METHOD ARTICLE

REVISED Using regulatory genomics data to interpret the function of disease variants and prioritise genes from expression studies [version 2; referees: 2 approved]

Enrico Ferrero

Computational Biology, GSK, Medicines Research Centre, Stevenage, SG1 2NY, UK

v2 **First published:** 29 Jan 2018, **7**:121 (doi: 10.12688/f1000research.13577.1)
Latest published: 23 Feb 2018, **7**:121 (doi: 10.12688/f1000research.13577.2)

Abstract

The identification of therapeutic targets is a critical step in the research and developement of new drugs, with several drug discovery programmes failing because of a weak linkage between target and disease.
Genome-wide association studies and large-scale gene expression experiments are providing insights into the biology of several common diseases, but the complexity of transcriptional regulation mechanisms often limits our understanding of how genetic variation can influence changes in gene expression. Several initiatives in the field of regulatory genomics are aiming to close this gap by systematically identifying and cataloguing regulatory elements such as promoters and enhacers across different tissues and cell types.
In this Bioconductor workflow, we will explore how different types of regulatory genomic data can be used for the functional interpretation of disease-associated variants and for the prioritisation of gene lists from gene expression experiments.

This article is included in the RPackage gateway.

This article is included in the Bioconductor gateway.

Open Peer Review

Referee Status: ✓ ✓

	Invited Referees 1	2
REVISED version 2 published 23 Feb 2018	✓ report	✓
version 1 published 29 Jan 2018	? report	? report

1 **Aaron T. L. Lun**, University of Cambridge, UK

2 **Vincent J. Carey**, Brigham and Women's Hospital and Harvard Medical School, USA

Discuss this article

Comments (0)

Corresponding author: Enrico Ferrero (enrico.x.ferrero@gsk.com)

Author roles: Ferrero E: Conceptualization, Formal Analysis, Investigation, Methodology, Project Administration, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: EF is a full time employee of GSK.

How to cite this article: Ferrero E. **Using regulatory genomics data to interpret the function of disease variants and prioritise genes from expression studies [version 2; referees: 2 approved]** *F1000Research* 2018, **7**:121 (doi: 10.12688/f1000research.13577.2)

Grant information: The author(s) declared that no grants were involved in supporting this work.

First published: 29 Jan 2018, **7**:121 (doi: 10.12688/f1000research.13577.1)

REVISED **Amendments from Version 1**

Several changes were made to the revised version of this workflow to address the referees' comments.
The main ones were:
- Most object dumps were removed, thus making the article easier to follow;
- Several sections were reworded or expanded to clarify ambiguous points or to provide more context and background.
- A number of sentences were removed or shortened, especially in the Introduction and the 'Gene expression data and differential gene expression analysis' section.
- Three new figures were added and discussed to illustrate the overall methodology and steps involved (Figure 1) and to provide more insights into the overall results (Figure 7 and Figure 8);
- Existing figures and figure legends were revised and annotated to make them clearer and more useful.
- All available samples were used in the analysis and were better characterised.
- The InteractionSet package was used to represent and operate on the promoter-capture Hi-C data.
- A new section, 'Functional analysis of prioritised hits' was added to provide a better characterisation of the final results from both biological and drug discovery perspectives.

See referee reports

Introduction

Discovering and bringing new drugs to the market is a long, expensive and inefficient process[1,2]. The majority of drug discovery programmes fail for efficacy reasons[3], with up to 40% of these failures due to lack of a clear link between the target and the disease under investigation[4]. Target selection, the first step in drug discovery programmes, is thus a critical decision point. It has previously been shown that therapeutic targets with a genetic link to the disease under investigation are more likely to progress through the drug discovery pipeline, suggesting that genetics can be used as a tool to prioritise and validate drug targets in early discovery[5,6].

One of the biggest challenges in translating findings from genome-wide association studies (GWASs) to therapies is that the great majority of single nucleotide polymorphisms (SNPs) associated with disease are found in non-coding regions of the genome, and therefore cannot be easily linked to a target gene[7]. Many of these SNPs could be regulatory variants, affecting the expression of nearby or distal genes by interfering with the transcriptional process[8].

The most established way to map disease-associated regulatory variants to target genes is to use expression quantitative trait loci (eQTLs)[9], variants that affect the expression of specific genes. The GTEx consortium profiled eQTLs across 44 human tissues by performing a large-scale mapping of genome-wide correlations between genetic variants and gene expression[10]. However, depending on the power of the study, it might not be possible to detect all existing regulatory variants as eQTLs. An alternative is to use information on the location of promoters and distal enhancers across the genome and link these regulatory elements to their target genes. Large, multi-centre initiatives such as ENCODE[11], Roadmap Epigenomics[12] and BLUEPRINT[13,14] mapped regulatory elements in the genome by profiling a number of chromatin features, including DNase hypersensitive sites (DHSs), several types of histone marks and binding of chromatin-associated proteins in a large number of cells and tissues. Similarly, the FANTOM consortium used cap analysis of gene expression (CAGE) to identify promoters and enhancers across hundreds of cells and tissues[15].

Knowing that a certain stretch of DNA is an enhancer is however not informative of the target gene(s). One way to infer links between enhancers and promoters *in silico* is to identify significant correlations across a large panel of cell types, an approach that was used for distal and promoter DHSs[16] as well as for CAGE-defined promoters and enhancers[17]. Experimental methods to assay interactions between regulatory elements also exist. Chromatin interaction analysis by paired-end tag sequencing (ChIA-PET)[18,19] couples chromatin immunoprecipitation with DNA ligation to identify DNA regions interacting thanks to the binding of a specific protein. Promoter capture Hi-C[20,21] extends chromatin conformation capture by using "baits" to enrich for promoter interactions and increase resolution.

Overall, linking genetic variants to their candidate target genes is not straightforward, not only because of the complexity of the human genome and transcriptional regulation, but also because of the variety of data types and approaches that can be used. To address this problem, we developed STOPGAP, a database of disease variants mapped to their most likely target gene(s) using several different types of regulatory genomic data[22]. The database is currently undergoing a

major overhaul and will eventually be superseded by POSTGAP. A valid and recent alternative is INFERNO[23], though it does only rely on eQTL data for target gene assignment. These resources implement some or all of the approaches that will be reviewed in the workflow and constitute good entry points for identifying the most likely target gene(s) of regulatory SNPs. However, as they tend to hide much of the complexity involved in the process, we will not use them and rely on the original datasets instead.

In this workflow, we will explore how regulatory genomic data can be used to connect the genetic and transcriptional layers by providing a framework for the discovery of novel therapeutic targets. We will use eQTL data from GTEx[10], FANTOM5 correlations between promoters and enhancers[17] and promoter capture Hi-C data[21] to annotate significant GWAS variants to putative target genes and to prioritise genes obtained from a differential expression analysis (Figure 1).

Workflow

Install required packages

R version 3.4.2 and Bioconductor version 3.6 were used for the analysis. The code below will install all required packages and dependencies from Bioconductor and CRAN:

```
source("https://bioconductor.org/biocLite.R")
# uncomment the following line to install packages
#biocLite(c("clusterProfiler", "DESeq2", "GenomicFeatures",
"GenomicInteractions", "GenomicRanges", "ggplot2", "Gviz", "gwascat",
"InteractionSet", "recount", "pheatmap", "RColorBrewer", "rtracklayer",
"R.utils", "splitstackshape", "VariantAnnotation"))
```

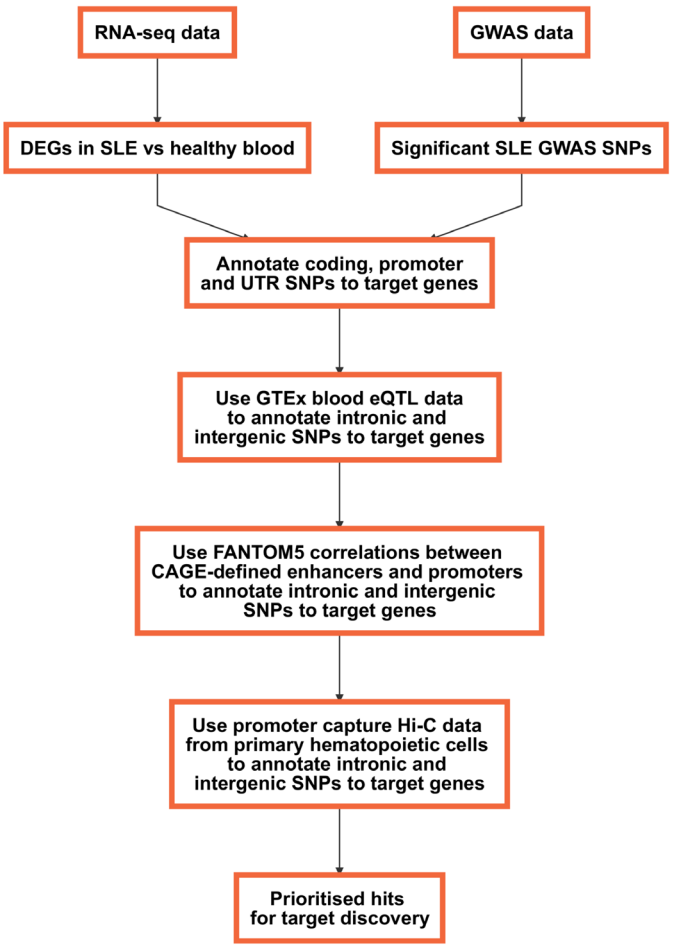

Figure 1. Diagram showing a schematic representation of the workflow and the steps involved.

Gene expression data and differential gene expression analysis

We start with a common scenario: we ran a RNA-seq experiment comparing patients with a disease and healthy individuals, and would like to discover key disease genes and potential therapeutic targets by integrating genetic information in our analysis.

The RNA-seq data we will be using comes from blood of patients with systemic lupus erythematosus (SLE) and healthy controls[24]. SLE is a chronic autoimmune disorder that can affect several organs with a significant unmet medical need[25]. It is a complex and remarkably heterogeneous disease, in terms of both genetics and clinical manifestations[26]. Early diagnosis and classification of SLE remain extremely challenging[27].

In the original study[24], the authors explore transcripts bound by Ro60, an RNA-binding protein against which some SLE patients produce autoantibodies. They identify Alu retroelements among these transcripts and use RNA-seq data to check their expression levels, observing that Alu elements are significantly more expressed in SLE patients, and particularly in those patients with anti-Ro antibodies and with a higher interferon signature metric (ISM).

We are going to use `recount`[28] to obtain gene-level counts:

```
library(recount)
# uncomment the following line to download dataset
#download_study("SRP062966")
load(file.path("SRP062966", "rse_gene.Rdata"))
rse <- scale_counts(rse_gene)
```

Other Bioconductor packages that can be used to access data from gene expression experiments directly in R are `GEOquery`[29] and `ArrayExpress`[30].

We have 117 samples overall. This is what the matrix of counts looks like:

```
assay(rse)[1:3, 1:3]
##                     SRR2443263  SRR2443262  SRR2443261
## ENSG00000000003.14          19           6          10
## ENSG00000000005.5            0           0           0
## ENSG00000000419.12         489         238         224
```

Each gene is a row and each sample is a column. We note that genes are annotated using the GENCODE[31] v25 annotation, which will be useful later on.

To check how we can split samples between cases and controls, we can have a look at the metadata contained in the `characteristics` column, which is a `CharacterList` object:

```
head(rse$characteristics, 3)
```

```
## CharacterList of length 3
## [[1]] disease status: healthy tissue: whole blood anti-ro: control ism:
control
## [[2]] disease status: healthy tissue: whole blood anti-ro: control ism:
control
## [[3]] disease status: healthy tissue: whole blood anti-ro: control ism:
control
```

We have information about the disease status of the sample, the tissue of origin, the presence and level of anti-ro autoantibodies and the value of the ISM. However, we note that basic information such as age or gender is missing.

