PSEA: Population-Specific Expression Analysis

Alexandre Kuhn[1]

May 9, 2023

Contents

1 Introduction

The characterization of molecular changes in diseased tissues can provide crucial information about pathophysiological mechanisms and is important for the development of targeted drugs and therapies. However, many disease processes are accompanied by changes of cell populations due to cell migration, proliferation or death. Identification of key molecular events can thus be overshadowed by confounding changes in tissue composition.

To address the issue of confounding between cell population composition and cellular expression changes, we developed Population-Specific Expression Analysis (PSEA) [1, 2]. This method works by exploiting linear regression modeling of queried expression levels to the abundance of each cell population. Since a direct measure of population size is often unobtainable (e.g. from human clinical or autopsy samples), PSEA instead tracks relative cell population size via levels of mRNAs expressed in a single population only. Thus, a reference measure is constructed for each cell population by averaging expression data for cell-type-specific mRNAs derived from the same expression profile.

Here we will demonstrate some of the functionalities in the PSEA package. We will first generate reference signals and deconvolve individual transcripts to illustrate the method. We will then show how to apply PSEA to entire expression profiles. Let us start by loading the package

```
> library(PSEA)
```

We have included expression data obtained from brain samples of 41 individuals as well as their phenotypes, i.e. control and Huntington's disease (HD) (the full data is deposited at http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE3790)

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

```
> data(example)
```

The example data contains the variable `expression`, a matrix with the expression levels of 23 transcripts and the variable `groups`, a vector with phenotypic information encoded as 0 and 1 (indicating control and disease, respectively). Detailed information about the data is provided in the corresponding manual pages (see `?expression` and `?groups`).

```
> expression[1:5,1:3]
```

```
            GSM86787.cel.gz GSM86789.cel.gz GSM86791.cel.gz
200850_s_at       4654.3390       6093.9529       2914.7702
201313_at          953.4176       1649.3578       2447.6300
201667_at         4529.0839       5857.4259       2672.2444
202429_s_at       1729.1979       3641.1434       5706.2037
203416_at          432.0814        682.8519        393.8298
```

```
> groups
```

```
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
[39] 1 1 1
```

2 Reference signals

We previously found that neurons, astrocytes, oligodendrocytes and microglia were the four neural cell populations that mostly contributed expression in these brain samples [1]. For each cell population, we then identified several probesets corresponding to mRNAs expressed in that cell type only, that can be used to monitor the abundance of the cell population. For neurons, we selected the following probe sets (see Supplementary Table 5 in [1])

```
> neuron_probesets <- list(c("221805_at", "221801_x_at", "221916_at"), "201313_at",
+                                "210040_at", "205737_at", "210432_s_at")
```

Note that they are assigned to a list where each item can contain one or more probesets measuring expression of the same gene. Here, the first three probesets measure expression of NEFL, and four additional genes are measured by one probeset each. The list structure allows us to average expression over probesets measuring the same transcript (for instance the first three probesets that measure NEFL transcripts) before averaging over several genes. This is what is achieved by the function `marker`, resulting in a neuronal "reference signal"

```
> neuron_reference <- marker(expression, neuron_probesets)
```

We also define marker probesets and calculate reference signals for the three other cell populations

```
> astro_probesets <- list("203540_at", c("210068_s_at", "210906_x_at"), "201667_at")
> astro_reference <- marker(expression, astro_probesets)
> oligo_probesets <- list(c("211836_s_at", "214650_x_at"), "216617_s_at", "207659_s_at",
+                         c("207323_s_at", "209072_at"))
> oligo_reference <- marker(expression, oligo_probesets)
> micro_probesets <- list("204192_at", "203416_at")
> micro_reference <- marker(expression, micro_probesets)
```

In addition, we will need a group indicator variable that codes controls as 0s and HD subjects as 1s. It is included in the example data, as explained above

```
> groups

 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
[39] 1 1 1
```

The indicator variable is used to generate an interaction regressor that will allow us to test for differences in cell population-specific expression across groups (HD versus control). For neurons, the interaction regressor is defined as

```
> neuron_difference <- groups * neuron_reference
```

