PSEA: Expression deconvolution of neural RNA mixtures (replication of results from the Nature Methods paper)

Alexandre Kuhn[1]

November 1, 2022

Contents

1 Introduction

This document shows how we applied PSEA to deconvolute the set of artificial RNA mixtures presented in [1].

Briefly, this dataset was generated and analyzed as follows: We obtained RNA samples from 4 individual neural cell types (neurons, astrocytes, oligodendrocytes and microglia). We generated 10 mixed RNA samples (with varying mixing proportions) and obtained gene expression profiles for the 10 mixed samples as well as for the 4 pure cell types. We then deconvoluted mixed samples and compared the predicted cell-type specific expression with the expression obtained from pure samples. The whole dataset (10 mixed samples and 4 replicates for each of the 4 cell types) is deposited in GEO (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE19380). More details can be found in the Supplementary material to [1] at https://www.nature.com/article-assets/npg/nmeth/journal/v8/n11/extref/nmeth.1710-S1.pdf.

2 Deconvolution of RNA mixtures

Here we explain how we applied PSEA to predict cell type-specific expression from the RNA mixtures and replicate some of the Supplementary Figures presented in [1].

To keep this vignette self-contained, we will start from the normalized expression data (provided with the PSEA package, as indicated below). However, you can download the raw data from GEO and generate the normalized expression data yourself, as explained at the beginning of this protocol.

[1] alexandre.m.kuhn@gmail.com

We start by loading the required libraries.

```
> library(PSEA)
> library(GEOquery)
> library(affy)
```

If you want to start from the raw microarray data, you can download them from GEO and retrieve the corresponding sample information.

```
> dataset<-getGEO(GEO="GSE19380",destdir=".")
> information<-pData(phenoData(dataset[["GSE19380_series_matrix.txt.gz"]]))
> sample_IDs<-as.character(information[,"geo_accession"])
> datafiles<-sapply(sample_IDs,function(x){rownames(getGEOSuppFiles(x))})
```

Note that if you have already downloaded the data as just shown and you are re-running the protocol, you can avoid downloading the data again and use the corresponding compressed file (that was stored locally after the initial download).

```
> dataset<-getGEO(GEO="GSE19380",filename="GSE19380_series_matrix.txt.gz")
> dataset<-list("GSE19380_series_matrix.txt.gz"=dataset)
> information<-pData(phenoData(dataset[["GSE19380_series_matrix.txt.gz"]]))
> sample_IDs<-as.character(information[,"geo_accession"])
> datafiles<-file.path(sample_IDs,paste(sample_IDs,".CEL.gz",sep=""))
```

To start the analysis from the raw microarray data (.CEL files), load them into R and perform normalization.

```
> raw_data<-ReadAffy(filenames=datafiles,compress=TRUE)
> expression_GSE19380<-2^exprs(rma(raw_data))
```

As already mentioned above, you can also run this protcol without downloading data as we provide the corresponding normalized expression data with the PSEA package. We will now load it and proceed with PSEA deconvolution (so you can skip this step if you want to use data you have just downloaded from GEO instead).

```
> data(expression_GSE19380)
```

We start by removing the control probesets.

```
> expression<-expression_GSE19380[1:31042,]
```

We then define marker probe sets (as per Supplementary Table 2 in [1])

```
> neuron_probesets<-list(c("1370058_at","1370059_at"),"1387073_at","1367845_at")
> astro_probesets<-list("1372190_at","1386903_at",c("1375120_at","1375183_at","1385923_at"))
> oligo_probesets<-list("1398257_at","1368861_a_at",c("1368263_a_at","1370434_a_at","1370500_a_at"))
```

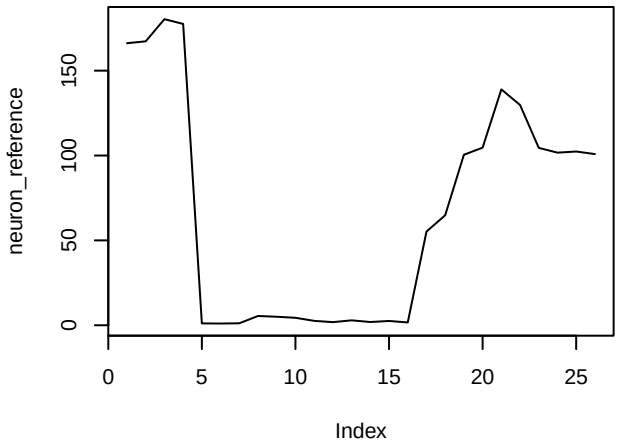
and generate reference signals. We normalize the signals using mixed samples only as we restrict the use of pure samples to the validation of deconvoluted expression. Mixed samples correspond to column 17 to 24 of the expression matrix, as indicated in the corresponding sample information (`information[,"characteristics_ch1.1"]` or `information[,"description"]`).

```
> mixedsamples<-c(17:24)
> neuron_reference<-marker(expression,neuron_probesets,sampleSubset=mixedsamples,targetMean=100)
```

```
> astro_reference<-marker(expression,astro_probesets,sampleSubset=mixedsamples,targetMean=100)
> oligo_reference<-marker(expression,oligo_probesets,sampleSubset=mixedsamples,targetMean=100)
```

