CellMix: Sample Analyses

Renaud Gaujoux Cathal Seoighe

April 29, 2013

Abstract

Gene expression experiments are commonly performed using samples that are composed of a variety of cell types, each contributing to the global gene expression measurements in different proportions. The *CellMix* package defines a general framework for gene expression deconvolution that integrates several of the currently available deconvolution algorithms, and facilitates the estimation of cell type proportions and/or cell-specific differential expression in gene expression experiments.

The package was designed to handle common data settings using a unified interface, and to help in dealing with specific issues such as cross-platform/species identifier conversions. This vignette illustrates some of the package's key functionalities on practical sample analyses.

Contents

1 Setting up the *CellMix* package

Until the package is released on CRAN (R Development Core Team 2011) or BioConductor (Gentleman et al. 2004), the *CellMix* package can be installed from our CRAN-like repository, using standard package installation commands:

```
# install biocLite if not already there
if (!require(BiocInstaller)) {
    # enable Bioconductor repositories -> add Bioc-software
    setRepositories()

    install.packages("BiocInstaller")
    library(BiocInstaller)
}
# or alternatively do: source('http://www.bioconductor.org/biocLite.R')

# install (NB: might ask you to update some of your packages)
biocLite("CellMix", siteRepos = "http://web.cbio.uct.ac.za/~renaud/CRAN")
# load
library(CellMix)
```

Note that to reproduce some of the examples in this vignettes, you will also need to have the *GEOquery* package[1] package installed as well as some Bioconductor annotation packages which will be installed when needed[2]:

```
biocLite("GEOquery")
```

2 Background and objectives

Gene expression experiments typically assess heterogeneous samples in a single experiement, measuring *global gene expression* levels. From a biological point of view, however, different cell populations are fundamental units, for which specific gene expression profiles, and, in some cases, proportions can provide more detailed and meaningful insights about the underlying processes.

Gene expression deconvolution methodologies have been developed by several authors in order to answer the following questions, using global gene expression data, possibly in combination with some other data such as known signatures or marker genes:

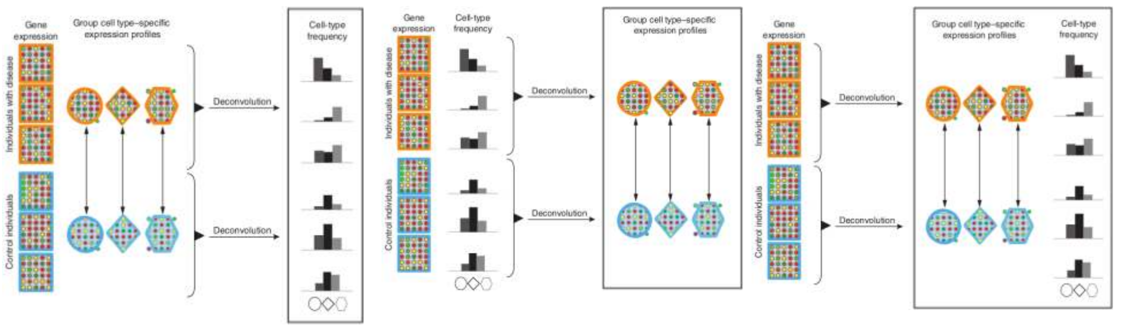
- cell proportions: what is the proportion of each cell type?
- are there differences in proportion that are associated with some disease status/covariate?
- cell-specific expression: what is the expression profile of each cell type?
- which genes are differently expressed in each cell types between groups of samples?

In terms of data requirement, most approaches fall into one of the three categories illustrated in Figure 1.

Gene expression deconvolution is naturally expressed as matrix decomposition problem. This is why we use the theoretical framework of Nonnegative Matrix Factorization (Lee et al. 1999; Paatero et al. 1994), for which we use the*NMF* package[3] (Gaujoux et al. 2010), because it provides a flexible interface to simultaneously handle both celll-specific signatures and proportion data, as well as a comprehensive set of features to run NMF(-like) algorithms.

[1] http://www.bioconductor.org/packages/release/bioc/html/GEOquery.html
[2] In an interactive R session, their installation is automated, after the user gives permission to do so.
[3] http://cran.r-project.org/package=NMF

(a) *Partial from available signatures* **(b)** *Partial from available proportions* **(c)** *Complete from global expression*



Graphic (b) was extracted from Shen-Orr et al. (2010), and modified to produce the other panels.

Figure 1: *Partial vs. Complete gene expression deconvolution.*
Partial deconvolution methods assume that either signatures (a) or proportions (b) are available and use them to infer the unknown proportions and signatures respectively. Complete deconvolution methods (c) infer both cell-type signatures and proportions from the global gene expression data, possibly using extra data such as marker genes to guide or seed the estimation.

3 Estimating cell proportions from known signatures

Consider some gene expression data, obtained from heterogeneous samples, i.e. in which the cell type proportions are expected to vary across samples. Laboratory techniques exist to measure these proportions, e.g., FACS, however, these measurments are not always performed, or are limited in terms of which cell types can be measured. Some computational methods use known cell-specific expression profiles/signatures to estimate their respective cell proportions in a mixture, given its global gene expression data.

3.1 Blood samples

Some deconvolution methdologies have been developed specifically for blood samples, which are arguably the most common type of samples for which gene expression is assessed. The *CellMix* package provides an easy interface to apply deconvolution methods that use known cell-specific expression signatures, e.g., generated in independent studies (Abbas et al. 2009; Gong et al. 2011).

For the purpose of this tutorial we will use the dataset GSE20300[4] on acute kidney transplant rejection (ACR) from Shen-Orr et al. (2010), for which Complete Blood Count (CBC) data are available. The *CellMix* package stores a pre-processing pipeline for this dataset, which extracts all relevant data from otherwise composite and unfriendly sample annotation data. The dataset is loaded using the function `ExpressionMix`, which requires an internet connection on first execution, when the actual data are retrieved from the Gene Expression Omnibus (GEO) database[5][6] (Barrett et al. 2010). Subsequent calls use a locally cached version of the data and are much faster as shown below.

```
# load data (normally requires an internet connection to GEO)
acr <- ExpressionMix("GSE20300", verbose = 2)

## Loading dataset 'GSE20300' ...
##  Loading expression data GSE20300 from cache binary file '/home/renaud/R-data/CellMix/GSE20300.rds'  OK
```

[4] http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE20300
[5] http://www.ncbi.nlm.nih.gov/geo/
[6] *CellMix* includes special functions that make GEOquery works even behind CNTLM proxies, e.g., your university proxy.

```
##  Adding covariate(s): CEL, SampleID, Group, Neutrophils, Lymphocytes, Monocytes, Eosinophils,
##                       Basophils, Status, Type
##
##  Loading basis signatures ... NONE [0 x 5]
##  Loading proportions ... OK [5 x 24]
##  Warpping into an ExpressionMix object ... OK
## OK


# the result is a combination of an ExpressionSet and an NMF model object
acr

## ExpressionMix (storageMode: lockedEnvironment)
## assayData: 54675 features, 24 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM508883 GSM508884 ... GSM508906 (24 total)
##   varLabels: title geo_accession ... Type (39 total)
##   varMetadata: labelDescription
## featureData
##   featureNames: 1007_s_at 1053_at ... AFFX-TrpnX-M_at (54675
##     total)
##   fvarLabels: ID GB_ACC ... Gene Ontology Molecular Function (16
##     total)
##   fvarMetadata: Column Description labelDescription
## experimentData: use 'experimentData(object)'
## Annotation: hgu133plus2.db
## Composition: 'Neutrophils', 'Lymphocytes', ..., 'Basophils' (5 total)
```

Abbas et al. (2009) defined a set of cell-specifc signature for 17 immune cell types in a variety of states (e.g., activated or resting). Although these signatures were defined on Affymetrix *HGU133A* and *HGU133B* chips, they can theoretically be used on data generated on other platforms. This, however, potentially requires to map the original Affymetrix probe ids to corresponding probes on the other platform. One might also have concerns about how well correlated gene expression measures are between different platforms.