We create similar interaction regressors for the other three populations

```
> astro_difference <- groups * astro_reference
> oligo_difference <- groups * oligo_reference
> micro_difference <- groups * micro_reference
```

3 Principle of PSEA

To illustrate how PSEA works, we will deconvolve the expression of Calcineurin A (or PPP3CA, measured by probeset 202429_s_at), a gene whose product was previously shown to be decreased in the striatum of HD patients. In PSEA, we use linear regression and model the expression of Calcineurin A in the control samples as a linear combination of the four reference signals

```
> model1 <- lm(expression["202429_s_at",] ~ neuron_reference + astro_reference +
+                 oligo_reference + micro_reference, subset=which(groups==0))
```

The dependence of expression on each reference signal can be visualized as follows

```
> par(mfrow=c(2,2), mex=0.8)
> crplot(model1, "neuron_reference", newplot=FALSE)
> crplot(model1, "astro_reference", newplot=FALSE)
> crplot(model1, "oligo_reference", newplot=FALSE)
> crplot(model1, "micro_reference", newplot=FALSE)
```

The plots show the strong and specific dependence of Calcineurin A expression on the neuronal reference signal (Figure 1). The fit summary provides further useful information on the model

```
> summary(model1)

Call:
lm(formula = expression["202429_s_at", ] ~ neuron_reference +
    astro_reference + oligo_reference + micro_reference, subset = which(groups ==
    0))

Residuals:
   Min     1Q Median     3Q    Max
-713.9 -364.4 -128.0  351.6 1019.3
```

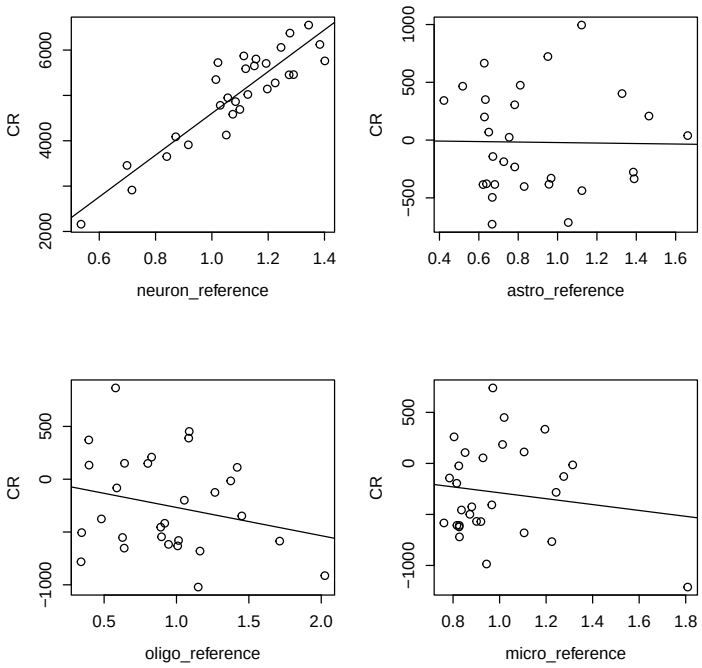

Figure 1: Component-plus-residual plots showing deconvolved neuronal, astrocytic, oligodendrocytic and microglial expression of Calcineurin A in control samples.

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)       103.60     918.19   0.113    0.911
neuron_reference 4604.72     505.05   9.117 2.89e-09 ***
astro_reference   -21.35     385.48  -0.055    0.956
oligo_reference  -267.37     276.29  -0.968    0.343
micro_reference  -288.05     491.12  -0.587    0.563
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 483.2 on 24 degrees of freedom
Multiple R-squared:  0.818,        Adjusted R-squared:  0.7877
F-statistic: 26.97 on 4 and 24 DF,  p-value: 1.428e-08
```

There is indeed a strong correlation between the expression of Calcineurin A and the neuronal reference signal (neuron_reference), as indicated by the highly significant ($p = 2.89 * 10^{-9}$) coefficient of this reference signal. This reflects the fact that Calcineurin A is expressed in neurons. The coefficient of the neuronal reference signal (4605) represents the normalized neuron-specific expression of this gene. It is the slope of the regression line in the first panel of Figure 1.

Next, we test for a difference in neuron-specific expression in HD versus control samples and model the expression of Calcineurin A as a combination of the neuronal reference signal and the neuron-specific group difference (neuronal interaction regressor)