We can plot the neuronal reference signal across all samples (replicates Supplementary Figure 3a, middle in [1]).

```
> par(cex=0.7)
> plot(neuron_reference,type="l")
```

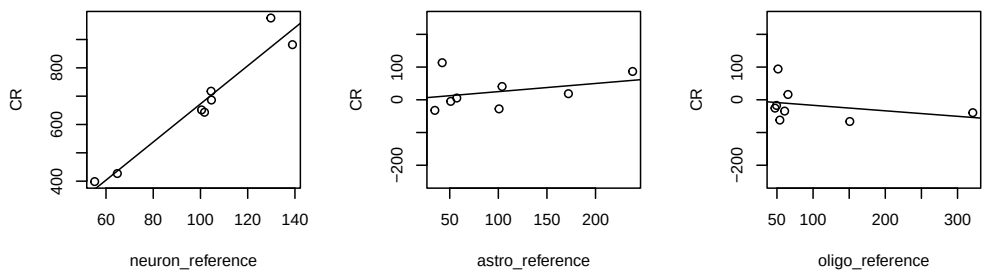


We fit the signal measured by probeset 1367660_at in the mixed samples with an expression model including all 3 populations (replicates Supplementary Figure 4a).

```
> model1<-lm(expression["1367660_at",]~neuron_reference+astro_reference+
+ oligo_reference,subset=mixedsamples)
```

We can use component-plus-residual plots to visualize the dependence of expression on the 4 reference signals.

```
> par(mfrow=c(1,3),cex=0.7)
> crplot(model1,"neuron_reference",newplot=FALSE)
> crplot(model1,"astro_reference",newplot=FALSE,ylim=c(-250,250))
> crplot(model1,"oligo_reference",newplot=FALSE,ylim=c(-250,250))
```



We can inspect the fitted expression model and in particular the p-values.

```
> summary(model1)

Call:
lm(formula = expression["1367660_at", ] ~ neuron_reference +
    astro_reference + oligo_reference, subset = mixedsamples)

Residuals:
GSM480959.CEL.gz GSM480960.CEL.gz GSM480961.CEL.gz GSM480962.CEL.gz
          27.181           -8.896          -24.306          -17.557
GSM480963.CEL.gz GSM480964.CEL.gz GSM480965.CEL.gz GSM480966.CEL.gz
         -52.938          102.794           14.659          -40.937

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       -8.2476   125.0569  -0.066  0.95058
neuron_reference   6.7273     0.9750   6.900  0.00231 **
astro_reference    0.2494     0.3872   0.644  0.55457
oligo_reference   -0.1683     0.2579  -0.653  0.54955
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 65.15 on 4 degrees of freedom
Multiple R-squared:  0.9333,        Adjusted R-squared:  0.8833
F-statistic: 18.67 on 3 and 4 DF,  p-value: 0.008147
```

We now deconvolute the entire expression profile (i.e. all probesets) obtained for the mixed samples. We start by defining the full model matrix

```
> model_matrix<-cbind(intercept=1,neuron_reference,astro_reference,oligo_reference)
```

and specify the subset of models under consideration (as specified in Supplementary Table 3 in [1])

```
> model_subset<-em_quantvg(c(2,3,4), tnv=3, ng=1)
```

We fit each probeset with all models in the subset and select the best model.

```
> models<-lmfitst(t(expression), model_matrix, model_subset, subset=mixedsamples)
```

Finally we extract coefficients, p-values and adjusted R^2 for the selected models

```
> regressor_names<-as.character(1:4)
> coefficients<-coefmat(models[[2]], regressor_names)
> pvalues<-pvalmat(models[[2]], regressor_names)
> models_summary<-lapply(models[[2]], summary)
> adjusted_R2<-slt(models_summary, 'adj.r.squared')
```

and filter satisfactory expression models

```
> negativecoefficient<-apply(coefficients[,-1]<0 & pvalues[,-1]<0.05,1,function(x){any(x,na.rm=TRUE)})
> average_expression<-apply(expression[,mixedsamples], 1, mean)
> filter<-!negativecoefficient & (coefficients[,1] / average_expression) < 0.5 & adjusted_R2 > 0.6
```

