DNA Methylation Deconvolution Protocol

Michael Scherer, Petr Nazarov, Reka Toth, Shashwat Sahay, Tony Kamoa, Valentin Maurer,

December 05, 2019

Introduction

This protocol aims at guiding reasearcher how to employ deconvolution of methylomes obtained from complex tissues. It will start with data retrieval from a public resource, but is equally applicable to in-house generated data. We will furthermore focus on the Illumina BeadChip series as a data source, although the protocol is also compatible with bisulfite sequencing, or any technology yielding single base pair resolution. Deconvolution here refers to creating two matrices (proportion matrix A and methylation pattern matrix T) from a single matrix of input DNA methylation data (dimension CpGs x samples). Non-negative matrix factorization is employed for this task, and we will discuss some of the advantages and caveats of the methods.

Installation (duration up to 2 h)

To execute the protocol described here, several R-packages (*RnBeads*, *DecompPipeline*, *MeDeCom*, and *FactorViz*) are required. First, you should have a recent R version installed and all your packages updated. The installation of the software is appliable for most *Linux distributions* directly through R, while for *MacOS* the binary release of *MeDeCom* (https://github.com/lutsik/MeDeCom/releases/download/v0.3.0/MeDeCom_0.3.0.tgz)[https://github.com/luts should be used. The protocol is also available as a Docker container (https://hub.docker.com/r/mscherer/medecom)[https://hub.docker.com/r/mscherer/medecom], which is the only option for Windows operating systems. For the remainder of this protocol, we assume a common Linux distribution as the operating system.

```
install.packages(c("devtools","BiocManager"))
BiocManager::install(c("RnBeads","RnBeads.gh19","RnBeads.mm10","RnBeads.hg38"),dependencies=
devtools::install_github(c("lutsik/MeDeCom","CompEpigen/DecompPipeline","CompEpigen/FactorVi
library(DecompPipeline)
```

Protocol

Data Retrival

Obtaining data from a public resource (duration ~5h)

We focus on DNA methylation data from cancer patients that has been generated in The Cancer Genome Atlas (TCGA) project. Since lung cancer has been shown to be a premier candidate for DNA methylation based deconvolution, we selected the lung adenocarcinoma dataset from the TCGA website (dataset TCGA-LUAD, https://portal.gdc.cancer.gov/legacy-archive/search/f). The dataset was generated using the Illumina Infinium 450k BeadChip and comprises 461 samples. The clinical metadata of the samples is available at https://portal.gdc.cancer.gov/projects/TCGA-LUAD and lists 585 samples, and is stored in the folder *anntotation*. The discrepancy between the number comes from recent progress within TCGA. We used the Genomic Data Commons (GDC) data download tool (https://gdc.cancer.gov/access-data/gdc-data-transfer-tool) to download the intensity data (IDAT) files listed in the manifest file and its associated metadata into a new folder *idat*. This metadata also includes the mapping between each of the samples and the IDAT files. To create a final mapping and to prepare the files for downstream analysis, the following code was employed.

```
clinical.data <- read.table("annotation/clinical.tsv",sep="\t",header=T)
idat.files <- list.files("idat",full.names = T)
meta.files <- list.files(idat.files[1],full.names = T)
untar(meta.files[3],exdir = idat.files[1])
meta.files <- untar(meta.files[3],list=T)
meta.info <- read.table(file.path(idat.files[1],meta.files[5]),sep="\t",header=T)
meta.info <- meta.info[match(unique(meta.info$Comment..TCGA.Barcode.),meta.info$Comment..TC
match.meta.clin <- match(clinical.data$submitter_id,substr(meta.info$Comment..TCGA.Barcode.,
anno.frame <- na.omit(data.frame(clinical.data,meta.info[match.meta.clin,]))
anno.frame$barcode <- unlist(lapply(lapply(as.character(anno.frame$Array.Data.File),functio
anno.frame$Sentrix_ID <- unlist(lapply(lapply(as.character(anno.frame$Array.Data.File),funct
anno.frame$Sentrix_Position <- unlist(lapply(lapply(as.character(anno.frame$Array.Data.File)
anno.frame$healthy_cancer <- ifelse(grepl("11A",anno.frame$Comment..TCGA.Barcode.),"healthy"
write.table(anno.frame,"annotation/sample_annotation.tsv",quote=F,row.names = F,sep="\t")
anno.frame <- read.table("annotation/sample_annotation.tsv",quote=F,row.names = F,sep="\t")

#' write idat files to parent directory
lapply(idat.files,function(x){
  is.idat <- list.files(x,pattern = ".idat",full.names = T)
  file.copy(is.idat,"idat/")
  unlink(x,recursive = T)
})
```