The function `gedBlood` enables to apply in a single call a pre-processing pipeline, which deals with such issues. We use argument `verbose=TRUE` to show some details about the pipeline workflow[7]:

```
# estimate proportions using signatures from Abbas et al. (2009)
res <- gedBlood(acr, verbose = TRUE)

## Loading basis signature from Abbas et al. (2009) ... OK [359 features x 17 cell types]
## Mapping signature ids onto target ids (method: auto) ... OK [321 features x 17 cell types]
## Limit/reorder to common set of features ... OK [321 features x 17 cell types]
## Checking data dimension compatibility ... OK [321 features x 17 cell types]
## Checking log-scale ... data:YES - signatures:NO
## Applying log-transform to signatures (base 2) ... OK
## Normalizing signatures and target together (method: quantiles) ... OK
##  Using ged algorithm: "lsfit"
##  Estimating cell proportions from cell-specific signatures [lsfit]
##  Timing:
##     user  system elapsed
##    0.697   0.012   0.710
##  GED final wrap up ...  OK
```

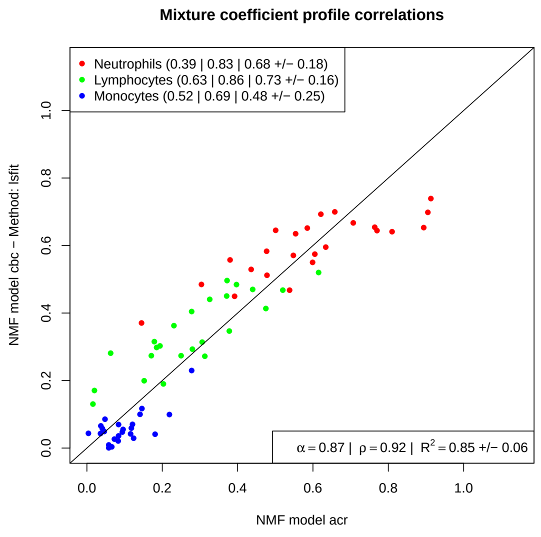
[7]More details can be displayed with `verbose=2`, `verbose=3`, etc..

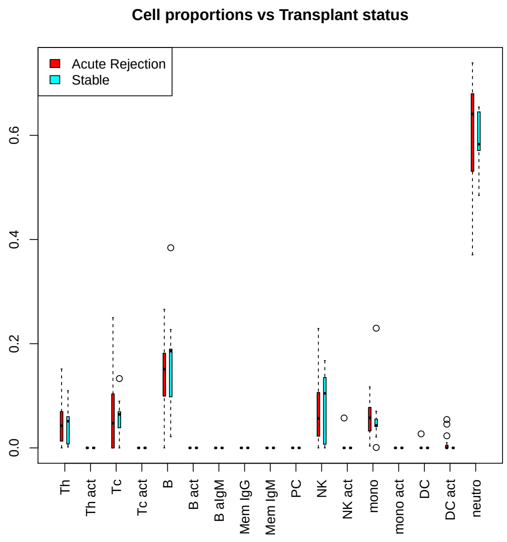
The result is stored in an NMF object that contains both the known basis signatures, and the estimated proportions in the coefficient matrix, which can be retrieve using the method **basis** and **coef**:

```
# result object
res

## <Object of class: NMFfit>
##  # Model:
##   <Object of class:NMFstd>
##   features: 321
##   basis/rank: 17
##   samples: 24
##  # Details:
##   algorithm:  lsfit
##   seed:  none
##   RNG: 403L, 1L, ..., -132249894L [50132e1245a756a255ba66ab526f5d6e]
##   distance metric:  'euclidean'
##   miscellaneous: lsfit=<list> . (use 'misc(object)')
##   Timing:
##      user  system elapsed
##     0.140   0.004   0.146


# proportions are stored in the coefficient matrix
dim(coef(res))

## [1] 17 24

coef(res)[1:3, 1:4]

##        GSM508883 GSM508884 GSM508885 GSM508886
## Th       0.06004   0.00427   0.09098  0.008294
## Th act   0.00000   0.00000   0.00000  0.000000
## Tc       0.08965   0.06816   0.01966  0.000000


# cell type names
basisnames(res)

##  [1] "Th"       "Th act"   "Tc"       "Tc act"   "B"        "B act"
##  [7] "B aIgM"   "Mem IgG"  "Mem IgM"  "PC"       "NK"       "NK act"
## [13] "mono"     "mono act" "DC"       "DC act"   "neutro"


# basis signatures (with converted IDs)
basis(res)[1:5, 1:3]

##                 Th Th act    Tc
## 232165_at    7.030  3.718 3.372
## 201369_s_at 12.482  6.880 8.084
## 204621_s_at  9.557  5.331 3.387
## 202861_at    8.325  4.196 5.421
## 222862_s_at  9.483  5.857 5.541
```

The proportions can be aggregated into CBC data using the function asCBC and, in this case, compared to the known proportions. Since the dataset contains a case/control group variable, Status for kidney transplant output, one can also highlight proportion differences between the two group of samples with the dedicated function boxplotBy:

```
# aggregate into CBC
cbc <- asCBC(res)
dim(cbc)

## [1] 321  24   3

basisnames(cbc)

## [1] "Lymphocytes" "Neutrophils" "Monocytes"


# plot against actual CBC
profplot(acr, cbc)
# plot cell proportion differences between groups
boxplotBy(res, acr$Status, main = "Cell proportions vs Transplant status")
```





3.2 Other tissues with known signatures

For other tissue types where a set of cell-specific signatures is available, the same analysis pipeline can be applied using the function `gedProportions`. For example, the dataset GSE19830[8] contains data from a controlled mixture experiment on Rat samples (Shen-Orr et al. 2010), where both the expression profiles from pure tissues (brain, liver and lung), as well as cell proportions from controlled mixtures are known. A data pre-processing pipline is also available in *CellMix*'s dataset registry, meaning that it can easily be loaded, as in the previous example. In particular, once the data is loaded, the average pure expression profiles, i.e. the cell-specific signature, are stored in the associated basis matrix, and can be retreived with the function `basis`, while the data restricted to all mixed and pure samples can be extracted via the functions `mixedSamples` and `pureSamples` respectively:

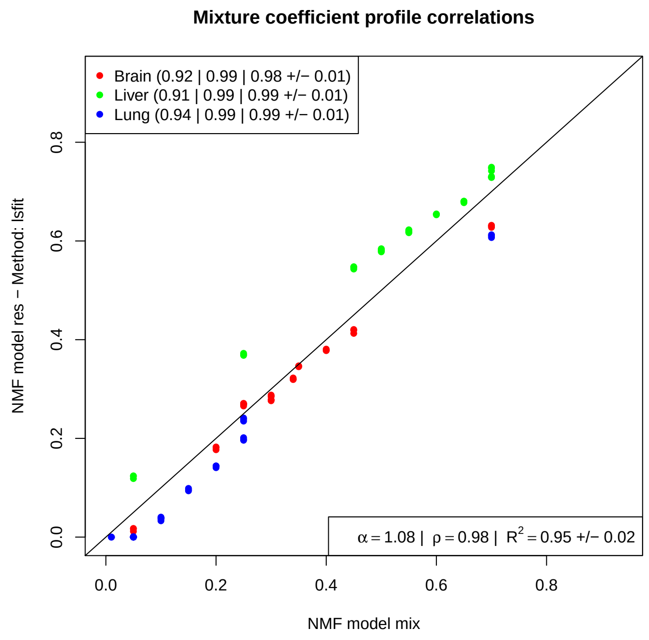
```
# load data (requires internet connection to GEO)
gse <- ExpressionMix("GSE19830", verbose = TRUE)

## Loading dataset 'GSE19830' ... OK

gse
```

[8] http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE19830

```
## ExpressionMix (storageMode: lockedEnvironment)
## assayData: 31099 features, 42 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM495209 GSM495210 ... GSM495250 (42 total)
##   varLabels: title geo_accession ... Type (35 total)
##   varMetadata: labelDescription
## featureData
##   featureNames: 1367452_at 1367453_at ... AFFX_ratb2/X14115_at
##     (31099 total)
##   fvarLabels: ID GB_ACC ... Gene Ontology Molecular Function (16
##     total)
##   fvarMetadata: Column Description labelDescription
## experimentData: use 'experimentData(object)'
## Annotation: rat2302.db
## Composition: 'Brain', 'Liver', 'Lung' (3 total)


# extract data for mixed samples only
mix <- mixedSamples(gse)

# extract stored known signatures (= average pure profiles)
sig <- basis(mix)
```

The same analysis pipeline used for blood samples can then be applied through the main interface function `ged`, provided that one also passes along the pure cell-specific signature matrix. In this example, the basis signatures in `sig` are those computed from the sorted cell samples themsleves as their average expression profile within each cell type. In practice, however, they could have been generated in an independent study. Moreover, no normalization is performed by default and we force the deconvolution to be performed in linear space (i.e. using non log-transformed expression values) by setting argument `log=FALSE`, since both the global expression and signature data come from the same dataset, and should not suffer from normalization or transformation discrepancies:

```
# estimate proportions from pure signatures (default: no normalization)
res <- ged(mix, sig, verbose = TRUE, log = FALSE)

##  Mapping signature ids onto target ids  (method: names) ...  ... SKIP
##  Limit/reorder to common set of features ...  OK [31099 features x 3 cell types]
##  Checking data dimension compatibility ...  OK [31099 features x 3 cell types]
##  Checking log-scale ...  data:YES  - signatures:YES
##  Reverting log-transform on signatures (base 2) ...  OK
##  Reverting log-transform on data (base 2) ...  OK
##  Normalizing signatures and target together (method: none) ...  SKIP
##   Using ged algorithm: "lsfit"
##    Estimating cell proportions from cell-specific signatures [lsfit]
##   Timing:
##    user  system elapsed
##   2.116   0.196   2.317
##   GED final wrap up ...   OK


# plot against known proportions
profplot(mix, res)
```



We notice that the estimates are separately very well correlated with the true proportions, but show some bias. This is fine if the proportions are not of interests, but are subsequently used to correct for proportion heterogeneity in the estimation of cell-specific differential expression. Indeed, estimates with high correlations will accurately reflect the variations in the true proportions, which is what eventually matters when estimating the effect of proportions on gene expression levels, at least using standard regression-like approaches.

However, more care is required if accurate proportion estimates are desired. Indeed, sets of complete expression profiles from pure samples generally need to be filtered to increase the deconvolution accuracy, and remove the bias due to noise or genes that are expressed across all/multiple cell types, whose expression is not expected to reflect cell proportions.

3.3 Building/filtering basis signatures

Abbas et al. (2009) proposed to select genes based on their cell type specificity, and build a basis signature matrix that provides the "maximum" deconvolution power, which the authors showed correlates well with its *condition number*. The main idea is to do so is to limit the signatures to a set of genes that discriminate well between cell types, and together constitute a basis matrix that is well conditonned. This also has the advantage of reducing the number of measurements needed to perform the deconvolution.

3.3.1 Extracting marker p-values

The first filtering step consists in computing p-values associated with cell type specificty. It is briefly described as follows in Abbas et al. (ibid.):

> [...] top differentially expressed (based on 95 change confidence intervals from Student's T-test) probesets were determined by comparing each probe's highest-expressed group with the next highest-expressed group in order to find probesets that are good markers for each cell population. This step was repeated with comparison between the top group and the third-highest group in order to also include probesets that were strong markers for two cell populations.
>
> From Abbas et al. (ibid.)

The *CellMix* package implements this step in the interface function `extractMarkers`, but by default only compares the top and second most expressing groups[9]:

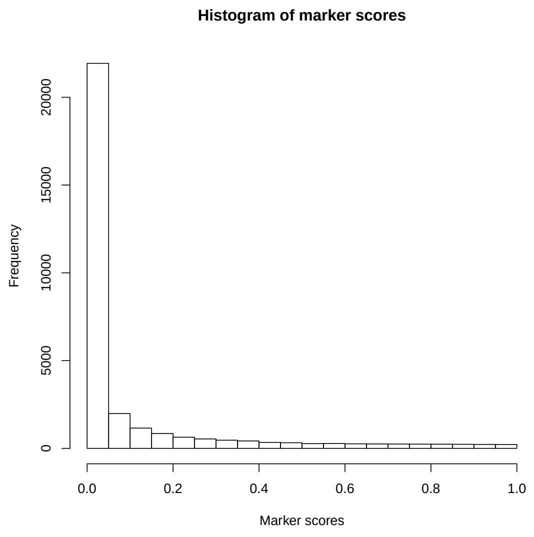
```
# extract data for pure samples only
pure <- pureSamples(gse)

# compute p-values for all probes
ml <- extractMarkers(pure, pure$Type, method = "Abbas")
# all probes get attributed a cell-type and a p-value
summary(ml)

## <object of class MarkerList>
## Types: 3 ['Brain', 'Liver', 'Lung']
## Mode: numeric
## Markers: 31099
## IDtype: .Affymetrix ['1387772_at', '1391092_at', ..., '1397229_at']
## Values: [5.84863829573202e-11, 5.63037162881734e-10, ..., 0.99972310651136]
## Source: rat2302.db
## Breakdown:
## Brain Liver  Lung
##  9104 12902  9093
```