Here is the number of filtered probesets (see Supplementary Table 9 in [1])

Number of probesets with non-negative coefficients:

```
> sum(!negativecoefficient)

[1] 23252
```

Number of probesets with relative intercept < 0.5:

```
> sum(coefficients[,1] / average_expression < 0.5)

[1] 4619
```

Number of probesets with adjusted $R^2 > 0.6$:

```
> sum(adjusted_R2 > 0.6)

[1] 15309
```

Number of probesets passing all 3 criteria:

```
> sum(filter)

[1] 4039
```

We can for instance inspect the expression model for probeset 1370431_at (replicates Supplementary Figure 4d, middle panel).

```
> selectedpsname<-"1370431_at"
> selectedps<-which(rownames(expression)==selectedpsname)
```

The neuron-specific expression for 1370431_at is

```
> coefficients[selectedps,2]

  coef.2
2.322765
```

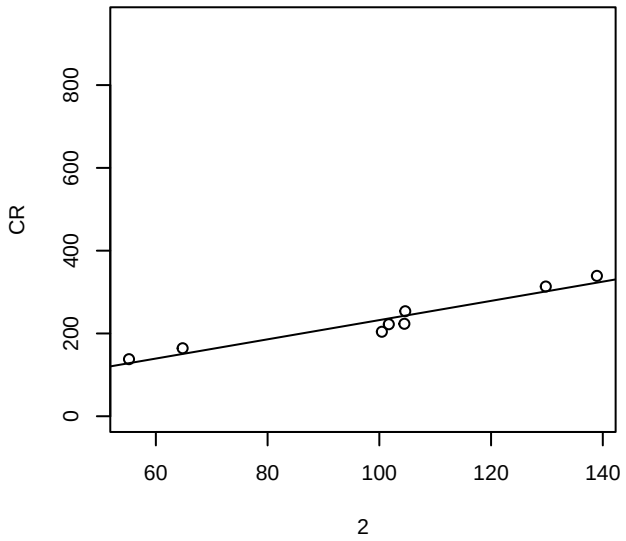
and the corresponding p-value is

```
> pvalues[selectedps,2]

    pvalue.2
9.917193e-05
```

The dependence on the neuronal reference signal is visualized as follows.

```
> crplot(models[[2]][[selectedps]],"2",ylim=c(0,950))
```



3 Session Information

The version number of R and packages loaded for generating the vignette were:

```
R version 4.2.1 Patched (2022-07-09 r82577)
Platform: x86_64-apple-darwin17.0 (64-bit)
Running under: macOS Big Sur ... 10.16

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_GB/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] affy_1.76.0         GEOquery_2.66.0     Biobase_2.58.0
[4] BiocGenerics_0.44.0 PSEA_1.32.0

loaded via a namespace (and not attached):
 [1] pillar_1.8.1          compiler_4.2.1        BiocManager_1.30.19
 [4] zlibbioc_1.44.0       tools_4.2.1           digest_0.6.30
```

```
 [7] preprocessCore_1.60.0 evaluate_0.17         lifecycle_1.0.3
[10] tibble_3.1.8          pkgconfig_2.0.3      rlang_1.0.6
[13] cli_3.4.1             DBI_1.1.3            yaml_2.3.6
[16] xfun_0.34             fastmap_1.1.0        dplyr_1.0.10
[19] knitr_1.40            xml2_1.3.3           generics_0.1.3
[22] vctrs_0.5.0           hms_1.1.2            tidyselect_1.2.0
[25] glue_1.6.2            data.table_1.14.4    R6_2.5.1
[28] fansi_1.0.3           rmarkdown_2.17       limma_3.54.0
[31] readr_2.1.3           tzdb_0.3.0           tidyr_1.2.1
[34] purrr_0.3.5           magrittr_2.0.3       htmltools_0.5.3
[37] ellipsis_0.3.2        MASS_7.3-58.1        assertthat_0.2.1
[40] BiocStyle_2.26.0      utf8_1.2.2           affyio_1.68.0
```

References

[1] Alexandre Kuhn, Doris Thu, Henry J Waldvogel, Richard LM Faull, and Ruth Luthi-Carter. Population-specific expression analysis (psea) reveals molecular changes in diseased brain. *Nat. Methods*, 9(8):945–947, 2011. URL: http://www.nature.com/nmeth/journal/v8/n11/full/nmeth.1710.html, doi:10.1038/nmeth.1710.