We can create some new columns with the available information so that they can be used for downstream analyses. We will also make sure that they are encoded as factors and that the correct reference layer is used:

```
# disease status
rse$disease_status <- sapply(rse$characteristics, "[", 1)
rse$disease_status <- sub("disease status: ", "", rse$disease_status)
rse$disease_status <- sub("systemic lupus erythematosus \\(SLE\\)", "SLE",
rse$disease_status)
rse$disease_status <- factor(rse$disease_status, levels = c("healthy", "SLE"))
# tissue
rse$tissue <- sapply(rse$characteristics, "[", 2)
rse$tissue <- sub("tissue: ", "", rse$tissue)
rse$tissue <- factor(rse$tissue)
# anti-ro
rse$anti_ro <- sapply(rse$characteristics, "[", 3)
rse$anti_ro <- sub("anti-ro: ", "", rse$anti_ro)
rse$anti_ro <- factor(rse$anti_ro)
# ism
rse$ism <- sapply(rse$characteristics, "[", 4)
rse$ism <-sub("ism: ", "", rse$ism)
rse$ism <- factor(rse$ism)
```

We can check how many samples we have in each group (note that we ignore `tissue` as it's always `whole blood`):

```
metadata <- data.frame(disease_status = rse$disease_status, anti_ro.ism =
paste(rse$anti_ro, rse$ism, sep = "."))
table(metadata)
```

```
##               anti_ro.ism
## disease_status control.control high.ISM_high high.ISM_low med.ISM_high
##        healthy              18             0            0            0
##        SLE                   0            23            1           21
##               anti_ro.ism
## disease_status med.ISM_low none.ISM_high none.ISM_low
##        healthy           0             0           0
##        SLE               2            31          21
```

Now we are ready to perform a simple differential gene expression analysis with `DESeq2`[32]. Note that we remove genes with a low number of counts (less than 50 across all 117 samples) to speed up execution and reduce the memory footprint:

```
library(DESeq2)
dds <- DESeqDataSet(rse, ~ disease_status)
dds <- DESeq(dds)
dds <- dds[rowSums(counts(dds)) >= 50, ]
```

We used an extremely simple model; in the real world we should be accounting for co-variables, potential confounders and interactions between them. For example, age and gender are usually included in this type of analysis, but we don't have access to this information for this dataset. Similarly, the value of the ISM and the presence of anti-Ro autoantibodies can't be included in the analysis due to the fact that these variables are collinear with the disease status variable (i.e.: the value of both `anti_ro` and `ism` is `control` for all samples with `disease_status` equal to `healthy`.) Like `DESeq2`, `edgeR`[33] and `limma`[34] can also deal with multiple cofactors and different experimental designs, and constitute good alternatives for performing differential expression analyses.

We can now look at the data in more detail to assess if we can observe a separation between the SLE and healthy samples and whether any batch effect is visible. We use the variance stabilising transformation (VST)[35] for visualisation purposes:

```
vsd <- vst(dds, blind = FALSE)
```

We will use the `pheatmap` and `RColorBrewer` packages to perform hierarchical clustering of the samples (Figure 2):

```
library(pheatmap)
library(RColorBrewer)
sampleDists <- dist(t(assay(vsd)))
sampleDistMatrix <- as.matrix(sampleDists)
annotation = data.frame(colData(vsd)[c("anti_ro", "ism", "disease_status")],
row.names = rownames(sampleDistMatrix))
colors <- colorRampPalette(rev(brewer.pal(9, "Blues")))(255)
pheatmap(sampleDistMatrix, clustering_distance_rows = sampleDists,
clustering_distance_cols = sampleDists, clustering_method = "complete",
annotation_col = annotation, col = colors, show_rownames = FALSE,
show_colnames = FALSE, cellwidth = 2, cellheight = 2)
```

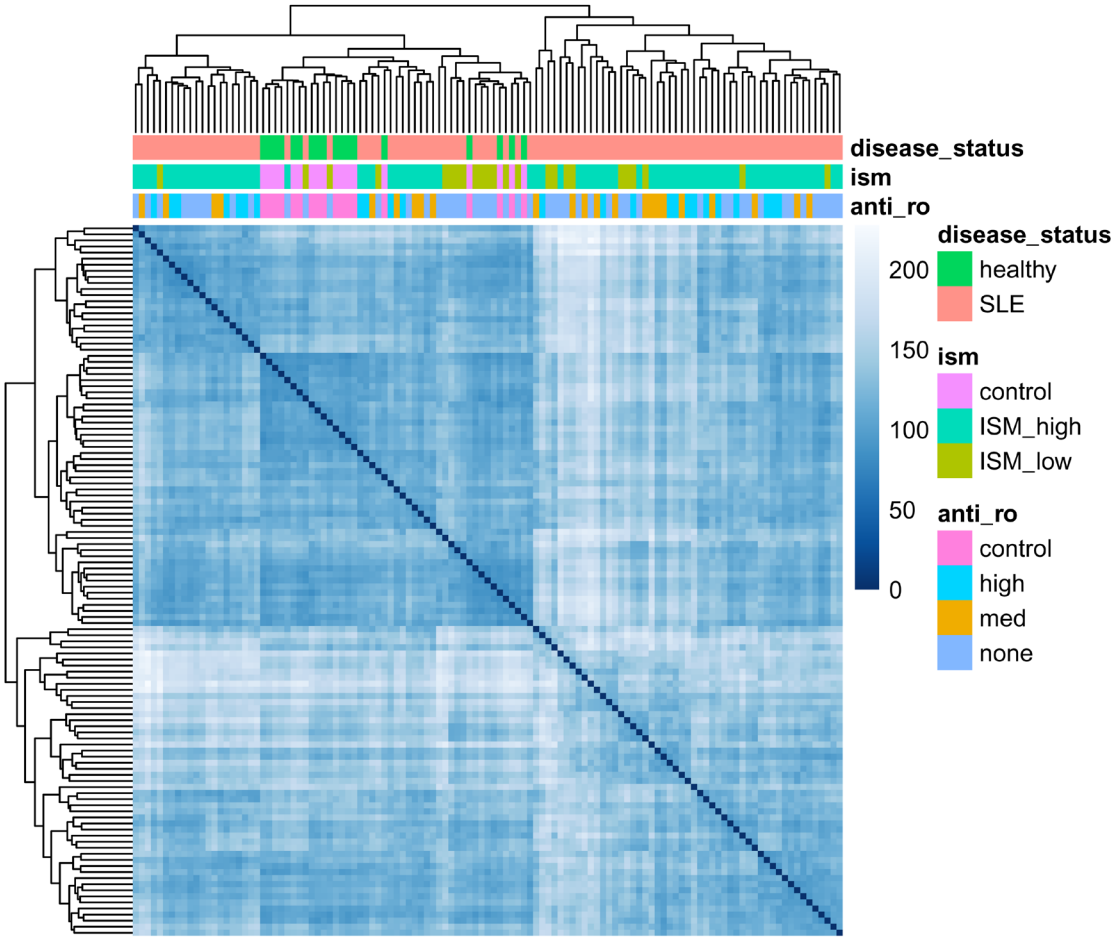

Figure 2. Heatmap showing Euclidean distances between samples clustered using complete linkage. Disease status and other experimental factors are visualised as column annotations.

While there isn't an unambiguous split between healthy and disease samples, the most distinct clusters (bottom right and top left) are entirely composed of SLE samples, with the central cluster containing all healthy samples and a number of SLE ones. The clusters don't appear to be due to the ISM or the presence of anti-Ro autoantibodies.

Similarly, we can perform a principal component analysis (PCA) on the most variable 500 genes (Figure 3). Note that we load `ggplot2`[36] to modify the look of the plot:

```
library(ggplot2)
plotPCA(vsd, intgroup = "disease_status") +
  coord_fixed()
```

We can see some separation of healthy and SLE samples along both PC1 and PC2, though some SLE samples appear very similar to the healthy ones. No obvious batch effects are visible from this plot.

Next, we select genes that are differentially expressed below a 0.05 adjusted *p*-value threshold:

```
res <- results(dds, alpha = 0.05)
summary(res)
```

```
##
## out of 32820 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)      : 4829, 15%
## LFC < 0 (down)    : 2709, 8.3%
## outliers [1]      : 0, 0%
## low counts [2]    : 2548, 7.8%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

We can visualise the shrunken log2 fold changes using an MA plot (Figure 4):

```
res_lfc <- lfcShrink(dds, coef = 2)
plotMA(res_lfc, ylim = c(-3, 3))
```

We observe large numbers of genes differentially expressed in both directions and across a range of fold changes, though the majority of significant genes appear to be upregulated in disease.

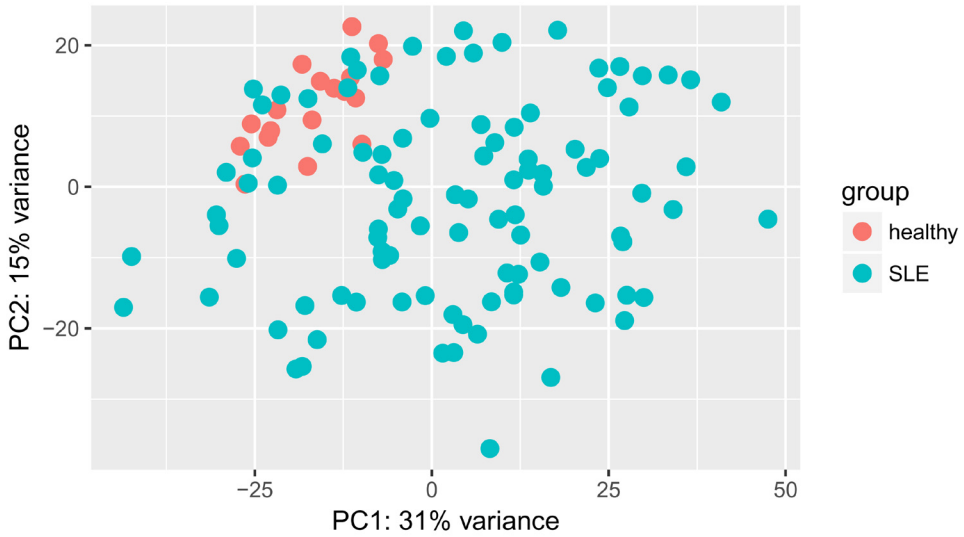

Figure 3. Scatter plot showing results of a PCA with samples coloured according to their disease status.

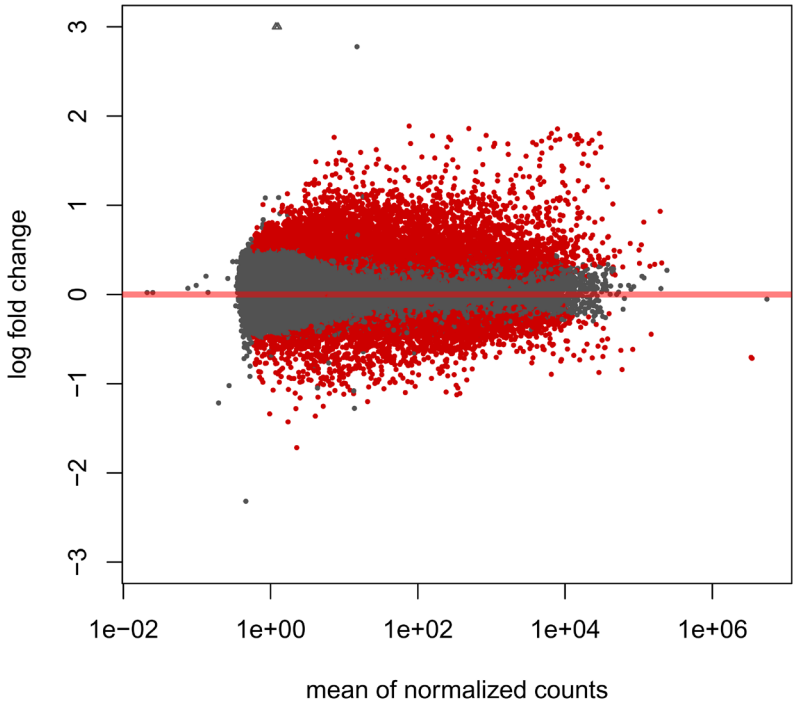

Figure 4. MA plot showing genes differentially expressed (red dots) in SLE patients compared to healthy patients.

For convenience, we will save our differentially expressed genes (DEGs) in another object and map the GENCODE gene IDs to gene symbols using the annotation in the original `RangedSummarizedExperiment object`

```
degs <- subset(res, padj < 0.05)
degs <- merge(rowData(rse), as.data.frame(degs), by.x = "gene_id", by.y =
"row.names", all = FALSE)
head(degs, 3)
```

```
## DataFrame with 3 rows and 9 columns
##                          gene_id bp_length symbol   baseMean
##                      <character> <integer> <list>  <numeric>
## ENSG00000000003 ENSG00000000003.14      4535 TSPAN6   8.739822
## ENSG00000000419 ENSG00000000419.12      1207   DPM1 431.485085
## ENSG00000000457 ENSG00000000457.13      6883  SCYL3 686.579323
##                 log2FoldChange      lfcSE      stat       pvalue
##                      <numeric>  <numeric> <numeric>    <numeric>
## ENSG00000000003     -0.4750382 0.18374822 -2.585267 9.730366e-03
## ENSG00000000419      0.5559772 0.10117967  5.494950 3.908216e-08
## ENSG00000000457      0.1927081 0.05928191  3.250707 1.151185e-03
##                          padj
##                     <numeric>
## ENSG00000000003 4.161281e-02
## ENSG00000000419 2.182977e-06
## ENSG00000000457 7.922475e-03
```

Accessing GWAS data

The differential expression analysis resulted in several thousands of DEGs. Since we know that genes with high levels of differential expression are more likely to harbour disease-associated variants[37] and that therapeutic targets with genetic evidence are more likely to progress through the drug discovery pipeline[6], one way to prioritise them is to check which of these can be genetically linked to SLE. To get hold of relevant GWAS data, we will be using the `gwascat`

Bioconductor package[38], which provides an interface to the GWAS catalog[39]. An alternative is to use the GRASP[40] database with the `grasp2db`[41] package.