```
> model2 <- lm(expression["202429_s_at",] ~ neuron_reference + neuron_difference)
```

The fitted model is visualized as follows

```
> crplot(model2, "neuron_reference", g="neuron_difference")
```

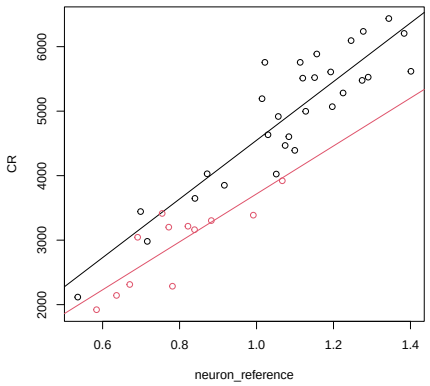

Figure 2: Component-plus-residual plot showing deconvolved neuron-specific expression in controls (black) and HD subjects (red).

It shows that neuronal expression of Calcineurin A is decreased in HD (red) compared to control (black) samples, as indicated by the smaller slope of the regression line for HD samples (Figure 2). The fit summary

```
> summary(model2)

Call:
lm(formula = expression["202429_s_at", ] ~ neuron_reference +
    neuron_difference)

Residuals:
    Min      1Q  Median      3Q     Max
-757.06 -317.30  -49.51  324.19 1109.58

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)         -387.8      379.6  -1.021 0.313490
neuron_reference    4548.5      345.5  13.165 9.82e-16 ***
neuron_difference   -831.5      215.5  -3.859 0.000428 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 435.6 on 38 degrees of freedom
Multiple R-squared:  0.8953,        Adjusted R-squared:  0.8898
F-statistic: 162.5 on 2 and 38 DF,  p-value: < 2.2e-16
```

reveals that the coefficient of the group-specific difference is negative (-831) and highly significant (0.0004). This reflects the fact that Calcineurin A expression is downregulated in neurons of HD patients. Normalized neuron-specific expression in the control group is given by the coefficient of the neuronal reference signal (4548) and normalized neuron-specific expression in HD is given by the sum of both coefficients (4548 - 831 = 3117). These two coefficients are the slopes of the regression lines in Figure 2. The fold change in neuronal expression can thus be easily calculated using the fitted coefficients

```
> foldchange <- (model2$coefficients[2] + model2$coefficients[3]) / model2$coefficients[2]
```

Finally, note that the model fit is excellent (adjusted $R^2 = 0.89$) which means that most of the variations in Calcineurin A expression across samples is explained by the variation in neuronal abundance (as measured by the neuronal reference signal) and the group-specific difference between HD and control samples.

4 Deconvolution of expression profiles

An important aspect of PSEA (and statistical model building in general) is how to choose the parameters to include in the model. Indeed, adding more parameters will always result in a better overall fit (and increase the coefficient of determination R^2) but will not necessarily result in a more informative or predictive expression model. The goal thus is to reach a balance between the number of parameters in the model and how much of the data it can account for.

The stepwise method is a classical approach to model selection. It can be applied to model building for PSEA (as in [2]) and `swlm` provides a simple wrapper function that performs stepwise model selection on every transcript in turn (see `?swlm` for details). However, it might not be computationally efficient when considering a large number of transcripts and might lack flexibility in model specification. Here we will illustrate the "all-subset" approach used in [1] in more details.

We will restrict the statistical models under consideration to those that provide appropriate gene expression models. In the present case, the small number of samples also makes it unlikely to robustly fit highly complex expression models and we might want to exclude models containing several parameters coding for expression changes in different cell populations. The function `lmfitst` efficiently fits a set of models to every transcript in an expression profile and selects the best model for each transcript.

We first need to define a model matrix containing all possible parameters as columns (including an intercept as the first column)