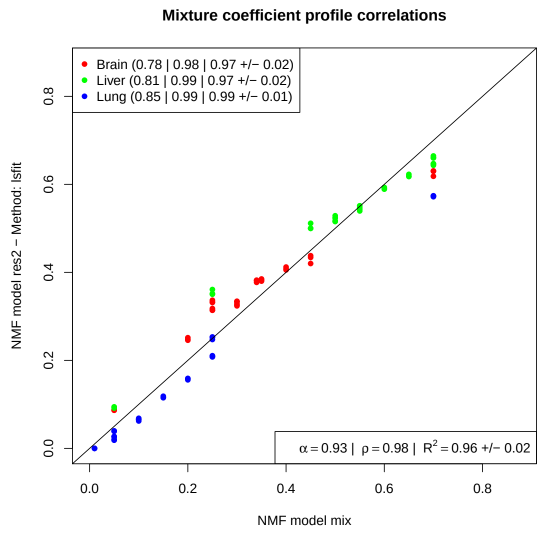
3.3.2 Filtering on p-value

One can estimate proportions only using markers with a p-value under a given threshold. This is facilitated by the rich interface of the S4 class `MarkerList`, which enables marker lists to be filtered in many different ways:

```
# Filtering 1: show p-values histogram
hist(ml, breaks = 20)
summary(ml <= 10^-8)

## <object of class MarkerList>
## Types: 3 ['Brain', 'Liver', 'Lung']
## Mode: logical
## Markers: 31099
## IDtype: .Affymetrix ['1387772_at', '1391092_at', ..., '1382072_at']
## Source: rat2302.db
## Breakdown (showing selected only):
## Brain Liver  Lung
##    17    38    14


# refit proportions using only the subset of markers with p-value <= 10^-8
res2 <- ged(mix, basis(gse), subset = ml <= 10^-8, log = FALSE)
# plot against known proportions
profplot(mix, res2)
```

[9]This can be changed setting argument `ntop > 2`, if some of the cell types are expected to be similar

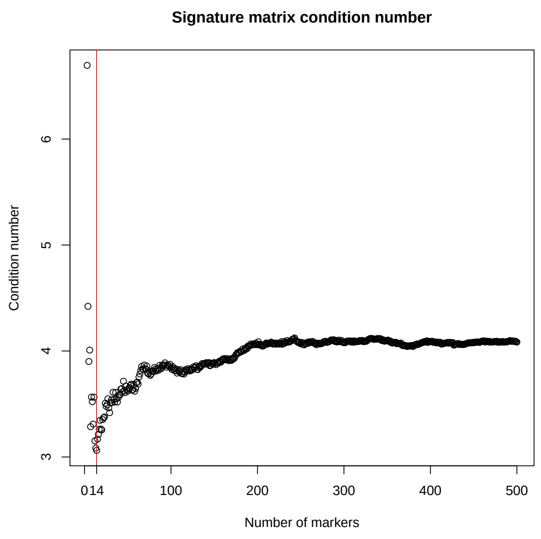


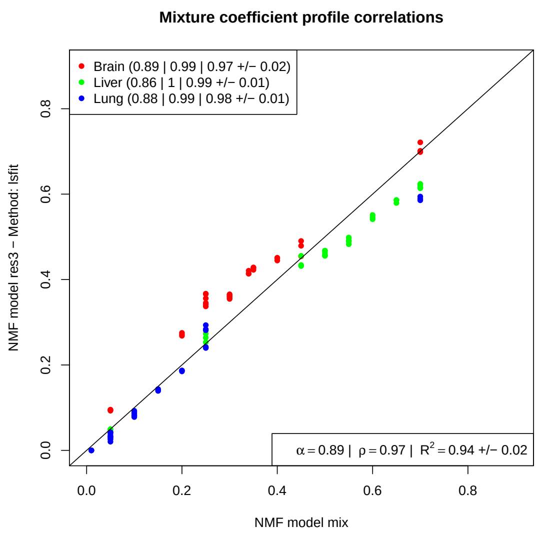


3.3.3 The condition number criterion

Abbas et al. (2009) suggested to further reduce the number of markers, by choosing the set for which the basis matrix's condition number is minimun, which aims at increasing the robustness and scale consistency of the basis signatures. This can be achieved using the function `screeplot`:

```
# Filtering 2: select limited number of markers based on the signature
# matrix's condition number as proposed by Abbas et al. (2009)
sel <- screeplot(ml, basis(gse), range = 1:500)
summary(sel)

## <object of class MarkerList>
## Types: 3 ['Brain', 'Liver', 'Lung']
## Mode: numeric
## Markers: 14
## IDtype: .Affymetrix ['1387772_at', '1391092_at', ..., '1388033_at']
## Values: [5.84863829573202e-11, 5.63037162881734e-10, ..., 2.13771098034808e-09]
## Source: rat2302.db
## Breakdown:
## Brain Liver  Lung
##     5     5     4


# refit proportions using the optimised set of markers
res3 <- ged(mix, basis(gse), subset = sel, log = FALSE)
# plot against known proportions
profplot(mix, res3)
```





3.4 From marker genes only

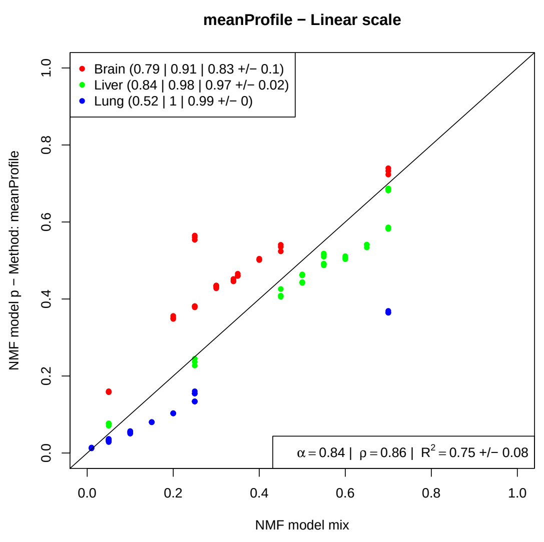
In the situation where no pure sample expression profile is available, deconvolution and estimation of cell type proportions can be still be performed, using sets of marker genes, i.e. genes that are know or at least expected to be expressed by only one of the cell types present in the mixture.

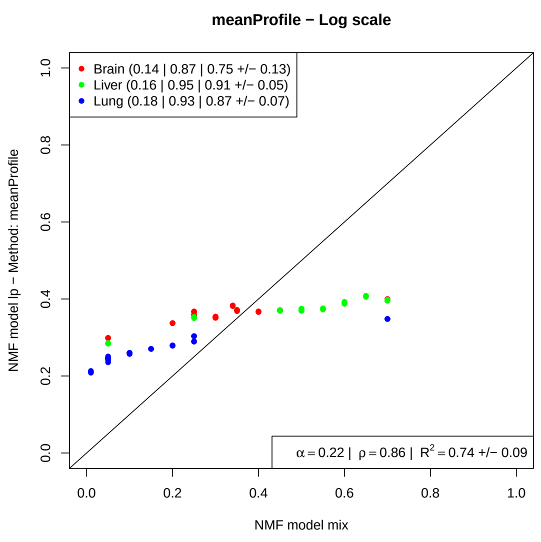
For example, supposing the markers we selected above were *a priori* known marker genes, for the cell types that compose the samples. A simple approach consists in using their mean expression profile, computed within their respective cell type, as implemented in the ged algorithm `'meanProfile'`. This is *part* of the approach used by Kuhn et al. (2011), who used these average profiles as proxies for the proportions, and plugged them into a linear regression framework to detect cell-specific differential expression. See Section 5 for how to apply an alternative approach, based on a semi-supervised NMF algorithm.