```
library(gwascat)
# uncomment the following line to download file and build the gwasloc object
all in one step
#snps <- makeCurrentGwascat()
# uncomment the following line to download file
#download.file("http://www.ebi.ac.uk/gwas/api/search/downloads/alternative",
destfile = "gwas_catalog_v1.0.1-associations_e90_r2017-12-04.tsv")
snps <- read.delim("gwas_catalog_v1.0.1-associations_e90_r2017-12-04.tsv",
check.names = FALSE, stringsAsFactors = FALSE)
snps <- gwascat:::gwdf2GRanges(snps, extractDate = "2017-12-04")
genome(snps) <- "GRCh38"
head(snps, 3)
```

```
## gwasloc instance with 3 records and 37 attributes per record.
## Extracted:  2017-12-04
## Genome:  GRCh38
## Excerpt:
## GRanges object with 3 ranges and 3 metadata columns:
##       seqnames                ranges strand | DISEASE/TRAIT        SNPS
##          <Rle>             <IRanges>  <Rle> |   <character> <character>
##   [1]     chr1 [203186754, 203186754]      * | YKL-40 levels   rs4950928
##   [2]    chr13 [ 39776775,  39776775]      * |     Psoriasis   rs7993214
##   [3]    chr15 [ 78513681,  78513681]      * |   Lung cancer   rs8034191
##         P-VALUE
##       <numeric>
##   [1]     1e-13
##   [2]     2e-06
##   [3]     3e-18
##   -------
##   seqinfo: 23 sequences from GRCh38 genome; no seqlengths
```

`snps` is a `gwasloc` object which is simply a wrapper around a `GRanges` object, the standard way to represent genomic ranges in Bioconductor.

We note here that the GWAS catalog uses GRCh38 coordinates, the same assembly used in the GENCODE v25 annotation. When integrating genomic datasets from different sources it is essential to ensure that the same genome assembly is used, especially because many datasets in the public domain are still using GRCh37 coordinates. As we will see below, it is possible and relatively straightforward to convert genomic coordinates between genome assemblies.

We can select only SNPs that are associated with SLE:

```
snps <- subsetByTraits(snps, tr = "Systemic lupus erythematosus")
```

We can visualise these as a Manhattan plot to look at the distribution of GWAS *p*-values over chromosomes on a negative $\log_{10}$ scale (Figure 5): Note that *p*-values lower than 1×10^{-25} are truncated in the figure:

```
traitsManh(gwr = snps, sel = snps, traits = "Systemic lupus erythematosus") +
  xlab("SLE GWAS SNPs") +
  ylab("-log10(p-value)") +
  theme(legend.position = "none",
        strip.text.x = element_text(size = 6),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank())
```

We observe several hits across most chromosomes, with many of them below a genome-wide significant threshold (*p*-value $< 1 \times 10^{-8}$), suggesting that genetics plays an important role in the pathogenesis of SLE.

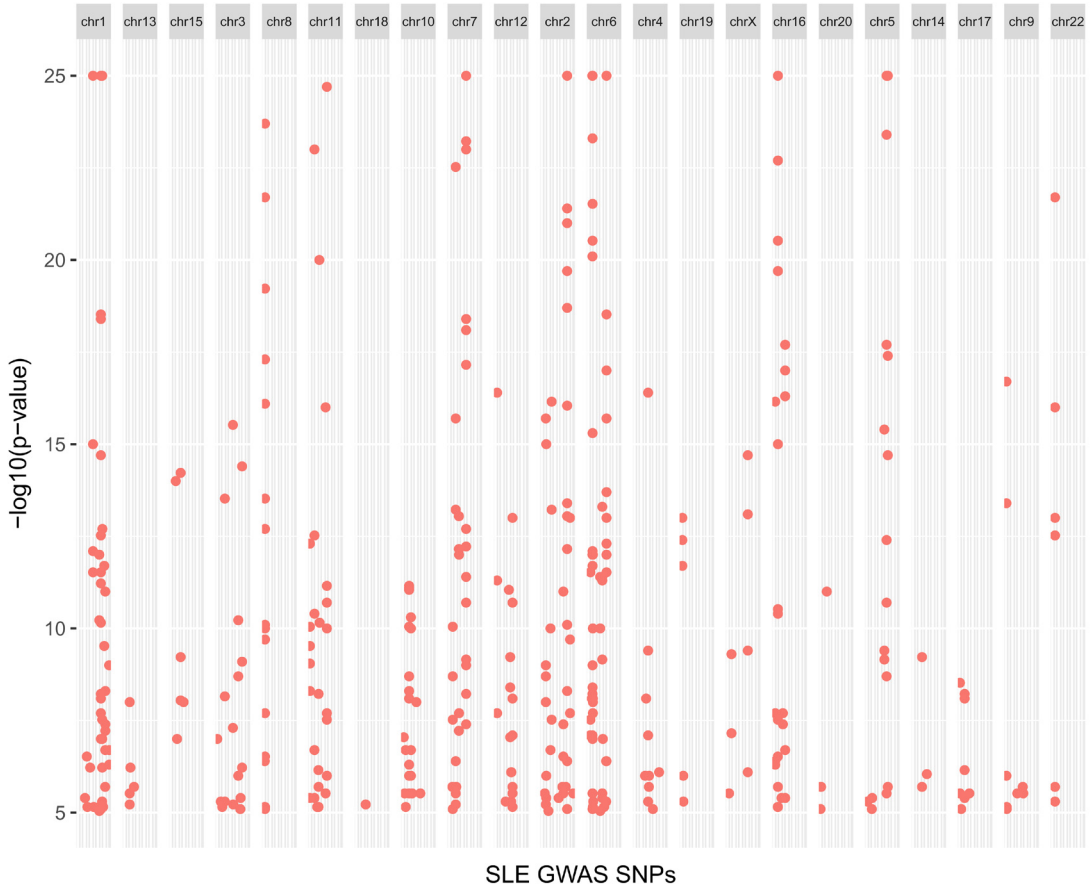

Figure 5. Manhattan plot showing GWAS variants significantly associated with SLE.

We note here that genotyping arrays typically include a very small fraction of all possible SNPs in the human genome, and there is no guarantee that the *tag* SNPs on the array are the true casual SNPs[42]. The alleles of other SNPs can be imputed from tag SNPs thanks to the structure of linkage disequilibrium (LD) blocks present in chromosomes. Thus, when linking variants to target genes in a real-world setting, it is important to take into consideration neighbouring SNPs that are in high LD (e.g.: $r^2 > 0.8$) and inherited with the tag SNPs. Unfortunately, at the time of writing there is no straightforward way to perform this LD expansion step using R or Bioconductor packages, possibly because of the large amount of reference data required. The `ldblock` package[43] used to provide this functionality by downloading the HapMap data from the NCBI website, but the dataset was retired in 2016. At present, the best option to do this programmatically is probably to query the Ensembl REST API[44].

Annotation of coding and proximal SNPs to target genes

In order to annotate these variants, we need a a `TxDb` object, a reference of where transcripts are located on the genome. We can build this using the `GenomicFeatures`[45] package and the GENCODE v25 gene annotation:

```
library(GenomicFeatures)
# uncomment the following line to download file
#download.file("ftp://ftp.sanger.ac.uk/pub/gencode/Gencode_human/release_25/
gencode.v25.annotation.gff3.gz", destfile = "gencode.v25.annotation.gff3.gz")
txdb <- makeTxDbFromGFF("gencode.v25.annotation.gff3.gz")
txdb <- keepStandardChromosomes(txdb)
```

We also have to convert the `gwasloc` object into a standard `GRanges` object:

```
snps <- GRanges(snps)
```

Let's check if the `gwasloc` and `TxDb` object use the same notation for chromosomes:

```
seqlevelsStyle(snps)
```

```
## [1] "UCSC"
```

```
seqlevelsStyle(txdb)
```

```
## [1] "UCSC"
```

OK, they do. Now we can annotate our SNPs to genes using the `VariantAnnotation`[46] package:

```
library(VariantAnnotation)
snps_anno <- locateVariants(snps, txdb, AllVariants())
snps_anno <- unique(snps_anno)
```

We use the `QUERYID` column in `snps_anno` to recover metadata such as SNP IDs and GWAS *p*-values from the original `snps` object:

```
snps_metadata <- snps[snps_anno$QUERYID]
mcols(snps_anno) <- cbind(mcols(snps_metadata)[c("SNPS", "P-VALUE")],
mcols(snps_anno))
```

We can visualise where these SNPs are located (Figure 6):

```
loc <- data.frame(table(snps_anno$LOCATION))
ggplot(data = loc, aes(x = reorder(Var1, -Freq), y = Freq)) +
  geom_bar(stat = "identity") +
  xlab("Genomic location of SNPs") +
  ylab("Number of SNPs")
```

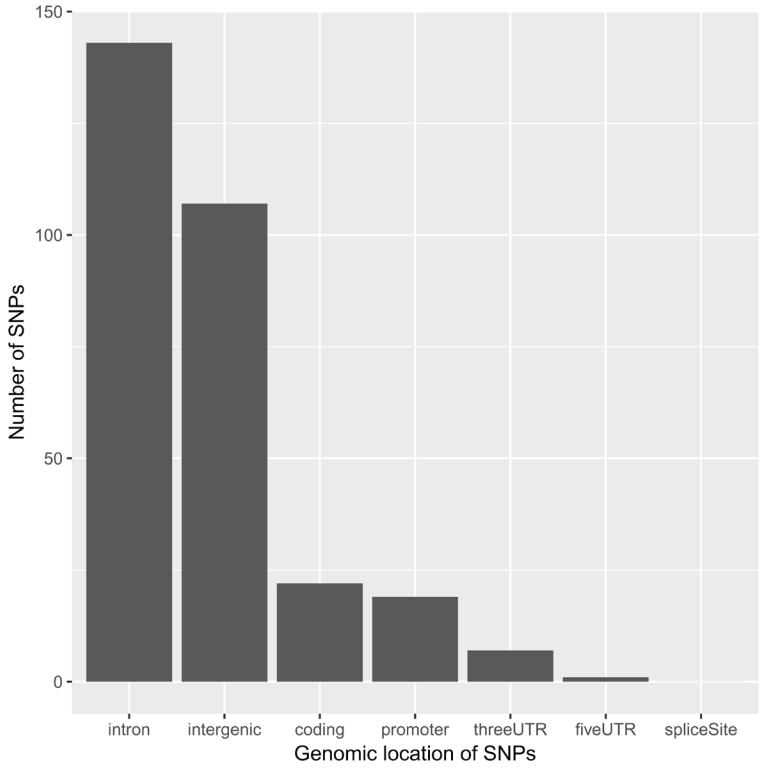

Figure 6. Bar plot showing genomic locations associated with SLE variants.

As expected[7], the great majority of SNPs are located within introns and in intergenic regions. For the moment, we will focus on SNPs that are either coding or in promoter and UTR regions, as these can be assigned to target genes rather unambiguously:

```
snps_easy <- subset(snps_anno, LOCATION == "coding" | LOCATION == "promoter" |
LOCATION == "threeUTR" | LOCATION == "fiveUTR")
snps_easy <- as.data.frame(snps_easy)
```

Now we can check if any of the genes we found to be differentially expressed in SLE is also genetically associated with the disease:

```
snps_easy_in_degs <- merge(degs, snps_easy, by.x = "gene_id", by.y = "GENEID",
all = FALSE)
```

We have 14 genes showing differential expression in SLE that are also genetically associated with the disease. While this is an interesting result, these hits are likely to be already well-known as potential SLE targets given their clear genetic association.

We will store essential information about these hits in a results `data.frame`:

```
prioritised_hits <- unique(data.frame(
  snp_id = snps_easy_in_degs$SNPS,
  snp_pvalue = snps_easy_in_degs$P.VALUE,
  snp_location = snps_easy_in_degs$LOCATION,
  gene_id = snps_easy_in_degs$gene_id,
  gene_symbol = snps_easy_in_degs$symbol,
  gene_pvalue = snps_easy_in_degs$padj,
  gene_log2foldchange = snps_easy_in_degs$log2FoldChange,
  method = "Direct overlap",
  row.names = NULL))
head(prioritised_hits, 3)
##       snp_id snp_pvalue snp_location            gene_id gene_symbol
## 1  rs1887428      1e-06     fiveUTR  ENSG00000096968.13        JAK2
## 2 rs58688157      5e-13     promoter ENSG00000099834.18       CDHR5
## 3  rs1990760      4e-08       coding  ENSG00000115267.5       IFIH1
##    gene_pvalue gene_log2foldchange         method
## 1 1.951160e-04            0.636590 Direct overlap
## 2 1.455662e-05            1.033372 Direct overlap
## 3 2.719420e-10            1.745324 Direct overlap
```

Use of regulatory genomic data to map intronic and intergenic SNPs to target genes

But what about all the SNPs in introns and intergenic regions? Some of those might be regulatory variants affecting the expression level of their target gene(s) through a distal enhancer. Let's create a dataset of candidate regulatory SNPs that are either intronic or intergenic and remove the annotation obtained with `VariantAnnotation`:

```
snps_hard <- subset(snps_anno, LOCATION == "intron" | LOCATION ==
"intergenic", select = c("SNPS", "P.VALUE", "LOCATION"))
```

eQTL data. A well-established way to gain insights into target genes of regulatory SNPs is to use eQTL data, where correlations between genetic variants and expression of genes are computed across different tissues or cell types[9]. Here, we will simply match GWAS SNPs and eQTLs according to their genomic locations, which is a rather crude way to integrate these two types of data. More robust alternatives such as PrediXcan[47], TWAS[48] and SMR[49] exist and should be adopted if possible. One downside of these methods is that they require subject-level or complete summary data, making them less practical in some circumstances.

We will use blood eQTL data from the GTEx consortium[10]. To get the data, you will have to register and download the file `GTEx_Analysis_v7_eQTL.tar.gz` from the GTEx portal to the current working directory:

```
# uncomment the following line to extract the gzipped archive file
#untar("GTEx_Analysis_v7_eQTL.tar.gz")
gtex_blood <-
```

```
read.delim(gzfile("GTEx_Analysis_v7_eQTL/Whole_Blood.v7.signif_variant_gene_
pairs.txt.gz"), stringsAsFactors = FALSE)
head(gtex_blood, 3)
```

```
##           variant_id           gene_id tss_distance ma_samples ma_count
## 1 1_231153_CTT_C_b37 ENSG00000223972.4       219284         13       13
## 2    1_61920_G_A_b37 ENSG00000238009.2       -67303         18       20
## 3    1_64649_A_C_b37 ENSG00000238009.2       -64574         16       16
##         maf pval_nominal    slope slope_se pval_nominal_threshold
## 1 0.0191740  3.69025e-08 1.319720 0.233538            1.35366e-04
## 2 0.0281690  7.00836e-07 0.903786 0.178322            8.26088e-05
## 3 0.0220386  5.72066e-07 1.110040 0.217225            8.26088e-05
##   min_pval_nominal   pval_beta
## 1      3.69025e-08 4.67848e-05
## 2      6.50297e-10 1.11312e-06
## 3      6.50297e-10 1.11312e-06
```

We have to extract the genomic locations of the SNPs from the IDs used by GTEx:

```
locs <- strsplit(gtex_blood$variant_id, "_")
gtex_blood$chr <- sapply(locs, "[", 1)
gtex_blood$start <- sapply(locs, "[", 2)
gtex_blood$end <- sapply(locs, "[", 2)
```

We can then convert the `data.frame` into a `GRanges` object:

```
gtex_blood <- makeGRangesFromDataFrame(gtex_blood, keep.extra.columns = TRUE)
```

We also need to ensure that the chromosome notation is consistent with the previous objects:

```
seqlevelsStyle(gtex_blood)
```

```
## [1] "NCBI"    "Ensembl"
```

```
seqlevelsStyle(gtex_blood) <- "UCSC"
```

From the publication[10], we know the genomic coordinates are mapped to genome reference GRCh37, so we will have to uplift them to GRCh38 using `rtracklayer`[50] and a mapping ("chain") file. The R.utils package is only required to extract the gzipped file:

```
library(rtracklayer)
library(R.utils)
# uncomment the following line to download file
#download.file("http://hgdownload.cse.ucsc.edu/goldenPath/hg19/liftOver/hg19To
Hg38.over.chain.gz", destfile = "hg19ToHg38.over.chain.gz")
# uncomment the following line to extract gzipped file
#gunzip("hg19ToHg38.over.chain.gz")
ch <- import.chain("hg19ToHg38.over.chain")
gtex_blood <- unlist(liftOver(gtex_blood, ch))
```

We will use the `GenomicRanges` package[45] to compute the overlap between GWAS SNPs and blood eQTLs:

```
library(GenomicRanges)
hits <- findOverlaps(snps_hard, gtex_blood)
snps_hard_in_gtex_blood = snps_hard[queryHits(hits)]
gtex_blood_with_snps_hard = gtex_blood[subjectHits(hits)]
mcols(snps_hard_in_gtex_blood) <- cbind(mcols(snps_hard_in_gtex_blood),
mcols(gtex_blood_with_snps_hard))
snps_hard_in_gtex_blood <- as.data.frame(snps_hard_in_gtex_blood)
```

We have 59 blood eQTL variants that are associated with SLE. We can now check whether any of the genes differentially expressed in SLE is an *eGene*, a gene whose expression is influenced by an eQTL. Note that gene IDs in GTEx

are mapped to GENCODE v19[10], while we are using the newer v25 for the DEGs. To match the gene IDs in the two objects, we will simply strip the last bit containing the GENCODE gene version, which effectively gives us Ensembl gene IDs:

```
snps_hard_in_gtex_blood$ensembl_id <- sub("(ENSG[0-9]+)\\.[0-9]+", "\\1",
snps_hard_in_gtex_blood$gene_id)
degs$ensembl_id <- sub("(ENSG[0-9]+)\\.[0-9]+", "\\1", degs$gene_id)
snps_hard_in_gtex_blood_in_degs <- merge(snps_hard_in_gtex_blood, degs, by =
"ensembl_id", all = FALSE)
```

We can add these 17 genes to our list:

```
prioritised_hits <- unique(rbind(prioritised_hits, data.frame(
  snp_id = snps_hard_in_gtex_blood_in_degs$SNPS,
  snp_pvalue = snps_hard_in_gtex_blood_in_degs$P.VALUE,
  snp_location = snps_hard_in_gtex_blood_in_degs$LOCATION,
  gene_id = snps_hard_in_gtex_blood_in_degs$gene_id.y,
  gene_symbol = snps_hard_in_gtex_blood_in_degs$symbol,
  gene_pvalue = snps_hard_in_gtex_blood_in_degs$padj,
  gene_log2foldchange = snps_hard_in_gtex_blood_in_degs$log2FoldChange,
  method = "GTEx eQTLs",
  row.names = NULL)))
```

FANTOM5 data. The FANTOM consortium profiled gene expression across a large panel of tissues and cell types using CAGE[15,17]. This technology allows mapping of transcription start sites and enhancer RNAs genome-wide. Correlations between these promoter and enhancer elements across a large panel of tissues and cell types can then be calculated to identify significant promoter - enhancer pairs. In turn, we will use these correlations to map distal regulatory SNPs to target genes.

Let's read in the enhancer - promoter correlation data:

```
# uncomment the following line to download the file
#download.file("http://enhancer.binf.ku.dk/presets/enhancer_tss_associations.
bed", destfile = "enhancer_tss_associations.bed")
fantom <- read.delim("enhancer_tss_associations.bed", skip = 1,
stringsAsFactors = FALSE)
head(fantom, 3)
```

```
##   X.chrom chromStart chromEnd
## 1    chr1     858252   861621
## 2    chr1     894178   956888
## 3    chr1     901376   956888
##                                                                    name
## 1                    chr1:858256-858648;NM_152486;SAMD11;R:0.404;FDR:0
## 2 chr1:956563-956812;NM_015658;NOC2L;R:0.202;FDR:8.01154668254404e-08
## 3     chr1:956563-956812;NM_001160184,NM_032129;PLEKHN1;R:0.422;FDR:0
##   score strand thickStart thickEnd itemRgb blockCount blockSizes
## 1   404      .     858452   858453   0,0,0          2   401,1001
## 2   202      .     956687   956688   0,0,0          2   1001,401
## 3   422      .     956687   956688   0,0,0          2   1001,401
##   chromStarts
## 1      0,2368
## 2     0,62309
## 3     0,55111
```

Everything we need is in the fourth column, `name`: genomic location of the enhancer, gene identifiers, Pearson correlation coefficient and significance. We will use the splitstackshape package to parse it:

```
library(splitstackshape)
fantom <- as.data.frame(cSplit(fantom, splitCols = "name", sep = ";",
direction = "wide"))
```

Now we can extract the genomic locations of the enhancers and the correlation values:

```
locs <- strsplit(as.character(fantom$name_1), "[:-]")
fantom$chr <- sapply(locs, "[", 1)
fantom$start <- as.numeric(sapply(locs, "[", 2))
fantom$end <- as.numeric(sapply(locs, "[", 3))
fantom$symbol <- fantom$name_3
fantom$corr <- sub("R:", "", fantom$name_4)
fantom$fdr <- sub("FDR:", "", fantom$name_5)
```

We can select only the enhancer - promoter pairs with a decent level of correlation and significance and tidy the data at the same time:

```
fantom <- unique(subset(fantom, corr >= 0.25 & fdr < 1e-5, select = c("chr",
"start", "end", "symbol")))
```

Now we would like to check whether any of our candidate regulatory SNPs are falling in any of these enhancers. To do this, we have to convert the `data.frame` into a `GRanges` object and uplift the GRCh37 coordinates[15] to GRCh38:

```
fantom <- makeGRangesFromDataFrame(fantom, keep.extra.columns = TRUE)
fantom <- unlist(liftOver(fantom, ch))
```

We can now compute the overlap between SNPs and enhancers:

```
hits <- findOverlaps(snps_hard, fantom)
snps_hard_in_fantom = snps_hard[queryHits(hits)]
fantom_with_snps_hard = fantom[subjectHits(hits)]
mcols(snps_hard_in_fantom) <- cbind(mcols(snps_hard_in_fantom),
mcols(fantom_with_snps_hard))
snps_hard_in_fantom <- as.data.frame(snps_hard_in_fantom)
```

Let's check if any of these genes is differentially expressed in our RNA-seq data:

```
snps_hard_in_fantom_in_degs <- merge(snps_hard_in_fantom, degs, by = "symbol",
all = FALSE)
```

We have identified 7 genes whose putative enhancers contain SLE GWAS SNPs. Let's add these to our list:

```
prioritised_hits <- unique(rbind(prioritised_hits, data.frame(
  snp_id = snps_hard_in_fantom_in_degs$SNPS,
  snp_pvalue = snps_hard_in_fantom_in_degs$P.VALUE,
  snp_location = snps_hard_in_fantom_in_degs$LOCATION,
  gene_id = snps_hard_in_fantom_in_degs$gene_id,
  gene_symbol = snps_hard_in_fantom_in_degs$symbol,
  gene_pvalue = snps_hard_in_fantom_in_degs$padj,
  gene_log2foldchange = snps_hard_in_fantom_in_degs$log2FoldChange,
  method = "FANTOM5 correlations",
  row.names = NULL)))
```

Promoter Capture Hi-C data. More recently, chromatin interaction data was generated across 17 human primary blood cell types using promoter capture Hi-C[21]. More than 30,000 promoter baits were used to capture promoter-interacting regions genome-wide, which were then mapped to enhancers based on annotation present in the Ensembl Regulatory Build[51]. This dataset provides a valuable resource for interpreting complex genomic data, especially in the context of autoimmune diseases (and other conditions where immune cells play a role). Significant interactions between enhancers and promoters can be accessed in the supplementary data of the paper:

```
# uncomment the following line to download file
#download.file("http://www.cell.com/cms/attachment/2086554122/2074217047/mmc4.
zip", destfile = "mmc4.zip")
# uncomment the following lines to extract zipped files
```

```
#unzip("mmc4.zip")
#unzip("DATA_S1.zip")
pchic <- read.delim("ActivePromoterEnhancerLinks.tsv", stringsAsFactors =
FALSE)
head(pchic, 3)
```

```
##   baitChr  baitSt baitEnd baitID oeChr    oeSt   oeEnd oeID
## 1    chr1 1206873 1212438    254  chr1  943676  957199  228
## 2    chr1 1206873 1212438    254  chr1 1034268 1040208  235
## 3    chr1 1206873 1212438    254  chr1 1040208 1043143  236
##                       cellType.s.
## 1                            nCD8
## 2 nCD4,nCD8,Mac0,Mac1,Mac2,MK,Mon
## 3     nCD4,nCD8,Mac0,Mac1,Mac2,MK
##
sample.s.
## 1
C0066PH1
## 2
S007DDH2,S007G7H4,C0066PH1,S00C2FH1,S00390H1,S001MJH1,S001S7H2,S0022IH2,S00622
H1,S00BS4H1,S004BTH2,C000S5H2
## 3
S007DDH2,S007G7H4,C0066PH1,S00C2FH1,S00390H1,S001MJH1,S001S7H2,S0022IH2,S00622
H1,S00BS4H1,S004BTH2
```

We will use the `InteractionSet` package[52], which is specifically designed for the representation of chromatin interaction data. We start by creating a `GInteractions` object:

```
library(InteractionSet)
promoters <- GRanges(seqnames = pchic$baitChr, ranges = IRanges(start =
pchic$baitSt, end = pchic$baitEnd))
enhancers <- GRanges(seqnames = pchic$oeChr, ranges = IRanges(start =
pchic$oeSt, end = pchic$oeEnd))
pchic <- GInteractions(promoters, enhancers)
```

As gene identifiers are not provided, we also have to map promoters to the respective genes so that we know which genes are regulated by which enhancers. We can do this by using the `TxDb` object we previously built to extract positions of transcription start sites (TSSs) and then add the GENCODE gene IDs as metadata to the `pchic` object:

