methodology, L.B., A.C.A.; software, L.B.; investigation, L.B.; visualization and data interpretation, L.B.; writing, L.B., A.C.A.; funding acquisition, L.B., A.C.A.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

1. Bonaguro, L., Schulte-Schrepping, J., Carraro, C., Sun, L.L., Reiz, B., Gemünd, I., Saglam, A., Rahmouni, S., Georges, M., Arts, P., et al. (2022). Human variation in population-wide gene expression data predicts gene perturbation phenotype. iScience *25*, 105328. https://doi.org/10.1016/j.isci.2022.105328.

2. Bonaguro, L., Köhne, M., Schmidleithner, L., Schulte-Schrepping, J., Warnat-Herresthal, S., Horne, A., Kern, P., Günther, P., Ter Horst, R., Jaeger, M., et al. (2020). CRELD1 modulates homeostasis of the immune system in mice and humans. Nat. Immunol. *21*, 1517–1527. https://doi.org/10.1038/s41590-020-00811-2.

3. Li, Y., Oosting, M., Smeekens, S.P., Jaeger, M., Aguirre-Gamboa, R., Le, K.T.T., Deelen, P., Ricaño-Ponce, I., Schoffelen, T., Jansen, A.F.M., et al. (2016). A functional genomics approach to understand variation in cytokine production in humans. Cell *167*, 1099–1110.e14. https://doi.org/10.1016/j.cell.2016.10.017.

4. Ter Horst, R., Jaeger, M., Smeekens, S.P., Oosting, M., Swertz, M.A., Li, Y., Kumar, V., Diavatopoulos, D.A., Jansen, A.F.M., Lemmers, H., et al. (2016). Host and environmental factors influencing individual human cytokine responses. Cell *167*, 1111–1124.e13. https://doi.org/10.1016/j.cell.2016.10.018.

5. Raj, T., Rothamel, K., Mostafavi, S., Ye, C., Lee, M.N., Replogle, J.M., Feng, T., Lee, M., Asinovski, N., Frohlich, I., et al. (2014). Polarization of the effects of autoimmune and neurodegenerative risk alleles in leukocytes. Science *344*, 519–523. https://doi.org/10.1126/science.1249547.

6. Momozawa, Y., Dmitrieva, J., Théâtre, E., Deffontaine, V., Rahmouni, S., Charloteaux, B., Crins, F., Docampo, E., Elansary, M., Gori, A.S., et al. (2018). IBD risk loci are enriched in multigenic regulatory modules encompassing putative causative genes. Nat. Commun. *9*, 2427. https://doi.org/10.1038/s41467-018-04365-8.

7. GTEx Consortium (2013). The genotype-tissue expression (GTEx) project. Nat. Genet. *45*, 580–585. https://doi.org/10.1038/ng.2653.

8. Love, M.I., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. *15*, 550. https://doi.org/10.1186/s13059-014-0550-8.

9. Zhang, Y., Parmigiani, G., and Johnson, W.E. (2020). ComBat-seq: batch effect adjustment for RNA-seq count data. NAR Genom. Bioinform. *2*, lqaa078. https://doi.org/10.1093/nargab/lqaa078.

10. Love, M.I., Anders, S., Kim, V., and Huber, W. (2015). RNA-Seq workflow: gene-level exploratory analysis and differential expression. F1000Research *4*, 1070. https://doi.org/10.12688/f1000research.7035.1.

11. GTEx Consortium (2020). The GTEx Consortium atlas of genetic regulatory effects across human tissues. Science *369*, 1318–1330. https://doi.org/10.1126/science.aaz1776.

12. Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., and Smyth, G.K. (2015). Limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. *43*, e47. https://doi.org/10.1093/nar/gkv007.

13. Korotkevich, G., Sukhov, V., Budin, N., Shpak, B., Artyomov, M.N., and Sergushichev, A. (2016). Fast gene set enrichment analysis. Preprint at bioRxiv. https://doi.org/10.1101/060012.

14. Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., and Mesirov, J.P. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. USA *102*, 15545–15550. https://doi.org/10.1073/pnas.0506580102.