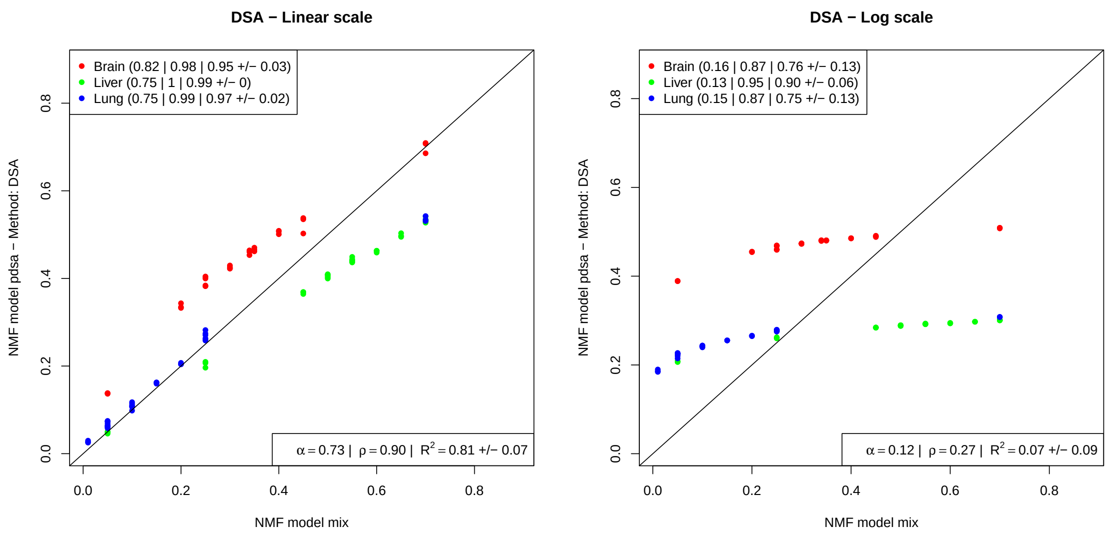
In this context, very recent work from Zhong et al. (2013) suggests that deconvolution of gene expression data should be conducted in linear space, rather than on log-transformed expression values, which are usually used in more classical analyses. They proposed a more sophisticated computation of the proportions – and cell-specific signatures – from average marker expression profiles in mixed samples, which they named *Digital Sorting Algorithm*, which is implemented in the ged algorithm `'DSA'`. The following sample code illustrates the difference in proportion estimates, when computed using plain average expression of markers and DSA, either in linear or log-space. Note that, in this particular example, because we are interested in the proportions only, we can speed up the DSA estimation by limiting the target expression matrix to the selected marker genes (`target = mix[sel]`).

```
# check if data is in log scale
is_logscale(mix)

## [1] TRUE


# compute mean expression profiles within each cell type
p <- ged(expb(mix, 2), sel, "meanProfile")
# plot against known proportions (p is by default not scaled)
profplot(mix, p, scale = TRUE, main = "meanProfile - Linear scale")

# compute mean expression profiles within each cell type
```

```
lp <- ged(mix, sel, "meanProfile")
# plot against known proportions (p is by default not scaled)
profplot(mix, lp, scale = TRUE, main = "meanProfile - Log scale")

# compute proportions using DSA methods [Zhong et al. (2013)]
pdsa <- ged(mix[sel], sel, "DSA", verbose = TRUE)

## Using ged algorithm: "DSA"

## Note: method with signature 'MatrixData#MarkerList#ANY' chosen for function 'nmf',
##  target signature 'ExpressionMix#MarkerList#character'.
##  "ExpressionSet#ANY#ANY" would also be valid

##    Estimating basis and mixture coefficients matrices from marker features [DSA]
##    Using 14/14 markers to estimate cell proportions:
##  Brain Liver  Lung
##      5     5     4
##     Checking data scale ...     NOTE [log]
##     Converting data to linear scale ...     OK [base: 2]
##     Computing proportions using DSA method ...     OK
##     Estimating basis matrix from mixture coefficients [qprog]
##     Not using any marker constraints
##  Timing:
##    user  system elapsed
##   0.864   0.008   0.875
##  GED final wrap up ...  OK

profplot(mix, pdsa, main = "DSA - Linear scale")
pdsa <- ged(mix[sel], sel, "DSA", log = FALSE)
profplot(mix, pdsa, main = "DSA - Log scale")
```







4 Estimating differential cell-specific expression

One is generally eventually interested in estimating cell-specific expression, and, in particular, detecting genes that are differentially expressed between two groups of samples. Being able to perform such analysis at the cell type level facilitates interpretation, i.e. the traduction of a list of genes into meaningfull biological hypotheses.

4.1 From measured proportions: *csSAM*

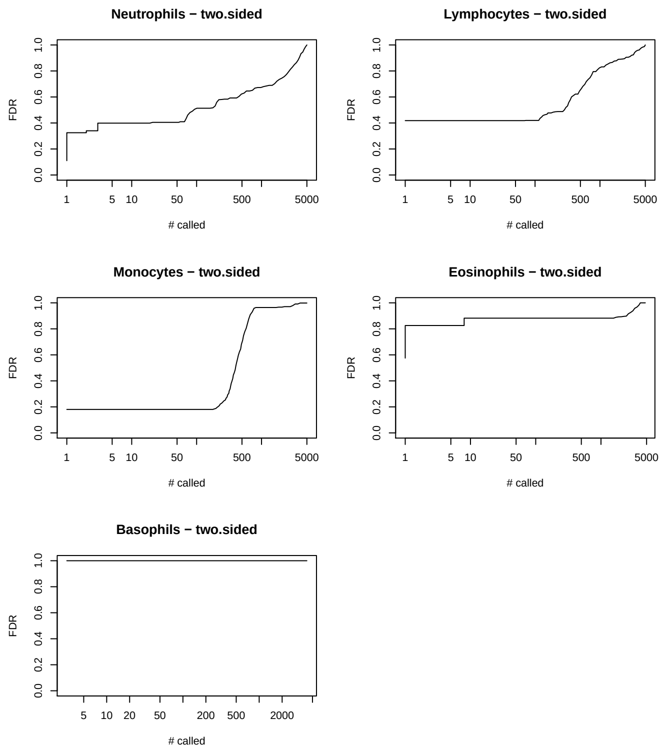
The *csSAM* algorithm (Shen-Orr et al. 2010) defines an extension of the popular *SAM* algorithm (Tusher et al. 2001), which estimates differential expression between cell types. This algorithm is implemented in the *csSAM* package[10] (Shen-Orr et al. 2013). The *CellMix* package provides access to this method within its comprehensive framework. In particular, some *csSAM*'s utility functions have been rewritten in *C++*, which results in a large performance gain. We illustrate its use on the ACR dataset, reproducing the analysis from the original `csSAM` paper:

```
# take the 5000 most variable genes
s <- esApply(acr, 1L, sd, na.rm = TRUE)
i <- order(s, decreasing = TRUE)[1:5000]
# fit csSAM for the groups defined in covariate 'Status'
rescs <- ged(acr[i, ], coef(acr), method = "csSAM", data = acr$Status, nperms = 200,
    verbose = TRUE)

## Using ged algorithm: "csSAM"
## Groups: Acute Rejection=15L | Stable=9L
## Fitting cell-specific linear model ...  OK
## Computing csSAM model statistics ...  OK
## Computing fdr using 200 permutations ...  OK
## Finding signature genes ...  OK
## Timing:
##    user  system elapsed
## 44.734   3.732  48.579
## GED final wrap up ...  OK
```

[10]http://cran.r-project.org/package=csSAM

The result may be plotted using the function `csplot`, which plots the number of genes that have a false discovery rate below a given threshold, within each cell type separately. This function is generic and may be called for any result from the `ged` function, as long as the deconvolution algorithm is suitable for such computation.