```
tsss <- promoters(txdb, upstream = 0, downstream = 1, columns = "gene_id")
hits <- nearest(promoters, tsss)
pchic$gene_id <- unlist(tsss[hits]$gene_id)
```

Next, we calculate the overlaps between SLE GWAS SNPs and enhancers (the *second* region of the `GInteractions` object) :

```
hits <- findOverlaps(snps_hard, pchic, use.region = "second")
snps_hard_in_pchic = snps_hard[queryHits(hits)]
pchic_with_snps_hard = pchic[subjectHits(hits)]
mcols(snps_hard_in_pchic) <- cbind(mcols(snps_hard_in_pchic),
mcols(pchic_with_snps_hard))
snps_hard_in_pchic <- as.data.frame(snps_hard_in_pchic)
```

We check if any of these enhancers containing SLE variants are known to putatively regulate genes differentially expressed in SLE:

```
snps_hard_in_pchic_in_degs <- merge(snps_hard_in_pchic, degs, by = "gene_id",
all = FALSE)
```

And finally we add these 13 genes to our list:

```
prioritised_hits <- unique(rbind(prioritised_hits, data.frame(
  snp_id = snps_hard_in_pchic_in_degs$SNPS,
  snp_pvalue = snps_hard_in_pchic_in_degs$P.VALUE,
  snp_location = snps_hard_in_pchic_in_degs$LOCATION,
  gene_id = snps_hard_in_pchic_in_degs$gene_id,
  gene_symbol = snps_hard_in_pchic_in_degs$symbol,
  gene_pvalue = snps_hard_in_pchic_in_degs$padj,
  gene_log2foldchange = snps_hard_in_pchic_in_degs$log2FoldChange,
  method = "Promoter capture Hi-C",
  row.names = NULL)))
```

These are the final results of our target identification exercise. We can have a look at the most significant SNPs mapped with each of the methods:

```
top_prioritised_hits <- prioritised_hits[order(prioritised_hits$snp_pvalue),]
top_prioritised_hits <- split(top_prioritised_hits,
top_prioritised_hits$method)
do.call(rbind, lapply(top_prioritised_hits, head, 1))
```

```
##                          snp_id snp_pvalue snp_location            gene_id
## Direct overlap        rs3757387      1e-48     promoter ENSG00000128604.18
## GTEx eQTLs            rs1270942     2e-165       intron ENSG00000166278.14
## FANTOM5 correlations  rs1150754      6e-29       intron  ENSG00000204421.2
## Promoter capture Hi-C rs1270942     2e-165       intron  ENSG00000219797.2
##                       gene_symbol  gene_pvalue gene_log2foldchange
## Direct overlap               IRF5 5.006707e-03           0.4041349
## GTEx eQTLs                     C2 1.625111e-03           0.9269526
## FANTOM5 correlations       LY6G6C 3.575357e-05           1.4327915
## Promoter capture Hi-C          NA 1.919459e-04           0.4556364
##                                      method
## Direct overlap               Direct overlap
## GTEx eQTLs                      GTEx eQTLs
## FANTOM5 correlations   FANTOM5 correlations
## Promoter capture Hi-C Promoter capture Hi-C
```

We can also visualise the relative contributions from the different approaches we used (Figure 7):

```
prioritised_genes <- unique(data.frame(gene_id = prioritised_hits$gene_id,
method = prioritised_hits$method))
ggplot(data = prioritised_genes, aes(x = method)) +
  geom_bar(aes(fill = method), stat = "count") +
  ylab("Number of genes") +
  theme(axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank())
```

We observe that all methods significantly contributed to the identification of genes associated with GWAS SNPs. The majority of genes were identified through the integration of the GTEx blood eQTL data, followed by the methods based on direct overlap, promoter capture Hi-C data and FANTOM5 correlations.

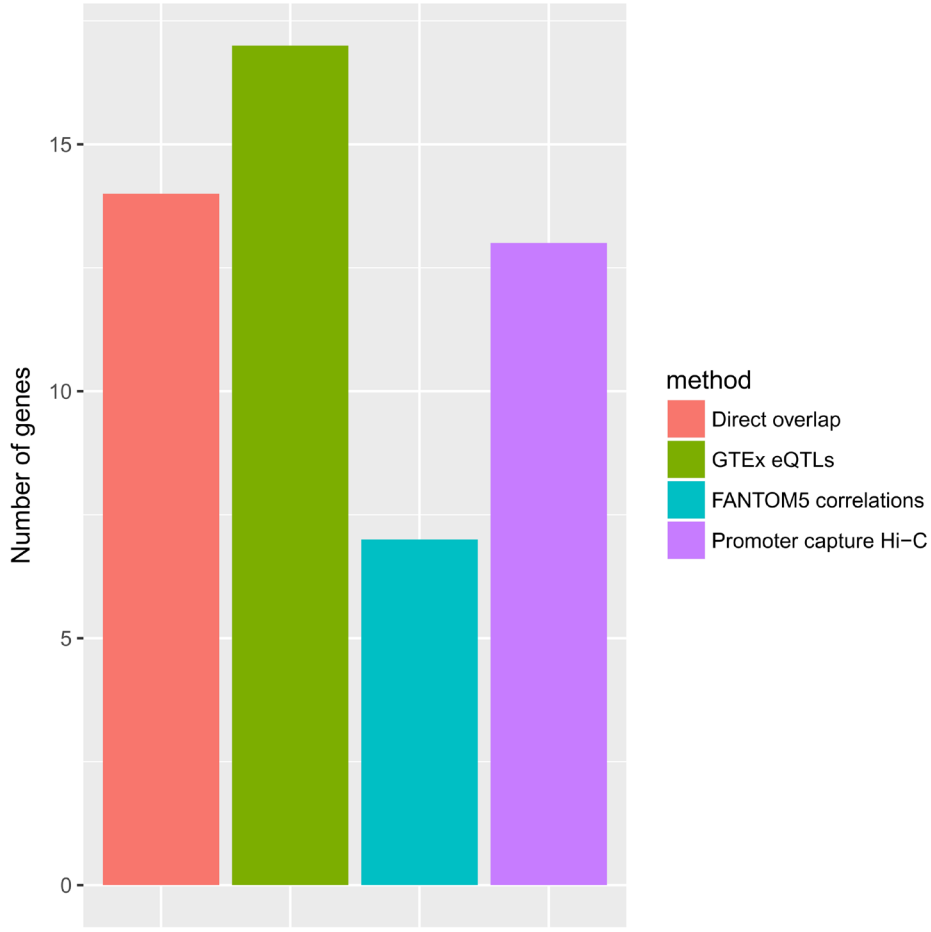

Figure 7. Bar plot showing number of genes identified by each variant mapping method.

Functional analysis of prioritised hits

We will use biological processes from the Gene Ontology[53] and the clusterProfiler package[54] to functionally characterise our list of genes:

```
library(clusterProfiler)
prioritised_hits_ensembl_ids <- unique(sub("(ENSG[0-9]+)\\.[0-9]+", "\\1",
prioritised_hits$gene_id))
all_genes_ensembl_ids <- unique(sub("(ENSG[0-9]+)\\.[0-9]+", "\\1",
rownames(rse)))
gobp_enrichment <- enrichGO(prioritised_hits_ensembl_ids,
                            universe = all_genes_ensembl_ids,
                            OrgDb = org.Hs.eg.db,
                            keyType = "ENSEMBL",
                            ont = "BP",
                            pAdjustMethod = "BH",
                            pvalueCutoff = 0.05,
                            qvalueCutoff = 0.05,
                            readable = TRUE)
```

We can visualise the most enriched terms (Figure 8):