Data Processing

Data Import and Quality Control in RnBeads (~3h)

After downloading the data, it has to be processed into a format that can be used by downstream software. We used RnBeads to convert the files into a data object and performed basic quality control steps on the dataset. Most notably, analysis options need to be specified for RnBeads, either through an XML file or in the command line. We will follow the latter strategy here, and deactivate the preprocessing, exploratory, covariate inference and differential methylation modules. In the next step, we specify the input to RnBeads: the created sample annotation sheet, the folder in which the IDAT files are stored and a folder to which the HTML report is to be saved. We additionally recommend to specify a temporary directory for the analysis. Then we start the RnBeads analysis.

```
suppressPackageStartupMessages(library(RnBeads))
rnb.options(
  assembly="hg19",
  identifiers.column="submitter_id",
  import=T,
  import.default.data.type="idat.dir",
  import.table.separator="\t",
  import.sex.prediction=T,
  qc=T,
  preprocessing=F,
  exploratory=F,
  inference=F,
  differential=F,
  export.to.bed=F,
  export.to.trackhub=NULL,
  export.to.csv=F
)
sample.anno <- "annotation/sample_annotation.tsv"
idat.folder <- "idat/"
dir.report <- paste0("report",Sys.Date(),"/")
temp.dir <- "/tmp"
options(fftempdir=temp.dir)
rnb.set <- rnb.run.analysis(dir.reports = dir.report, sample.sheet = sample.anno, data.dir =
```

```
## 2019-12-11 09:56:43     1.1  STATUS STARTED RnBeads Pipeline
## 2019-12-11 09:56:43     1.1    INFO     Initialized report index and saved to index.html
## 2019-12-11 09:56:43     1.1  STATUS     STARTED Loading Data
## 2019-12-11 09:56:43     1.1    INFO         Number of cores: 1
## 2019-12-11 09:56:43     1.1    INFO         Loading data of type "idat.dir"
## 2019-12-11 09:56:43     1.1  STATUS         STARTED Loading Data from IDAT Files
```

```
## 2019-12-11 09:56:45     1.1    INFO             Detected platform: HumanMethylation450
## 2019-12-11 10:02:22     1.5  STATUS         COMPLETED Loading Data from IDAT Files
## 2019-12-11 10:37:38     2.0  STATUS         Loaded data from idat/
## 2019-12-11 10:37:55     7.6  STATUS         Predicted sex for the loaded samples
## 2019-12-11 10:38:03     7.1  STATUS         Added data loading section to the report
## 2019-12-11 10:38:03     7.1  STATUS         Loaded 461 samples and 485577 sites
## 2019-12-11 10:38:03     7.1    INFO         Output object is of type RnBeadRawSet
## 2019-12-11 10:38:03     7.1  STATUS     COMPLETED Loading Data
## 2019-12-11 10:51:21     7.1    INFO     Initialized report index and saved to index.html
## 2019-12-11 10:51:21     7.1  STATUS     STARTED Quality Control
## 2019-12-11 10:51:21     7.1    INFO         Number of cores: 1
## 2019-12-11 10:51:22     7.1  STATUS         STARTED Quality Control Section
## 2019-12-11 10:51:45     2.0  STATUS             Added quality control box plots
## 2019-12-11 10:55:46     2.0  STATUS             Added quality control bar plots
## 2019-12-11 10:56:10     2.0  STATUS             Added negative control boxplots
## 2019-12-11 10:56:10     2.0  STATUS         COMPLETED Quality Control Section
## 2019-12-11 10:56:10     2.0  STATUS         STARTED Visualizing SNP Probe Data
## 2019-12-11 10:56:10     2.0  STATUS             STARTED Mixups Visualization Section
## 2019-12-11 10:56:28     5.4  STATUS                 Added SNP Heatmap
## 2019-12-11 10:56:28     5.4  STATUS                 Calculated Manhattan distances betwee
## 2019-12-11 10:56:31     5.4  STATUS                 Added SNP-based Distances
## 2019-12-11 10:56:31     5.4  STATUS             COMPLETED Mixups Visualization Section
## 2019-12-11 10:56:31     5.4  STATUS         COMPLETED Visualizing SNP Probe Data
## 2019-12-11 10:56:49     7.6  STATUS     COMPLETED Quality Control
## 2019-12-11 10:56:49     7.6    INFO     Initialized report index and saved to index.html
## 2019-12-11 10:56:49     7.6  STATUS     STARTED Saving RData
## 2019-12-11 10:56:49     7.6  STATUS     COMPLETED Saving RData
## 2019-12-11 10:56:49     7.6  STATUS COMPLETED RnBeads Pipeline
```