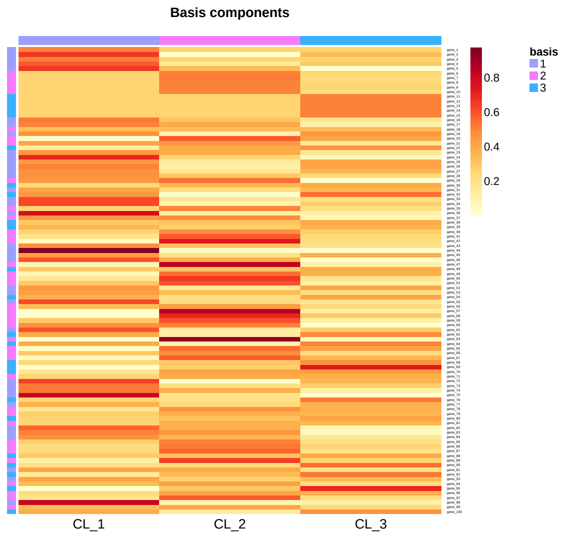
```
csplot(rescs)
```

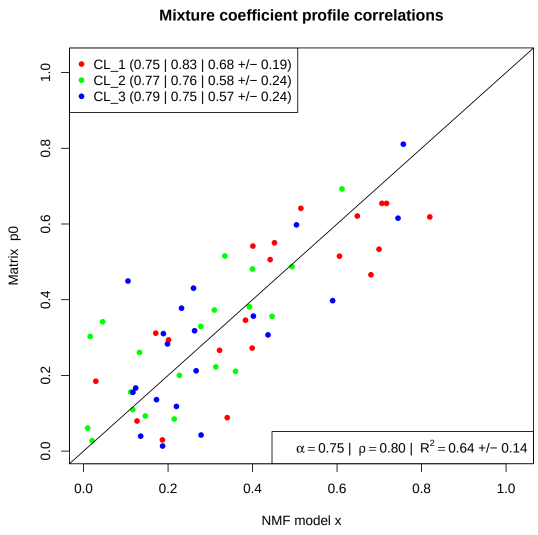


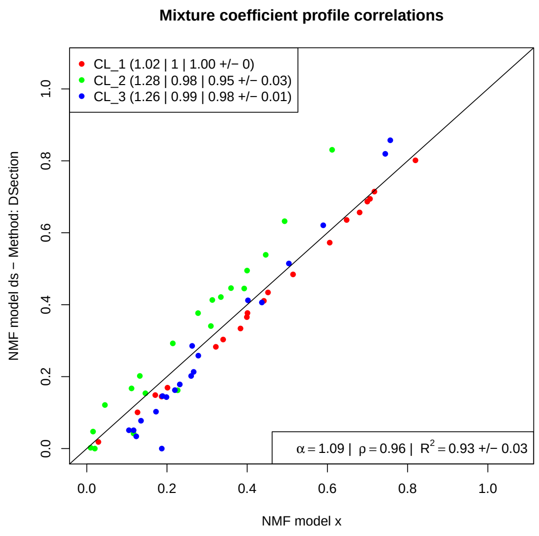
4.2 From proportion priors: *DSection*

Another algorithm, similar to *csSAM* in its input and output, is the `DSection` algorithm. This algorithm is a Monte-Carlo-Markov-Chain (MCMC) approach, which models uncertainity in the proportions measurments, considering them as Bayesian priors, and computes posterior probabilities and estimates for cell type-specific signatures and proportions (Erkkila et al. 2010). It can perform the deconvolution in multiple groups of samples, possibly more than two. We illustrate its use on a small simulated dataset, as it is very computationally intensive: for large datasets, one would need several hundreds/thousands of samples, as well as a burnin period, to ensure the convergence of the Markov chain. Note that *DSection* is expected to perform very well on such simulated toy data, because they are generated using noise assumptions that are close to its underlying statistical model.

```
# generate random data: 3 cell types, 20 samples, 100 genes and 5 markers
# per cell type
x <- rmix(3, 100, 20, markers = 5)
x

## ExpressionMix (storageMode: lockedEnvironment)
## assayData: 100 features, 20 samples
##   element names: exprs
## protocolData: none
## phenoData: none
## featureData: none
## experimentData: use 'experimentData(object)'
## Annotation:
## Composition: 'CL_1', 'CL_2', 'CL_3' (3 total)

# show true basis signatures (the markers clearly appear)
basismap(x, Rowv = NA)

# add noise to the proportions
p0 <- abs(coef(x) + rmatrix(coef(x), dist = rnorm, sd = 0.2))
# rescale into proportion (columns sum up to one)
p0 <- scoef(p0)
# see how noisy they got
profplot(x, p0)

# fit DSection MCMC model
ds <- ged(x, p0, "DSection", maxIter = 20, verbose = TRUE)

##  Using ged algorithm: "DSection"

## Loading required package:  RcppOctave

##  Timing:
##    user  system elapsed
##  17.505   0.132  17.784
##  GED final wrap up ...  OK

# check reduction of noise on proportions
profplot(x, ds)
```







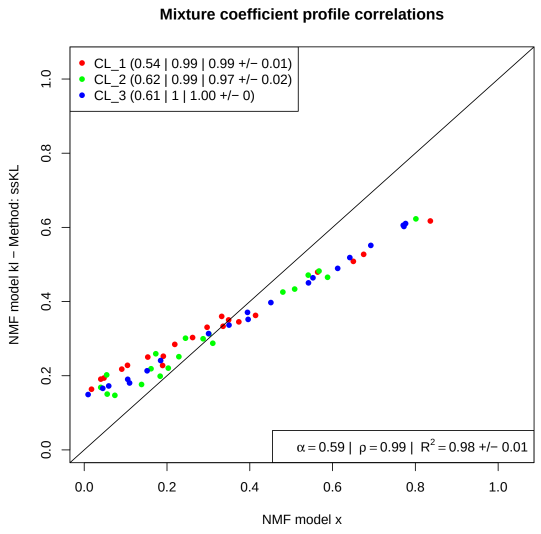
5 Complete deconvolution using marker genes