```
dotplot(gobp_enrichment, showCategory = 20)
```

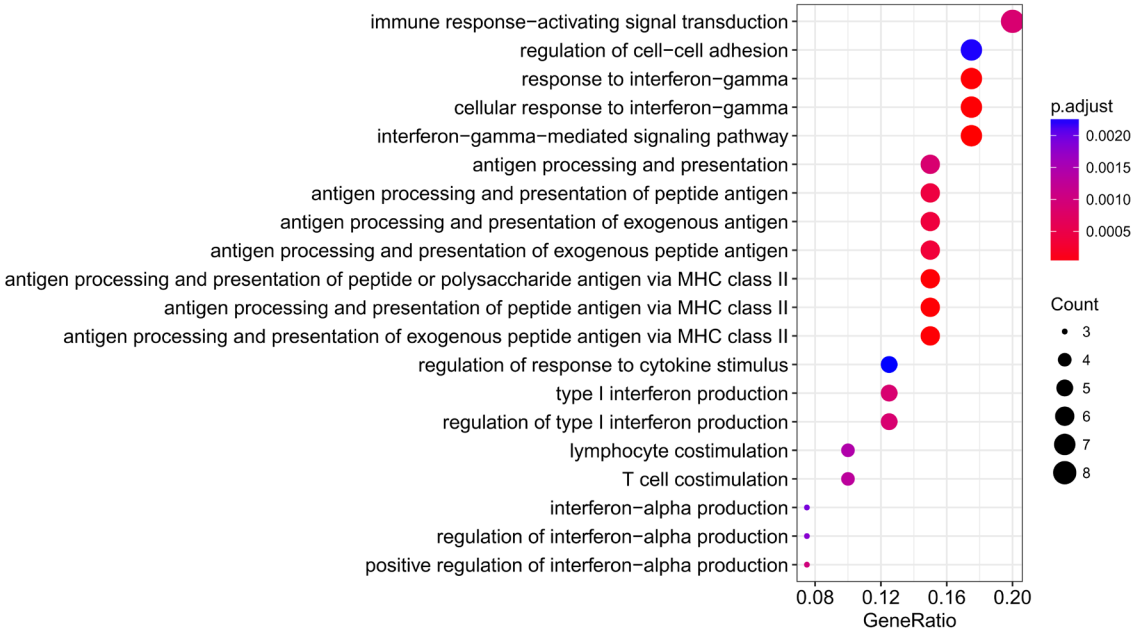

Figure 8. Dot plot showing enrichment of Gene Ontology biological processes for the list of prioritised genes.

We observe a significant enrichment for interferon responses, antigen processing and presentation, and T cell stimulation, all processes which are well-known to play key roles in the pathogenesis of SLE[55–57].

From a drug discovery perspective, JAK2 is probably the most attractive target: rs1887428 (p-value = 1×10^{-6} is located in its 5' UTR and the genes is significantly upregulated in disease. Tofacitinib, a pan-JAK inhibitor, showed promising results in mouse[58] and is currently being tested or safety in a phase I clinical trial. We find 7 GWAS SNPs that are blood eQTLs linked to the expression of C2, a protease active in the complement signalling cascade. The most significant variant is rs1270942 (p-value = 2×10^{-165}) and is found in an intron of CFB, another component of the complement system. As with other autoimmune diseases, the complement plays a key role SLE in and has been investigated as a therapeutic approach[59]. Another potentially interesting hit is TAX1BP1: rs849142 (p-value = 1×9^{-11}) is found within an intron of JAZF1, but can be linked to TAX1BP1 via a chromatin interaction with its promoter. TAX1BP1 inhibits TNF-induced apoptosis[60] and is involved in the IL1 signalling cascade[61], another relevant pathway in SLE that could be therapeutically targeted[62].

Conclusions

In this Bioconductor workflow we have used several packages and datasets to demonstrate how regulatory genomic data can be used to annotate significant hits from GWASs and prioritise gene lists from expression studies, providing an intermediate layer connecting genetics and transcriptomics. Overall, we identified 46 SLE-associated SNPs that we mapped to 49 genes differentially expressed in SLE, using eQTL data[10] and enhancer - promoter relationships from CAGE[15] and promoter capture Hi-C experiments[21]. These genes are involved in key inflammatory signalling pathways and some of them could develop into therapeutic targets for SLE.

The workflow also demonstrates some real-world challenges encountered when working with genomic data from different sources, such as the use of different genome assemblies and gene annotation systems, the parsing of files with custom formats into Bioconductor objects and the mapping of genomic locations to genes. While options for the visualisations of genomic data and interactions are outside the scope of this workflow, at least three good alternatives exist in Bioconductor: ggbio[63], Sushi[64] and Gviz[65] coupled with the GenomicInteractions package[66]. We refer the reader to these publications and package vignettes for examples.

As the sample size and power of GWASs and gene expression studies continue to increase, it will become more and more challenging to identify truly significant hits and interpret them. The use of regulatory genomics data as presented here can be an important tool to gain insights into large biomedical datasets and help in the identification of biomarkers and therapeutic targets.

Data and software availability

Download links for all datasets are part of the workflow. Software packages required to reproduce the analysis can be installed as part of the workflow. Source code is available at: https://github.com/enricoferrero/bioconductor-regulatory-genomics-workflow. Archived source code as at the time of publication is available at: https://doi.org/10.5281/zenodo.1154124[67].

Competing interests

EF is a full time employee of GSK.

Grant information

The author(s) declared that no grants were involved in supporting this work.

References

1. Waring MJ, Arrowsmith J, Leach AR, *et al.*: **An analysis of the attrition of drug candidates from four major pharmaceutical companies.** *Nat Rev Drug Discov.* 2015; **14**(7): 475–86.
PubMed Abstract | Publisher Full Text
2. DiMasi JA, Grabowski HG, Hansen RW: **Innovation in the pharmaceutical industry: New estimates of R&D costs.** *J Health Econ.* 2016; **47**: 20–33.
PubMed Abstract | Publisher Full Text
3. Harrison RK: **Phase II and phase III failures: 2013–2015.** *Nat Rev Drug Discov.* 2016; **15**(12): 817–8.
PubMed Abstract | Publisher Full Text
4. Cook D, Brown D, Alexander R, *et al.*: **Lessons learned from the fate of AstraZeneca's drug pipeline: a five-dimensional framework.** *Nat Rev Drug Discov.* 2014; **13**(6): 419–31.
PubMed Abstract | Publisher Full Text
5. Plenge RM, Scolnick EM, Altshuler D: **Validating therapeutic targets through human genetics.** *Nat Rev Drug Discov.* 2013; **12**(8): 581–94.
PubMed Abstract | Publisher Full Text
6. Nelson MR, Tipney H, Painter JL, *et al.*: **The support of human genetic evidence for approved drug indications.** *Nat Genet.* 2015; **47**(8): 856–60.
PubMed Abstract | Publisher Full Text
7. Maurano MT, Humbert R, Rynes E, *et al.*: **Systematic localization of common disease-associated variation in regulatory DNA.** *Science.* 2012; **337**(6099): 1190–5.
PubMed Abstract | Publisher Full Text | Free Full Text
8. Ward LD, Kellis M: **Interpreting noncoding genetic variation in complex traits and human disease.** *Nat Biotechnol.* 2012; **30**(11): 1095–106.
PubMed Abstract | Publisher Full Text | Free Full Text
9. Albert FW, Kruglyak L: **The role of regulatory variation in complex traits and disease.** *Nat Rev Genet.* 2015; **16**(4): 197–212.
PubMed Abstract | Publisher Full Text
10. GTEx Consortium, Laboratory, Data Analysis & Coordinating Center (LDACC)—Analysis Working Group, Statistical Methods groups—Analysis Working Group, *et al.*: **Genetic effects on gene expression across human tissues.** *Nature.* 2017; **550**(7675): 204–13.
PubMed Abstract | Publisher Full Text | Free Full Text
11. ENCODE Project Consortium: **An integrated encyclopedia of DNA elements in the human genome.** *Nature.* 2012; **489**(7414): 57–74.
PubMed Abstract | Publisher Full Text | Free Full Text
12. Roadmap Epigenomics Consortium, Kundaje A, Meuleman W, *et al.*: **Integrative analysis of 111 reference human epigenomes.** *Nature.* 2015; **518**(7539): 317–30.
PubMed Abstract | Publisher Full Text | Free Full Text
13. Adams D, Altucci L, Antonarakis SE, *et al.*: **BLUEPRINT to decode the epigenetic signature written in blood.** *Nat Biotechnol.* 2012; **30**(3): 224–6.
PubMed Abstract | Publisher Full Text
14. Stunnenberg HG; International Human Epigenome Consortium, Hirst M: **The International Human Epigenome Consortium: A Blueprint for Scientific Collaboration and Discovery.** *Cell.* 2016; **167**(7): 1897.
PubMed Abstract | Publisher Full Text
15. FANTOM Consortium and the RIKEN PMI and CLST (DGT), Forrest AR, Kawaji H, *et al.*: **A promoter-level mammalian expression atlas.** *Nature.* 2014; **507**(7493): 462–70.
PubMed Abstract | Publisher Full Text | Free Full Text
16. Thurman RE, Rynes E, Humbert R, *et al.*: **The accessible chromatin landscape of the human genome.** *Nature.* 2012; **489**(7414): 75–82.
PubMed Abstract | Publisher Full Text | Free Full Text
17. Andersson R, Gebhard C, Miguel-Escalada I, *et al.*: **An atlas of active enhancers across human cell types and tissues.** *Nature.* 2014; **507**(7493): 455–61.
PubMed Abstract | Publisher Full Text | Free Full Text
18. Fullwood MJ, Wei CL, Liu ET, *et al.*: **Next-generation DNA sequencing of paired-end tags (PET) for transcriptome and genome analyses.** *Genome Res.* 2009; **19**(4): 521–32.
PubMed Abstract | Publisher Full Text | Free Full Text
19. Zhang Y, Wong CH, Birnbaum RY, *et al.*: **Chromatin connectivity maps reveal dynamic promoter-enhancer long-range associations.** *Nature.* 2013; **504**(7479): 306–10.
PubMed Abstract | Publisher Full Text | Free Full Text
20. Mifsud B, Tavares-Cadete F, Young AN, *et al.*: **Mapping long-range promoter contacts in human cells with high-resolution capture Hi-C.** *Nat Genet.* 2015; **47**(6): 598–606.
PubMed Abstract | Publisher Full Text
21. Javierre BM, Burren OS, Wilder SP, *et al.*: **Lineage-Specific Genome Architecture Links Enhancers and Non-coding Disease Variants to Target Gene Promoters.** *Cell.* 2016; **167**(5): 1369–1384.e19.
PubMed Abstract | Publisher Full Text | Free Full Text
22. Shen J, Song K, Slater AJ, *et al.*: **STOPGAP: a database for systematic target opportunity assessment by genetic association predictions.** *Bioinformatics.* 2017; **33**(17): 2784–6.
PubMed Abstract | Publisher Full Text
23. Amlie-Wolf A, Tang M, Mlynarski EE, *et al.*: **INFERNO - INFERring the molecular mechanisms of NOncoding genetic variants.** *bioRxiv.* 2017; 211599.
Publisher Full Text
24. Hung T, Pratt GA, Sundararaman B, *et al.*: **The Ro60 autoantigen binds endogenous retroelements and regulates inflammatory gene expression.** *Science.* 2015; **350**(6259): 455–9.
PubMed Abstract | Publisher Full Text | Free Full Text
25. Kaul A, Gordon C, Crow MK, *et al.*: **Systemic lupus erythematosus.** *Nat Rev Dis Primers.* 2016; **2**: 16039.
PubMed Abstract | Publisher Full Text
26. Marion TN, Postlethwaite AE: **Chance, genetics, and the heterogeneity of disease and pathogenesis in systemic lupus erythematosus.** *Semin Immunopathol.* 2014; **36**(5): 495–517.
PubMed Abstract | Publisher Full Text
27. Amezcua-Guerra LM, Higuera-Ortiz V, Arteaga-García U, *et al.*: **Performance of the 2012 Systemic Lupus International Collaborating Clinics and the 1997 American College of Rheumatology classification criteria for systemic lupus erythematosus in a real-life scenario.** *Arthritis Care Res (Hoboken).* 2015; **67**(3): 437–41.
PubMed Abstract | Publisher Full Text
28. Collado-Torres L, Nellore A, Kammers K, *et al.*: **Reproducible RNA-seq**

analysis using *recount2*. *Nat Biotechnol.* 2017; **35**(4): 319–21.
PubMed Abstract | Publisher Full Text

29. Davis S, Meltzer PS: **GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor.** *Bioinformatics.* 2007; **23**(14): 1846–7.
PubMed Abstract | Publisher Full Text

30. Kauffmann A, Rayner TF, Parkinson H, *et al.*: **Importing ArrayExpress datasets into R/Bioconductor.** *Bioinformatics.* 2009; **25**(16): 2092–4.
PubMed Abstract | Publisher Full Text | Free Full Text

31. Harrow J, Frankish A, Gonzalez JM, *et al.*: **GENCODE: the reference human genome annotation for The ENCODE Project.** *Genome Res.* 2012; **22**(9): 1760–74.
PubMed Abstract | Publisher Full Text | Free Full Text

32. Love MI, Huber W, Anders S: **Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2.** *Genome Biol.* 2014; **15**(12): 550.
PubMed Abstract | Publisher Full Text | Free Full Text

33. Robinson MD, McCarthy DJ, Smyth GK: **edgeR: a Bioconductor package for differential expression analysis of digital gene expression data.** *Bioinformatics.* 2010; **26**(1): 139–40.
PubMed Abstract | Publisher Full Text | Free Full Text

34. Ritchie ME, Phipson B, Wu D, *et al.*: ***limma* powers differential expression analyses for RNA-sequencing and microarray studies.** *Nucleic Acids Res.* 2015; **43**(7): e47.
PubMed Abstract | Publisher Full Text | Free Full Text

35. Anders S, Huber W: **Differential expression analysis for sequence count data.** *Genome Biol.* 2010; **11**(10): R106.
PubMed Abstract | Publisher Full Text | Free Full Text

36. Wickham H: **Ggplot2.** New York, NY: Springer New York; 2009.
Publisher Full Text

37. Chen R, Morgan AA, Dudley J, *et al.*: **FitSNPs: highly differentially expressed genes are more likely to have variants associated with disease.** *Genome Biol.* 2008; **9**(12): R170.
PubMed Abstract | Publisher Full Text | Free Full Text

38. Carey VJ: **Gwascat.** 2017.
Publisher Full Text

39. MacArthur J, Bowler E, Cerezo M, *et al.*: **The new NHGRI-EBI Catalog of published genome-wide association studies (GWAS Catalog).** *Nucleic Acids Res.* 2017; **45**(D1): D896–901.
PubMed Abstract | Publisher Full Text | Free Full Text

40. Eicher JD, Landowski C, Stackhouse B, *et al.*: **GRASP v2.0: an update on the Genome-Wide Repository of Associations between SNPs and phenotypes.** *Nucleic Acids Res.* 2015; **43**(Database issue): D799–804.
PubMed Abstract | Publisher Full Text | Free Full Text

41. Carey VJ: **Grasp2db.** 2007.
Publisher Full Text

42. Bush WS, Moore JH: **Chapter 11: Genome-wide association studies.** *PLoS Comput Biol.* 2012; **8**(12): e1002822.
PubMed Abstract | Publisher Full Text | Free Full Text

43. Carey VJ: **Ldblock.** 2017.
Publisher Full Text

44. Yates A, Beal K, Keenan S, *et al.*: **The Ensembl REST API: Ensembl Data for Any Language.** *Bioinformatics.* 2015; **31**(1): 143–5.
PubMed Abstract | Publisher Full Text | Free Full Text

45. Lawrence M, Huber W, Pagès H, *et al.*: **Software for computing and annotating genomic ranges.** *PLoS Comput Biol.* 2013; **9**(8): e1003118.
PubMed Abstract | Publisher Full Text | Free Full Text

46. Obenchain V, Lawrence M, Carey V, *et al.*: **VariantAnnotation: a Bioconductor package for exploration and annotation of genetic variants.** *Bioinformatics.* 2014; **30**(14): 2076–8.
PubMed Abstract | Publisher Full Text | Free Full Text

47. Gamazon ER, Wheeler HE, Shah KP, *et al.*: **A gene-based association method for mapping traits using reference transcriptome data.** *Nat Genet.* 2015; **47**(9): 1091–8.
PubMed Abstract | Publisher Full Text | Free Full Text

48. Gusev A, Ko A, Shi H, *et al.*: **Integrative approaches for large-scale transcriptome-wide association studies.** *Nat Genet.* 2016; **48**(3): 245–52.
PubMed Abstract | Publisher Full Text | Free Full Text

49. Zhu Z, Zhang F, Hu H, *et al.*: **Integration of summary data from GWAS and eQTL studies predicts complex trait gene targets.** *Nat Genet.* 2016; **48**(5): 481–7.
PubMed Abstract | Publisher Full Text

50. Lawrence M, Gentleman R, Carey V: **rtracklayer: an R package for interfacing with genome browsers.** *Bioinformatics.* 2009; **25**(14): 1841–2.
PubMed Abstract | Publisher Full Text | Free Full Text

51. Zerbino DR, Wilder SP, Johnson N, *et al.*: **The ensembl regulatory build.** *Genome Biol.* 2015; **16**(1): 56.
PubMed Abstract | Publisher Full Text | Free Full Text

52. Lun AT, Perry M, Ing-Simmons E: **Infrastructure for genomic interactions: Bioconductor classes for Hi-C, ChIA-PET and related experiments [version 2; referees: 2 approved].** *F1000Res.* 2016; **5**: 950.
PubMed Abstract | Publisher Full Text | Free Full Text

53. Ashburner M, Ball CA, Blake JA, *et al.*: **Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.** *Nat Genet.* 2000; **25**(1): 25–9.
PubMed Abstract | Publisher Full Text | Free Full Text

54. Yu G, Wang LG, Han Y, *et al.*: **clusterProfiler: an R package for comparing biological themes among gene clusters.** *OMICS.* 2012; **16**(5): 284–7.
PubMed Abstract | Publisher Full Text | Free Full Text

55. Oon S, Wilson NJ, Wicks I: **Targeted therapeutics in SLE: emerging strategies to modulate the interferon pathway.** *Clin Transl Immunology.* 2016; **5**(5): e79.
PubMed Abstract | Publisher Full Text | Free Full Text

56. Morris DL, Fernando MM, Taylor KE, *et al.*: **MHC associations with clinical and autoantibody manifestations in European SLE.** *Genes Immun.* 2014; **15**(4): 210–7.
PubMed Abstract | Publisher Full Text | Free Full Text

57. Suárez-Fueyo A, Bradley SJ, Tsokos GC: **T cells in Systemic Lupus Erythematosus.** *Curr Opin Immunol.* 2016; **43**: 32–8.
PubMed Abstract | Publisher Full Text | Free Full Text

