PECA: Probe-level Expression Change Averaging

Tomi Suomi

November 4, 2026

tomi.suomi@utu.fi

Contents

1 Introduction **2**

2 Proteomics data **3**

3 Affymetrix microarray data **4**

1 Introduction

PECA determines differential gene/protein expression using directly the probe/peptide-level measurements from Affymetrix gene expression microarrays or proteomic datasets, instead of the common practice of using pre-calculated gene/protein-level values. An expression change between two groups of samples is first calculated for each measured probe/peptide. The gene/protein-level expression changes are then defined as medians over the probe/peptide-level changes. This is illustrated in fig 1. For more details about the probe-level expression change averaging (PECA) procedure, see Elo et al. (2005), Laajala et al. (2009) and Suomi et al. (2015).

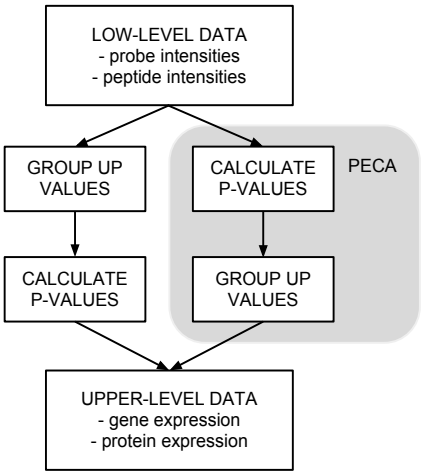


Figure 1: Probe-level expression change averaging.

PECA calculates the probe/peptide-level expression changes using the ordinary or modified t-statistic. The ordinary t-statistic is calculated using the function rowttests in the Bioconductor genefilter package. The modified t-statistic is calculated using the linear modeling approach in the Bioconductor limma package. Both paired and unpaired tests are supported.

The significance of an expression change is determined based on the analytical p-value of the gene-level test statistic. Unadjusted p-values are reported along with the corresponding p-values looked up from beta distribution taking into account the number of probes/peptide per gene/protein. The quality control and filtering of the data (e.g. based on low intensity or probe specificity) is left to the user.

2 Proteomics data

This is an example of tab-separated text file being used as an input. First row of the file contains the column names, which can be later used to select the sample groups for comparison (here from A1 to B3). First column should always contain the id (e.g. UniProt Accession) for which the results are summarized for. Also the column names should not start with a number or contain any special letters.

```
> filePath <- system.file("extdata", "peptides.tsv", package="PECA")
> data <- read.csv(file=filePath, sep="\t")
> head(data)
```

```
  Accession     Sequence           A1          A2         A3         B1
1    P00890 AIGVLPQLIIDR  473957.7045 797849.8423 889740.345 814284.358
2    P00890 AIGVLPQLIIDR    2938.4231   2624.9122   4251.228  10801.785
3    P00890 AIGVLPQLIIDR     523.6715    267.5509   1114.278   5234.959
4    P00890    ALSADLAAR 1088987.8817 752050.9742 619136.168 934525.787
5    P00890  ANQEVLEWLFK  105059.5950  66372.1954 158287.020 207838.534
6    P00890  ANQEVLEWLFK    2106.2001   5952.6551   4077.048  16127.419
          B2         B3
1 898365.633 789901.918
2  18419.565  15483.544
3   4311.213   2235.556
4 750771.408 832069.894
5 151461.468 300008.469
6   9989.237   6234.200
```

PECA can be run by loading the package and setting the sample names of groups to be compared. These should match the column names in the input text file. After this the PECA_tsv function can be called using the input file path and groups as parameters. Note that the sample names provided by user are used to subset the data so that the input file may contain more columns (e.g. samples or other information) than those used for comparison.

```
> library(PECA)
> group1 <- c("A1", "A2", "A3")
> group2 <- c("B1", "B2", "B3")
> results <- PECA_tsv(filePath, group1, group2)
```

Results can then be viewed, stored to disk, or processed further using R.

```
> head(results)
```

```
              slr          t     score  n            p      p.fdr
P00890 -0.4740294 -1.3423222 0.2506136 34 0.0008385463 0.01048183
```

```
P00899 -0.3112597 -0.8144448 0.4610984  6 0.4211490342 0.76510811
P00924  0.1707304  0.5772173 0.5947238  4 0.6579379772 0.96755585
P00925 -0.1036027 -0.5590090 0.6059719 14 0.7929340751 1.00000000
P00927 -0.3864132 -0.6227038 0.5672214  5 0.6245295607 0.96755585
P00931 -0.1835956 -0.8090350 0.4638697  9 0.4117016088 0.76510811
```

3 Affymetrix microarray data

First we load the example library SpikeIn containing the SpikeIn133 dataset that we can use for input. This package needs to be installed separately from Bioconductor.

```
> library(SpikeIn)
> data(SpikeIn133)
```

We subset the original dataset for our purposes, which are two groups with three replicates. First group contains indexes 1, 15, and 29. Second group contains indexes 2, 16, and 30. This subset is then used as input for PECA.

```
> data <- SpikeIn133[,c(1,15,29,2,16,30)]
> results <- PECA_AffyBatch(normalize="true", affy=data)
```

Results can then be viewed, stored to disk, or processed further using R.

```
> head(results)
```

```
                  slr          t     score  n         p p.fdr
1007_s_at -0.062497118 -0.7880544 0.4747409 16 0.4186358     1
1053_at   -0.033076247 -0.2606295 0.8072457 16 0.9974403     1
117_at     0.016630316  0.1743281 0.8700752 16 0.9998545     1
121_at    -0.001890579 -0.0300819 0.9774428 16 1.0000000     1
1255_g_at -0.018923755 -0.1613340 0.8796512 16 0.9999185     1
1294_at   -0.024047196 -0.4637408 0.6669455 16 0.9188036     1
```