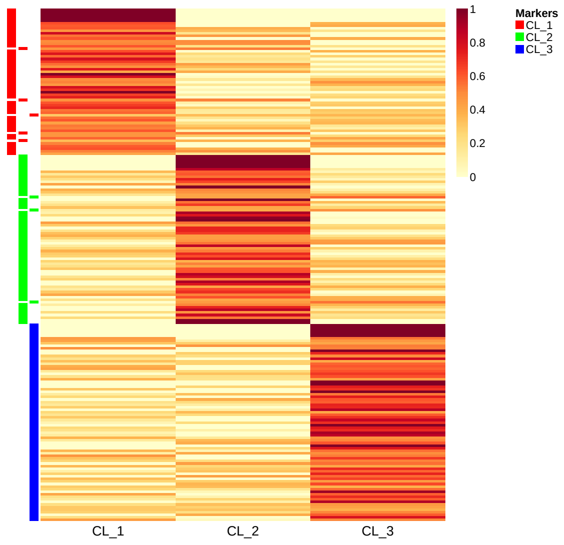
5.1 A priori: enforce marker expression patterns

In previous work, we proposed a simple strategy to modify some common NMF algorithms, in order to improve the estimation of cell-specific signatures, by enforcing the expected expression pattern of known marker genes (Gaujoux et al. 2011). The strategy was shown to greatly improve the ability of those algorithms to extract meaningful signatures. The algorithms 'ssKL', 'ssFrobenius' implement this strategy, for the Kullback-Leibler divergence and the euclidean distance respectively. We illustrate their use on a sythetic dataset.

```r
# generate random data with 5 markers per cell type
x <- rmix(3, 200, 20, markers = 5)
m <- getMarkers(x)

# deconvolve using KL-divergence metric
kl <- ged(x, m, "ssKL", log = FALSE, rng = 1234, nrun = 10)
```

```
# plot against known proportions
profplot(x, kl)
# check consistency of most expressing cell types in known basis
# signatures
basismarkermap(basis(x), kl)
# correlation with known signatures
basiscor(x, kl)

##          CL_1     CL_2     CL_3
## CL_1  0.90603  0.09381 -0.07245
## CL_2 -0.10075  0.94935 -0.12459
## CL_3 -0.01925 -0.26083  0.96573
```



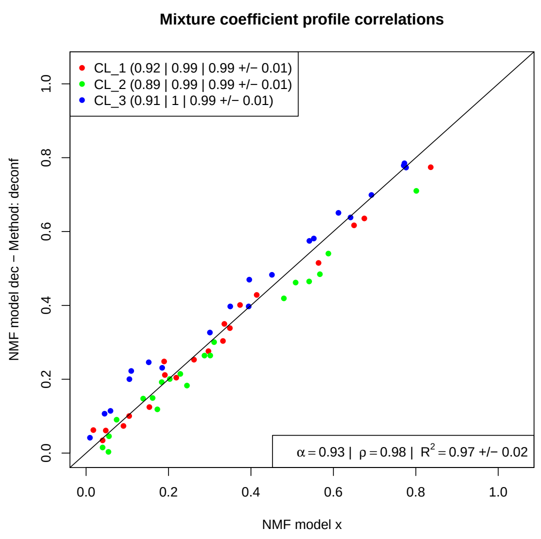


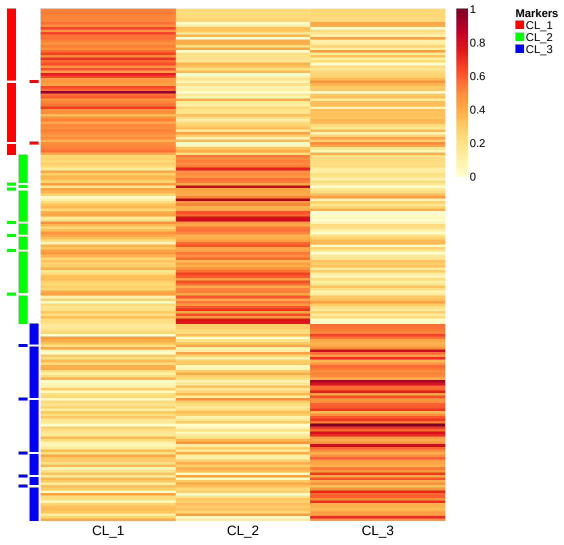
Note that experiments showed, however, that enforcing markers could introduce a bias in the proportion estimates – which can be seen on the scatterplot above. Further investigation would be required in order to design a strategy that enforces markers without introducing such bias.

5.2 A posteriori: assign signatures to cell types

Another NMF based method that uses marker genes is the *deconf* algorithm (Repsilber et al. 2010). This algorithm use markers *a posteriori*, to assign the estimated signatures each cell type. Its original implementation is very slow. The *CellMix* package implements a faster version of this algorithm, which uses the fast combinatorial nonnegative least-squares algorithm (FCNNLS) from Van Benthem et al. (2004), as provided by the *NMF* package package. Similarly to *DSection*, the algorithm is expected to perform very well on the toy simulated data.

```
# deconvolve using KL divergence metric
dec <- ged(x, m, "deconf", rng = 1234, nrun = 10)

# plot against known proportions
profplot(x, dec)
# check consistency of most expressing cell types in known signatures
basismarkermap(basis(x), dec)
# correlation with known signatures
basiscor(x, dec)

##          CL_1     CL_2    CL_3
## CL_1  0.98206  0.09458  0.1125
## CL_2  0.26516  0.99578 -0.1253
## CL_3 -0.06807 -0.09600  0.9986
```





6 Conclusion

The *CellMix* package provides a comprehensive framework for researchers working with gene expression deconvolution, whether to apply algorithms or develop new methods. Its objective is to gather in one place as many relevant components as possible, so as to facilitate gene expression deconvolution, and extract as much as possible from gene expression data, despite sample heterogeneity. We plan to develop further the package, along with our own research in the area, and contribute towards the improvment and the popularisation of deconvolution techniques.

7 R session details

- R version 3.0.0 (2013-04-03), `i686-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=en_US.UTF-8`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=C`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, graphics, grDevices, grid, methods, parallel, stats, utils
- Other packages: annotate 1.38.0, AnnotationDbi 1.22.3, Biobase 2.20.0, BiocGenerics 0.6.0, CellMix 1.5.1, corpcor 1.6.5, csSAM 1.2.4, DBI 0.2-6, digest 0.6.3, doParallel 1.0.1, foreach 1.4.0, graph 1.38.0, GSEABase 1.22.0, hgu133a.db 2.9.0, hgu133b.db 2.9.0, hgu133plus2.db 2.9.0, iterators 1.0.6, knitr 1.2, limSolve 1.5.4, matrixStats 0.6.2, NMF 0.16.2, org.Hs.eg.db 2.9.0, pkgmaker 0.15.5, RColorBrewer 1.0-5, Rcpp 0.10.3, RcppOctave 0.9, registry 0.2, rngtools 1.2, RSQLite 0.11.3, stringr 0.6.2
- Loaded via a namespace (and not attached): beeswarm 0.1.5, BiocInstaller 1.10.1, codetools 0.2-8, colorspace 1.2-2, compiler 3.0.0, evaluate 0.4.3, formatR 0.7, genefilter 1.42.0, gridBase 0.4-6, gtools 2.7.1, IRanges 1.18.0, lpSolve 5.6.7, MASS 7.3-26, preprocessCore 1.22.0, quadprog 1.5-5, R.methodsS3 1.4.2, splines 3.0.0, stats4 3.0.0, survival 2.37-4, tools 3.0.0, XML 3.96-1.1, xtable 1.7-1

References

[1] Alexander R Abbas, Kristen Wolslegel, Dhaya Seshasayee, Zora Modrusan, and Hilary F Clark. “Deconvolution of blood microarray data identifies cellular activation patterns in systemic lupus erythematosus.” In: *PloS one* 4.7 (Jan. 2009), e6098. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0006098. URL: http://www.ncbi.nlm.nih.gov/pubmed/19568420 (see pp. 3, 4, 8, 10).