58. Furumoto Y, Smith CK, Blanco L, *et al.*: **Tofacitinib Ameliorates Murine Lupus and Its Associated Vascular Dysfunction.** *Arthritis Rheumatol.* 2017; **69**(1): 148–60.
PubMed Abstract | Publisher Full Text | Free Full Text

59. Leffler J, Bengtsson AA, Blom AM: **The complement system in systemic lupus erythematosus: an update.** *Ann Rheum Dis.* 2014; **73**(9): 1601–6.
PubMed Abstract | Publisher Full Text

60. De Valck D, Jin DY, Heyninck K, *et al.*: **The zinc finger protein A20 interacts with a novel anti-apoptotic protein which is cleaved by specific caspases.** *Oncogene.* 1999; **18**(29): 4182–90.
PubMed Abstract | Publisher Full Text

61. Ling L, Goeddel DV: **T6BP, a TRAF6-interacting protein involved in IL-1 signaling.** *Proc Natl Acad Sci U S A.* 2000; **97**(17): 9567–72.
PubMed Abstract | Publisher Full Text | Free Full Text

62. Rönnblom L, Elkon KB: **Cytokines as therapeutic targets in SLE.** *Nat Rev Rheumatol.* 2010; **6**(6): 339–47.
PubMed Abstract | Publisher Full Text

63. Yin T, Cook D, Lawrence M: **ggbio: an R package for extending the grammar of graphics for genomic data.** *Genome biol.* 2012; **13**(8): R77.
PubMed Abstract | Publisher Full Text | Free Full Text

64. Phanstiel DH, Boyle AP, Araya CL, *et al.*: **Sushi.R: flexible, quantitative and integrative genomic visualizations for publication-quality multi-panel figures.** *Bioinformatics.* 2014; **30**(19): 2808–10.
PubMed Abstract | Publisher Full Text | Free Full Text

65. Hahne F, Ivanek R: **Visualizing Genomic Data Using Gviz and Bioconductor.** *Methods Mol Biol.* 2016; **1418**: 335–51.
PubMed Abstract | Publisher Full Text

66. Harmston N, Ing-Simmons E, Perry M, *et al.*: **GenomicInteractions: An R/Bioconductor package for manipulating and investigating chromatin interaction data.** *BMC genomics.* 2015; **16**: 963.
PubMed Abstract | Publisher Full Text | Free Full Text

67. Ferrero E: **enricoferrero/bioconductor-regulatory-genomics-workflow: Version 2 (Version v2.2).** *Zenodo.* 2018.
Data Source

Open Peer Review

Current Referee Status: ✔ ✔

Version 2

Referee Report 09 March 2018

doi:10.5256/f1000research.15281.r31142

✔ **Vincent J. Carey**
Departments of Medicine, Brigham and Women's Hospital and Harvard Medical School, Boston, MA, USA

Competing Interests: No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Referee Report 05 March 2018

doi:10.5256/f1000research.15281.r31141

✔ **Aaron T. L. Lun**
CRUK (Cancer Research UK) Cambridge Institute, University of Cambridge, Cambridge, UK

- There are some sentences where the text says "7 genes", or "13 genes".
While this is fine, it is good practice to print "nrow(prioritised_hits)" or similar to show to readers that, yes, there are indeed the stated number of genes.

- My suggestion for "linkOverlaps" was to do something like:

linked <- linkOverlaps(pchic, snps_hard, tsss, use.region="same")

... which would give you the physical interactions between the SNPs (linked$subject1) and the TSS's (linked$subject2). I only mention this for future reference, as this may be more convenient in some settings.

Competing Interests: No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Version 1

Referee Report 07 February 2018

doi:10.5256/f1000research.14748.r30353

? **Vincent J. Carey**
Departments of Medicine, Brigham and Women's Hospital and Harvard Medical School, Boston, MA, USA

I want to preface this long review with some very broad comments. I think this undertaking is very worthwhile from several perspectives. Bioconductor is used along various avenues to create a unifiable analytic process from very diverse data resources: state-of-the-art transcriptomics from recount, **current** GWAS catalog from EMBL/EBI, variant annotation for SLE GWAS hits from the eponymous package using GENCODE for gene models, eQTL data from GTEx, enhancer annotation from FANTOM5, and promoter capture data whose origins could be better described. This is a *tour de force* but I feel it should be communicated more clearly and executed more cleanly. The paper is full of "dumps" of show events for R objects that impede the narrative flow drastically. A diagram that shows how the various resources combine in a scientifically coherent way would be a huge step forward for the paper and for practitioners. More reckoning of limitations that arise from complexity is also in order. eQTLs are far from simple, and should not be used as 'lists'. Enhancer and promoter 'lists' also need to be used with care.

What then about this paper? It shows the resources and it shows **a path**. Isn't that enough? I don't think so. If Bioconductor and online publication make it **easier** to do and to publish complex analyses, then the presentation should be of at least as high a quality as we find in articles that are behind paywalls. In this case I feel the quality would be improved through condensation. The object dumps should be removed and replaced by meaningful tabulations and diagrams. The big picture should be stated more clearly and concisely. The limitations should also be discussed clearly. I would love to see a small set of functions that carry out the salient operations chained together to produce the solution.
Then, given the programmatic compactness, we can discuss how to evaluate the robustness of the results of the analysis by carrying out **sensitivity analysis**. In particular, it would be great to see how the different elements of the system contribute to the ultimate enumeration of targets.

The premise of this article is that "therapeutic targets with a genetic link to the disease under investigation are more likely to progress through the drug discovery pipeline". GWAS, PheWAS, eQTLs, epigenomic roadmap projects, and other general studies of gene regulation should be harvested to improve capacity to define genetic and genomic origins of disease, with an aim to fostering design of treatments that are focused on the molecular events underlying the disease process. The introduction concludes with mention of STOPGAP, and POSTGAP, and INFERNO, but it is not clear whether the paper is intended to describe how content of STOPGAP is developed from basic data resources like those readily available to Bioconductor users. I feel that the introduction, though well-referenced, is too long and does not clearly state the paper's main goal.

There is no discussion of the experimental design underlying the RNA-seq study. Presumably the data were generated from this component of ref 28:

"Finally, we tested the levels of Alu transcripts in blood cells of SLE patients and controls(22) using RNAseq (99 active SLE, 18 healthy controls; Fig. S12). RNA-seq reads mapping to Alu elements were

found at significantly higher levels in SLE subjects than controls (p=6.5E-6), Fig. 4E). Hierarchical clustering of the most highly expressed Alu RNAs (Fig. S13) segregated Interferon Signature Metric (ISM)-high SLE subjects from control and ISM-low patients"....

There is no discussion of heterogeneity of SLE or the difficulty of learning from a collection of 18 cases. A reference to https://www.ncbi.nlm.nih.gov/pubmed/25102991 may be in order.

Even though online publications are often free from page count limitations, entirely too much space is consumed by long row-broken R print events. On the one hand the recoding of SRA annotation on phenotype is important and should be exposed, on the other hand, the author could carry out the recoding programmatically in a well-parameterized function and simply update the key object by applying this function. The function can go in a package related to the paper/workflow. Instead of printing out a dataframe on p.7, it would be much better to have a contingency table showing the final layout of case and control characteristics.

p.7 "For simplicity, we select the first 18 (healthy) and the last 18 (SLE) samples from the original RangedSummarizedExperiment object". Is this essential to the performance of the workflow? Would a more systematic matching be possible? What kind of "simplicity" does this arbitrary selection create? I understand that the main purpose of the paper is to illustrate a process, but if this thinning of the data is not essential to the illustration, why do it?

p. 8: "Note that we used an extremely simple model; in the real world you will probably need to account for co-variables, potential confounders and interactions between them. edgeR and limma are good alternatives to DESEq2 for performing differential expression analyses." This suggests that you can't adjust for confounders in DESeq2, is this so? Did you not have access to any relevant cofactors in the SLE data?

p. 9: You are really using 59000 genes after vst to do exploratory visualization of SLE vs control expression patterns? Would gene filtering be helpful? Is there any chance of batch effect or other surrogate variable effect that should be assessed prior to such presentations?

By page 12 we have completed a relatively elementary differential expression analysis. It seems to me that the length of this part of the process is excessive, because the real interest is in learning about regulatory elements from other resources.

At this point I hope I have made clear how I think the rest of the paper should be revised to make its points more effectively.

Is the rationale for developing the new method (or application) clearly explained?
Yes

Is the description of the method technically sound?
Partly

Are sufficient details provided to allow replication of the method development and its use by others?
Yes

If any results are presented, are all the source data underlying the results available to ensure full reproducibility?
Partly

Are the conclusions about the method and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 14 Feb 2018

Enrico Ferrero, GlaxoSmithKline, UK

Vincent, many thanks for reviewing my paper in depth.
In response to your comments (please note that I took the liberty to format some of your points and omit some parts for readability):

- [...] I think this undertaking is very worthwhile from several perspectives [..] and promoter capture data whose origins could be better described.
--> Promoter capture Hi-C is indeed briefly introduced as a technique in the introduction. I have now added some more context on the Javierre et al., 2016 dataset and emphasised its relevance for this workflow at the beginning of the "Promoter capture Hi-C data" subsection.

- [...] The paper is full of "dumps" of show events for R objects that impede the narrative flow drastically.
--> I didn't realise how annoying this was until you mentioned it. I removed the great majority of dumps, leaving only a few to document the structure of datasets just imported or very final objects. For all dumps, I also ensured that a minimal amount of rows were printed.

- A diagram that shows how the various resources combine in a scientifically coherent way would be a huge step forward for the paper and for practitioners.
--> I included a diagram providing a schematic overview of the workflow as figure 1 and referenced it in the last paragraph of the introduction. Please note that the diagram is created in R with the DiagrammeR package but the code is hidden as it is not strictly relevant for the purposes of the workflow.

- More reckoning of limitations that arise from complexity is also in order. eQTLs are far from simple, and should not be used as 'lists'. Enhancer and promoter 'lists' also need to be used with care.
--> I added a few sentences at the beginning of the "eQTL data" subsection cautioning on the complexity of GWAS/eQTL integration and provided a short overview of available alternatives which are more methodologically robust.

- [...] In particular, it would be great to see how the different elements of the system contribute to

the ultimate enumeration of targets.
--> I added figure 7 to show the relative contributions of the different approaches to the final results.

- [...] The introduction concludes with mention of STOPGAP, and POSTGAP, and INFERNO, but it is not clear whether the paper is intended to describe how content of STOPGAP is developed from basic data resources like those readily available to Bioconductor users.
--> I expanded that paragraph to provide more context on STOPGAP, POSTGAP and INFERNO and to clarify the intent of mentioning those resources in the introduction.

- I feel that the introduction, though well-referenced, is too long and does not clearly state the paper's main goal.
--> I shortened the introduction by removing the paragraph about GWAS and PheWAS and by removing or shortening several other sentences. I added a short, final paragraph stating more clearly the main goals of the workflow.

- There is no discussion of the experimental design underlying the RNA-seq study. Presumably the data were generated from this component of ref 28: [...]
--> That's correct. I added more context on the original study, including an overview of the experimental design, in the third paragraph of the "Gene expression data and differential gene expression analysis" section.

- There is no discussion of heterogeneity of SLE or the difficulty of learning from a collection of 18 cases. A reference to https://www.ncbi.nlm.nih.gov/pubmed/25102991 may be in order.
--> I addressed this point with a better introduction to SLE and its heterogeneity in the second paragraph of the "Gene expression data and differential gene expression analysis" section.

- [...] Instead of printing out a dataframe on p.7, it would be much better to have a contingency table showing the final layout of case and control characteristics.
--> I agree this is a more effective way to summarise the data. I removed the dataframe printing and included a contingency table showing features of case and control samples.

- p.7 "For simplicity, we select the first 18 (healthy) and the last 18 (SLE) samples from the original RangedSummarizedExperiment object". Is this essential to the performance of the workflow? Would a more systematic matching be possible? What kind of "simplicity" does this arbitrary selection create? I understand that the main purpose of the paper is to illustrate a process, but if this thinning of the data is not essential to the illustration, why do it?
--> Indeed, this was mostly done to speed up execution while compiling the document. I removed that chunk and all 117 samples are now used in the analysis.

- p. 8: "Note that we used an extremely simple model; in the real world you will probably need to account for co-variables, potential confounders and interactions between them. edgeR and limma are good alternatives to DESEq2 for performing differential expression analyses." This suggests that you can't adjust for confounders in DESeq2, is this so? Did you not have access to any relevant cofactors in the SLE data?
--> I reworded that sentence to clarify that DESEq2 is equivalent to edgeR and limma when it comes to multiple cofactors in the model. I also included a better description of the metadata available for this dataset and explained why it is not possible to include demographic statistics (unavailable) or other experimental factors (collinear with disease status) in the model.

- p. 9: You are really using 59000 genes after vst to do exploratory visualization of SLE vs control expression patterns? Would gene filtering be helpful? Is there any chance of batch effect or other surrogate variable effect that should be assessed prior to such presentations?