BREAKPOINT We provide an example report containing the processed RnBSet object for further analysis. It can be obtained from (http://www.computational-epigenomics.com/downloads/DecompProtocol/RnBeads_Report_TCGA_LUAD/i
epigenomics.com/downloads/DecompProtocol/RnBeads_Report_TCGA_LUAD/index.html]
or directly loaded via:

```
rnb.set <- load.rnb.set("URL")
```

RnBeads creates an interactive HTML report, specifying the steps performed and the associated results. Data was of good quality such that it can be used for further analysis.

Preprocessing and Filtering

For further analysis, we use the *DecompPipeline* package (https://github.com/CompEpigen/DecompPipeline), which provides a comprehensive workflow including crucial data preparation

steps for methylome deconvolution experiments. The options are provided through the individual function parameters. We follow a stringent filtering strategy: (i) all samples having fewer than 3 beads covered are filtered, as well as those probes that are in the 0.05 and 0.95 overall intensity quantiles, respectively. (ii) We then remove all probes containing missing values, outside of CpG context, that overlap with annotated SNPs, on the sex chromosomes and probes that have been shown to be cross-reactive on the chip. (iii) Then, BMIQ normalization [@bmiq] is employed to account for the chip's design bias. Accounting for potential confounding factor is crucial in epigenomic studies. Especially, the influence of donor sex and age on the DNA methylation pattern is well-studied and strong. Furthermore, genetic differences between groups of individuals may influence the DNA methylation paterrn. We used Independent Component Analysis (ICA) to account for DNA methylation differences that are due to these confounding factors. ICA detects components in the data accounting for most of the variance similar to PCA, but does not require orthogonality of the components but statistical independence. We used an external library (http://sablab.net/scripts/LibICA.r)[http://sablab.net/scripts/LibICA.r] for performing ICA to adjust for sex, age, race and ethnicity.

```
suppressPackageStartupMessages(library(DecompPipeline))
data.prep <- prepare_data(RNB_SET = rnb.set,
                          analysis.name = "TCGA_LUAD4",
                          NORMALIZATION = "bmiq",
                          FILTER_BEADS = T,
                          MIN_N_BEADS = 3,
                          FILTER_INTENSITY = T,
                          MIN_INT_QUANT = 0.001,
                          MAX_INT_QUANT = 0.999,
                          FILTER_NA = T,
                          FILTER_CONTEXT = T,
                          FILTER_SNP = T,
                          FILTER_SOMATIC = T,
                          FILTER_CROSS_REACTIVE = T,
                          execute.lump=T,
                          remove.ICA=T,
                          conf.fact.ICA=c("age_at_diagnosis","race","gender","ethnicity"),
                          ica.setting=c("alpha.fact"=1e-5,"save.report"=T,"nmin"=30,"nmax"=5
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 30 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:26:17
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 12:31:46
## Time difference of 5.477402 mins
```

```
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 31 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:31:50
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 12:36:32
## Time difference of 4.692923 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 32 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:36:37
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 12:42:15
## Time difference of 5.629724 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 33 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:42:20
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 12:47:38
## Time difference of 5.301141 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 34 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:47:43
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 12:53:56
## Time difference of 6.217387 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 35 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 12:54:02
## Execute one-core analysis, showing progress every 1 run(s)
```

```
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:00:08
## Time difference of 6.110375 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 36 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:00:14
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:07:19
## Time difference of 7.086401 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 37 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:07:25
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:14:35
## Time difference of 7.181232 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 38 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:14:41
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:22:31
## Time difference of 7.828324 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 39 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:22:37
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:31:11
## Time difference of 8.578227 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
```

```
## *** System: unix
## *** 40 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:31:18
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:40:55
## Time difference of 9.616655 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 41 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:41:01
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:50:26
## Time difference of 9.419989 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 42 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 13:50:32
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 13:59:56
## Time difference of 9.391567 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 43 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:00:02
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 14:11:14
## Time difference of 11.20148 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 44 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:11:21
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
```

```
## *** End time: 2019-12-11 14:20:46
## Time difference of 9.415461 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 45 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:20:52
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 14:31:01
## Time difference of 10.13918 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 46 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:31:07
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 14:42:34
## Time difference of 11.44213 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 47 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:42:41
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 14:56:22
## Time difference of 13.68558 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 48 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 14:56:29
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 15:12:16
## Time difference of 15.7844 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 49 components, 1 runs, 230223 features, 461 samples.
```

```
## *** Start time: 2019-12-11 15:12:23
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 15:24:37
## Time difference of 12.23424 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 50 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 15:24:45
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 15:38:18
## Time difference of 13.55522 mins
## Calculate ||X-SxM|| and r2 between component weights
## *** Starting  calculation on 1 core(s)...
## *** System: unix
## *** 46 components, 1 runs, 230223 features, 461 samples.
## *** Start time: 2019-12-11 15:38:33
## Execute one-core analysis, showing progress every 1 run(s)
## try # 1 of 1
## *** Done!
## *** End time: 2019-12-11 15:51:05
## Time difference of 12.53453 mins
```