[2] Tanya Barrett, Dennis B Troup, Stephen E Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F Kim, Maxim Tomashevsky, Kimberly a Marshall, Katherine H Phillippy, Patti M Sherman, Rolf N Muertter, Michelle Holko, Oluwabukunmi Ayanbule, Andrey Yefanov, and Alexandra Soboleva. “NCBI GEO: archive for functional genomics data sets–10 years on.” In: *Nucleic acids research* 39.November 2010 (2010), pp. 1005–1010. ISSN: 1362-4962. DOI: 10.1093/nar/gkq1184. URL: http://www.ncbi.nlm.nih.gov/pubmed/21097893 (see p. 3).

[3] Timo Erkkila, Saara Lehmusvaara, Pekka Ruusuvuori, Tapio Visakorpi, Ilya Shmulevich, and Harri Lahdesmaki. “Probabilistic analysis of gene expression measurements from heterogeneous tissues.” In: *Bioinformatics (Oxford, England)* 26.20 (2010), pp. 2571–7. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btq406. URL: http://www.ncbi.nlm.nih.gov/pubmed/20631160http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2951082\&tool=pmcentrez\&rendertype=abstract (see p. 14).

[4] Renaud Gaujoux and Cathal Seoighe. “A flexible R package for nonnegative matrix factorization”. In: *BMC Bioinformatics* 11.1 (2010), p. 367. ISSN: 1471-2105. DOI: 10.1186/1471-2105-11-367. URL: http://www.biomedcentral.com/1471-2105/11/367 (see p. 2).

[5] Renaud Gaujoux and Cathal Seoighe. “Semi-supervised Nonnegative Matrix Factorization for gene expression deconvolution: A case study.” In: *Infection, genetics and evolution : journal of molecular epidemiology and evolutionary genetics in infectious diseases* (2011). ISSN: 1567-7257. DOI: 10.1016/j.meegid.2011.08.014. URL: http://www.ncbi.nlm.nih.gov/pubmed/21930246 (see p. 16).

[6] Robert C Gentleman et al. “Bioconductor: open software development for computational biology and bioinformatics.” In: *Genome biology* 5.10 (2004), R80. ISSN: 1465-6914. DOI: 10.1186/gb-2004-5-10-r80. URL: http://www.ncbi.nlm.nih.gov/pubmed/15461798 (see p. 1).

[7] Ting Gong, Nicole Hartmann, Isaac S Kohane, Volker Brinkmann, Frank Staedtler, Martin Letzkus, Sandrine Bongiovanni, and Joseph D Szustakowski. “Optimal deconvolution of transcriptional profiling data using quadratic programming with application to complex clinical blood samples.” In: *PloS one* 6.11 (Jan. 2011), e27156. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0027156. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3217948\&tool=pmcentrez\&rendertype=abstract (see p. 3).

[8] Alexandre Kuhn, Doris Thu, Henry J Waldvogel, Richard L M Faull, and Ruth Luthi-Carter. “Population-specific expression analysis (PSEA) reveals molecular changes in diseased brain.” In: *Nature methods* 8.11 (2011), pp. 945–7. ISSN: 1548-7105. DOI: 10.1038/nmeth.1710. URL: http://www.ncbi.nlm.nih.gov/pubmed/21983921 (see p. 11).

[9] D D Lee and H S Seung. “Learning the parts of objects by non-negative matrix factorization.” In: *Nature* 401.6755 (1999), pp. 788–91. ISSN: 0028-0836. DOI: 10.1038/44565. URL: http://www.ncbi.nlm.nih.gov/pubmed/10548103 (see p. 2).

[10] Pentti Paatero and Unto Tapper. “Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values”. In: *Environmetrics* 5.2 (1994), pp. 111–126. DOI: 10.1002/env.3170050203. URL: http://www3.interscience.wiley.com/cgi-bin/abstract/113468839/ABSTRACT (see p. 2).

[11] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing. Vienna, Austria, 2011. URL: http://www.r-project.org (see p. 1).

[12] Dirk Repsilber, Sabine Kern, Anna Telaar, Gerhard Walzl, Gillian F Black, Joachim Selbig, Shreemanta K Parida, Stefan H E Kaufmann, and Marc Jacobsen. “Biomarker discovery in heterogeneous tissue samples -taking the in-silico deconfounding approach.” In: *BMC bioinformatics* 11 (2010), p. 27. ISSN: 1471-2105. DOI: 10.1186/1471-2105-11-27. URL: http://www.ncbi.nlm.nih.gov/pubmed/20070912 (see p. 17).

[13] Shai Shen-Orr, Rob Tibshirani, Narasimhan Balasubramanian, and David Wang. *csSAM: csSAM - cell-specific Significance Analysis of Microarrays*. R package version 1.2.4/r3. 2013. URL: http://R-Forge.R-project.org/projects/cellmix/ (see p. 13).

[14] Shai S Shen-Orr, Robert Tibshirani, Purvesh Khatri, Dale L Bodian, Frank Staedtler, Nicholas M Perry, Trevor Hastie, Minnie M Sarwal, Mark M Davis, and Atul J Butte. “Cell type-specific gene expression differences in complex tissues.” In: *Nature methods* 7.4 (Apr. 2010), pp. 287–9. ISSN: 1548-7105. DOI: 10.1038/nmeth.1439. URL: http://www.ncbi.nlm.nih.gov/pubmed/20208531 (see pp. 3, 6, 13).

[15] V G Tusher, Robert Tibshirani, and G Chu. “Significance analysis of microarrays applied to the ionizing radiation response.” In: *Proceedings of the National Academy of Sciences of the United States of America* 98.9 (2001), pp. 5116–21. ISSN: 0027-8424. DOI: 10.1073/pnas.091062498. URL: http://www.ncbi.nlm.nih.gov/pubmed/11309499 (see p. 13).

[16] Mark H. Van Benthem and Michael R. Keenan. “Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems”. In: *Journal of Chemometrics* 18.10 (2004), pp. 441–450. ISSN: 0886-9383. DOI: 10.1002/cem.889. URL: http://doi.wiley.com/10.1002/cem.889 (see p. 17).

[17] Yi Zhong, Yin-Wooi Wan, Kaifang Pang, Lionel Ml Chow, and Zhandong Liu. “Digital sorting of complex tissues for cell type-specific gene expression profiles”. In: *BMC Bioinformatics* 14.1 (2013), p. 89. ISSN: 1471-2105. DOI: 10.1186/1471-2105-14-89. URL: http://www.biomedcentral.com/1471-2105/14/89 (see p. 11).