--> I have now applied a simple filter to remove genes with extremely low counts directly on the dds object and ahead of VST, as documented in the DESeq2 vignette [1] and the Bioconductor RNA-seq workflow [2]. This reduces the number of genes considerably, helping to speed up code execution too. I also clarified in the "Gene expression data and differential gene expression analysis" section that one of the aims of the hierarchical clustering and PCA in figure 2 and 3 is indeed to assess presence of batch effects or surrogate variables. Note that all available experimental variables are now included as annotation in the heatmap in figure 1.

- By page 12 we have completed a relatively elementary differential expression analysis. It seems to me that the length of this part of the process is excessive, because the real interest is in learning about regulatory elements from other resources.
--> The "Gene expression data and differential gene expression analysis" has now been considerably condensed by removing superfluous object dumps, merging code chunks and reducing the text to a minimum. One could go as far as removing the exploratory data analysis and figures, but I'd rather keep them to provide some context and a minimal differential expression analysis to be used as the starting point for the integration of the GWAS data.

- At this point I hope I have made clear how I think the rest of the paper should be revised to make its points more effectively.
--> Indeed. The workflow was largely redacted, condensed and improved by limiting R object dumps, providing more context on the features of the datasets used and more insights into the methodology and results of the analysis through the use of visualisations and data summaries.

[1] https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html
[2] https://www.bioconductor.org/help/workflows/rnaseqGene/

Competing Interests: No competing interests were disclosed.

Referee Report 05 February 2018

doi:10.5256/f1000research.14748.r30355

? **Aaron T. L. Lun**
CRUK (Cancer Research UK) Cambridge Institute, University of Cambridge, Cambridge, UK

This workflow is clear, well-written and makes good use of existing resources and Bioconductor software. It addresses an interesting problem in the integration of RNA-seq, GWAS, eQTL and Hi-C data for causal gene discovery in disease contexts. However, it would benefit from some more elaboration in certain sections. I have listed my comments below in more detail, ordered by the location in the workflow they refer to. For most part, I believe they are easily addressed.

- The final paragraph of the introduction seems out of place; I do not see any reference to POSTGAP, STOPGAP or INFERNO anywhere else in the article. Was the workflow presented here used to identify the candidate genes in these resources?

- A more comprehensive description of the SLE data set, and the motivation behind using it, would be helpful.

- There seems to be a typo when loading the SRP062966 dataset; it should be **load(file.path("SRP062966", "rse_gene.Rdata"))**, at least on my machine.

- I don't see why it's desirable to call **scale_counts()**. Major DE analysis frameworks are easily capable of handling differences in library sizes. Direct scaling would actually be detrimental to NB models like *edgeR* and *DESeq2*, as it distorts the mean-variance relationship. In particular, scaled counts can have sub-Poisson variation, which cannot be handled by NB models. It seems better to call **read_counts()** to obtain the gene-level read counts.

- **rse$FIELD** can be used instead rather than **colData(rse)$FIELD**, which may simplify the code.

- Some explanation of the other factors (anti-rho, ISM) would be helpful, given that the effort has already been taken to define them.

- The simplicity of the model used in the DE analysis is probably unhelpful in the context described in the workflow. I would like to see more elaboration on how to handle batch effects and other confounding factors that are almost definitely present in large-scale studies. For example, what happens to the DE genes when additional explanatory factors are added to the model, e.g., anti-rho or ism status? Presumably age and sex are also relevant factors, if that information is available in the data set.

- Generally, some of the plots could be accompanied by more commentary in text, explaining how to interprete the plot. For example, the MA plot in Figure 3 shows that DE genes are detected in both directions, at a range of abundances. It would be similarly useful to have text for the heatmap in Figure 1 and the Manhattan plot in Figure 4, among others.

- LD expansion seems like quite an important step, especially when SNPs are being linked to genes based on overlaps to promoters/UTRs. If the LD blocks are large, expansion would result in many more potential causal SNPs and a greater number of overlaps (and thus candidate genes). While I appreciate the attempt to simplify the workflow, skipping this step seems like it would unnecessarily reduce the number of candidate genes.

- **snps** seems to have GRCh38 coordinates. Is this also the case for GENCODE 25? It would be helpful to have a cautionary note regarding the need to make sure the same version of the genome is used throughout a workflow. I recognise that this is mentioned later when **liftOver()** is used, but it is better to be explicit about this where possible.

- Oscillating between **head()** and **tail()** to preview the dataset is unhelpful and confusing.

- While I don't expect a thorough examination of the set of (7 easy, 4 hard, 3 via Hi-C) candidate genes for SLE, some discussion of the biological significance of the detected genes would be appreciated. It would provide a high-level validation of the workflow and link it back to the drug discovery context.

- For the promoter Hi-C section, you could consider using the **linkOverlaps()** method in the *InteractionSet* package, to link SNPs to gene promoters via the identified Hi-C interactions. This might be simpler than the current code, and possibly faster; the **nearest()** step in particular takes quite a long time.

Is the rationale for developing the new method (or application) clearly explained?
Yes

Is the description of the method technically sound?
Yes

Are sufficient details provided to allow replication of the method development and its use by others?
Yes

If any results are presented, are all the source data underlying the results available to ensure full reproducibility?
Partly

Are the conclusions about the method and its performance adequately supported by the findings presented in the article?
Partly

Competing Interests: No competing interests were disclosed.

Referee Expertise: Computational biology, bioinformatics

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 14 Feb 2018

Enrico Ferrero, GlaxoSmithKline, UK

Aaron, many thanks for reviewing my paper in depth.
In response to your comments:

- The final paragraph of the introduction seems out of place; I do not see any reference to POSTGAP, STOPGAP or INFERNO anywhere else in the article. Was the workflow presented here used to identify the candidate genes in these resources?
--> I expanded the paragraph to provide more context on STOPGAP, POSTGAP and INFERNO and to clarify why they are mentioned in the introduction but not used in the actual workflow.

- A more comprehensive description of the SLE data set, and the motivation behind using it, would be helpful.
--> More background and details on the dataset have been added in the second and third paragraph of the section "Gene expression data and differential gene expression analysis".

- There seems to be a typo when loading the SRP062966 dataset; it should be load(file.path("SRP062966", "rse_gene.Rdata")), at least on my machine.
--> Fixed, thanks.

- I don't see why it's desirable to call scale_counts(). Major DE analysis frameworks are easily capable of handling differences in library sizes. Direct scaling would actually be detrimental to NB models like edgeR and DESeq2, as it distorts the mean-variance relationship. In particular, scaled counts can have sub-Poisson variation, which cannot be handled by NB models. It seems better to call read_counts() to obtain the gene-level read counts.
--> For this section, I followed the recount quick start guide [1] and workflow [2]. Both show scaling of the counts with scale_counts() before feeding these to DESeq2. I tried switching to read_counts() but, somewhat counter-intuitively, the function returns values with decimal numbers, which in turn causes an error ("some values in assay are not integers") when calling the DESeqDataSet() function. As both scale_counts() and read_counts() seem to be acceptable, but the first one is the preferred approach by the recount developers, I switched back to scale_counts() after encountering the DESeq2 error above. The other option would have been to manually round the numbers returned by read_counts() but that seemed more questionable to me than scaling them.

- rse$FIELD can be used instead rather than colData(rse)$FIELD, which may simplify the code.
--> Simplified, thanks.

- Some explanation of the other factors (anti-rho, ISM) would be helpful, given that the effort has already been taken to define them.
--> I added context for these experimental factors in the third paragraph of the "Gene expression data and differential gene expression analysis" section and after printing the rse$characteristics object is printed.

- The simplicity of the model used in the DE analysis is probably unhelpful in the context described in the workflow. I would like to see more elaboration on how to handle batch effects and other confounding factors that are almost definitely present in large-scale studies. For example, what happens to the DE genes when additional explanatory factors are added to the model, e.g., anti-rho or ism status? Presumably age and sex are also relevant factors, if that information is available in the data set.
--> I agree this is not ideal, but there are good reasons why other factors are not included. First, age, gender or other demographics are not available for this dataset. Second, the ISM and anti-Ro factors are disease characteristics and are obviously only measured on the SLE patients (and not on the healthy ones). If either or both of those factors are included in the model, you get the classic "model matrix is not full rank" error [3] because they are both collinear with the disease status (all healthy samples are "control" for both anti-Ro and ISM). I've been more explicit about these shortcomings in the paragraph following the code chunk where the model is built.

- Generally, some of the plots could be accompanied by more commentary in text, explaining how to interprete the plot. For example, the MA plot in Figure 3 shows that DE genes are detected in both directions, at a range of abundances. It would be similarly useful to have text for the heatmap in Figure 1 and the Manhattan plot in Figure 4, among others.
--> I expanded the main text and legends for figures 2, 4 and 5 (previously 1, 3 and 4) to include a better description and explanation of the plots. I believe figures 3 and 6 (previously 2 and 5) were already adequately described. I also added 3 new figures (1, 7 and 8) to clarify the steps involved in the workflow and to provide a more in-depth understanding of the final results.

- LD expansion seems like quite an important step, especially when SNPs are being linked to genes based on overlaps to promoters/UTRs. If the LD blocks are large, expansion would result in

many more potential causal SNPs and a greater number of overlaps (and thus candidate genes). While I appreciate the attempt to simplify the workflow, skipping this step seems like it would unnecessarily reduce the number of candidate genes.
--> Unfortunately I can't come up with a good way to perform this step in R as part of the workflow at present. The ldblock package hasn't been updated in a while and its functions rely on downloading the HapMap data from the NCBI website, which was retired in 2016 and is no longer available for download [4]. Even if it was still available, it would require downloading several GBs of data, one chromosome at a time. The previously referenced trio package uses data structures specific to case - parent trio studies which are not compatible with the use case presented in the workflow and are not designed for hundreds of SNPs, and was thus removed. The Ensembl LD Calculator is a web UI with a limit of 20 SNPs per query that can't be integrated in a programmatic workflow, so it was removed too. I guess the Ensembl REST API could be an option, but it would require introducing a few new libraries and a considerable amount of code to interact with the API and parse its output into R/Bioconductor objects, with the risk of distracting the reader from the main purpose of this (Bioconductor) workflow. It would also require performing several hundreds queries in a for loop making compilation of the document extremely long and following the workflow impractical. I modified the text in the manuscript to communicate more clearly the reasons for skipping this step. If you have other suggestions on how to do this, I would be happy to consider them.

- snps seems to have GRCh38 coordinates. Is this also the case for GENCODE 25? It would be helpful to have a cautionary note regarding the need to make sure the same version of the genome is used throughout a workflow. I recognise that this is mentioned later when liftOver() is used, but it is better to be explicit about this where possible.
--> I added a clarification and a warning about this in the third paragraph of the "Accessing GWAS data" section, after importing the GWAS data.

- Oscillating between head() and tail() to preview the dataset is unhelpful and confusing.
--> I removed all instances of tail() and replaced them with head().

- While I don't expect a thorough examination of the set of (7 easy, 4 hard, 3 via Hi-C) candidate genes for SLE, some discussion of the biological significance of the detected genes would be appreciated. It would provide a high-level validation of the workflow and link it back to the drug discovery context.
--> I added a new section "Functional analysis of prioritised hits" (and a new figure, 8) where I describe the biological significance and functional relevance of the results, while also discussing some of the hits in more detail from a drug discovery perspective.

- For the promoter Hi-C section, you could consider using the linkOverlaps() method in the InteractionSet package, to link SNPs to gene promoters via the identified Hi-C interactions. This might be simpler than the current code, and possibly faster; the nearest() step in particular takes quite a long time.
--> Thanks, I had heard of the InteractionSet package but hadn't used it before. I agree it's better to represent the promoter capture Hi-C data in this native structure. I still had to use the nearest() function (which executes almost instantaneously on my laptop) to map promoters to gene IDs though. Also, note that I didn't need the linkOverlaps() function in the end and simply used findOverlaps(..., use.region = "second") instead.

[1] https://bioconductor.org/packages/3.7/bioc/vignettes/recount/inst/doc/recount-quickstart.html
[2] https://f1000research.com/articles/6-1558/v1
[3] http://seqanswers.com/forums/showthread.php?t=33032
[4] https://www.ncbi.nlm.nih.gov/variation/news/NCBI_retiring_HapMap/

Competing Interests: No competing interests were disclosed.