BREAKPOINT We provide a list of CpGs that are selected for the downstream analysis at (http://www.computational-epigenomics.com/downloads/DecompProtocol/sites_passing_complete_epigenomics.com/downloads/DecompProtocol/sites_passing_complete_filtering.csv].
You can directly select those sites from you RnBSet object.

```
rem.sites <- rep(TRUE,nrow(rnb.set))
sel.sites <- read.csv("URL")
rem.sites[row.names(annotation(rnb.set))%in%as.character(sel.sites[,1])] <- FALSE
rnb.set <- remove.sites(rnb.set,remove.sites)
data.prep <- list(rnb.set.filtered=rnb.set)
```

Selecting informative features (CpGs)

The next step is selecting a subset of sites that are informative about the cell type composition of your sample. This can be done in various ways, and *DecompPipeline* provides a list of them through the `prepare_CG_subsets` function. However, we focus on selecting the most variable sites across the samples, which is a common strategy in epigenomic studies. Since many sites are constant for

all samples, focusing on the ones that show the highest variablity across the samples reduces running time substantially. Lowly variable sites tend not to contribute to pattern discovery. Here, we focus on the 5,000 most variable sites.

```
cg_subset <- prepare_CG_subsets(rnb.set=data.prep$rnb.set.filtered,
                                MARKER_SELECTION = "var",
                                N_MARKERS = 5000)
names(cg_subset)
```

Methylome Deconvolution

Performing Deconvolution

In this step, the actual deconvolution experiment is performed. There are different approaches, which are conceptually similar, yet different in their performance, running time and robustness. Among others, *EDec*, *RefFreeCellMix* from the *RefFreeEWAS* package and *MeDeCom* can be used to execute non-negative matrix factorization on your data. This will lead to two matrices, the proportions matrix of potential cell types (here referred to as LMCs) and the matrix of the latent methylation components (LMCs). We here focus on *MeDeCom* as the Deconvolution tool, although *DecompPipeline* also supports *RefFreeCellMix* and *EDec*.

```
md.res <- start_medecom_analysis(
  rnb.set=data.prep$rnb.set.filtered,
  cg_groups = cg_subset,
  Ks=2:15,
  LAMBDA_GRID = c(0,10^-(2:5)),
  factorviz.outputs = T,
  analysis.name = "TCGA_LUAD4",
  cores = 15
)
## [1] "Did not write the variable dump: should only be executed from an environment with al
## [2019-12-11 16:22:58, Main:] checking inputs
## [2019-12-11 16:22:58, Main:] preparing data
## [2019-12-11 16:22:58, Main:] preparing jobs
## [2019-12-11 16:22:58, Main:] 3570 factorization runs in total
## [2019-12-13 04:21:09, Main:] finished all jobs. Creating the object
```

BREAKPOINT In case the analysis does not finish successfully, we provide the resulting `FactorViz_outputs` from (http://www.computational-epigenomics.com/downloads/DecompProtocol/FactorViz_outputs.tar.gz)[http://www.computational-epigenomics.com/downloads/DecompProtocol/FactorViz_outputs.tar.gz]. It can be directly loaded into your R session and further explored.

```
download.file("URL")
untar("FactorViz_outputs.tar.gz")
```

Downstream analysis

After performing deconvolution, results need to be visualized and interpreted. Most notably, the contribution matrix can be linked to phenotypic information about the samples to indicate different cellular compositions of the groups and the LMC matrix can be used for enrichment analysis and biological interpretation. For visualization and downstream analysis, we use *FactorViz*. *LOLA* and *GO* enrichment analysis can be employed on sites that are specifically methylated/unmethylated in one of the LMCs. The LMC proportions can also be linked to expression of marker genes/methylation of marker CpGs.

```
suppressPackageStartupMessages(library(FactorViz))
startFactorViz(file.path(getwd(),"TCGA_LUAD","FactorViz_outputs"))
```

References