```
> model_matrix <- fmm(cbind(neuron_reference, astro_reference,
+                           oligo_reference, micro_reference), groups)
```

We then specify the subset of models that we want to fit as a list. Each list item represents a model by specifying the included parameters (as their column indices in the model matrix). The function `em_quantvg` enumerates models automatically

```
> model_subset <- em_quantvg(c(2,3,4,5), tnv=4, ng=2)
```

For instance, the 17th model in the list,

```
> model_subset[[17]]

[1] 1 2 6
```

represents an expression model containing an intercept (column 1 in `model_matrix`), the neuronal reference signal (column 2) and the neuron-specific expression change (column 6).

We can then fit each probeset in the expression profile with all models in the subset and for each probeset select the best expression model (using AIC as a criterion)

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

The function `lmfitst` returns two lists. The first contains the identity of the best and next best models for each transcript. The second contains details of the (fitted) best model for each transcript. For PPP3CA, for instance, the selected expression model contains the parameters corresponding to neuronal expression and neuron-specific expression change (as we previously manually worked out)

```
> summary(models[[2]][["202429_s_at"]])

Call:
lm(formula = y[, x] ~ ., data = data.frame(fmdlm[, st[[wcrto1[x]]][-1]]))

Residuals:
    Min      1Q  Median      3Q     Max
-757.06 -317.30  -49.51  324.19 1109.58

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
1   -387.8      379.6  -1.021 0.313490
2   4548.5      345.5  13.165 9.82e-16 ***
6   -831.5      215.5  -3.859 0.000428 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 435.6 on 38 degrees of freedom
Multiple R-squared:  0.8953,        Adjusted R-squared:  0.8898
F-statistic: 162.5 on 2 and 38 DF,  p-value: < 2.2e-16
```

We can then check that the selected models provide appropriate expression models and focus on transcripts with features of interest like e.g. expression in a particular cell population or significant population-specific expression change. To this end, we extract the coefficients, p-values and adjusted R^2 for the selected models using a few ad hoc functions

```
> regressor_names <- as.character(1:9)
> 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')
```

We use specific criteria to filter satisfactory expression models (e.g. sufficient R^2 and small intercept, see [1] for more details)

```
> average_expression <- apply(expression, 1, mean)
> filter <- adjusted_R2 > 0.6 & coefficients[,1] / average_expression < 0.5
```

We can now list transcripts that we would like to focus on (excluding transcripts used to construct reference signals). Here we for instance identify transcripts with significant expression in oligodendrocytes (corresponding to column 4 in `model_matrix`). There is one such transcript in our small example dataset

```
> filter[match(unlist(c(neuron_probesets, astro_probesets, oligo_probesets, micro_probesets)),
+         rownames(expression))] <- FALSE
> select <- which(filter & pvalues[, 4] < 0.05)
> coefficients[select,]
```

```
  coef.1   coef.2   coef.3   coef.4   coef.5   coef.6   coef.7   coef.8
19.38714       NA       NA 69.27515       NA       NA       NA 33.26382
  coef.9
      NA
```

5 Session Information

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

```
R version 4.3.0 RC (2023-04-13 r84266)
Platform: aarch64-apple-darwin20 (64-bit)
Running under: macOS Monterey 12.6.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3

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

time zone: America/New_York
tzcode source: internal

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

other attached packages:
[1] PSEA_1.34.0

loaded via a namespace (and not attached):
 [1] digest_0.6.31       fastmap_1.1.1       xfun_0.38
 [4] knitr_1.42          BiocGenerics_0.46.0 Biobase_2.60.0
 [7] htmltools_0.5.5     rmarkdown_2.21      cli_3.6.1
[10] compiler_4.3.0      tools_4.3.0         evaluate_0.20
[13] yaml_2.3.7          BiocManager_1.30.20 rlang_1.1.0
[16] BiocStyle_2.28.0    MASS_7.3-58.4
```

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.

[2] Alexandre Kuhn, Azad Kumar, Alexandre Beilina, Allissa Dillman, Mark R Cookson, and Andrew B Singleton. Cell population-specific expression analysis of human cerebellum. *BMC Genomics*, 13(610), 2012. URL: http://www.biomedcentral.com/1471-2164/13/610, doi:doi:10.1186/1471-2164-13-610.