

# PLGEM overview

Mattia Pelizzola and Norman Pavelka

October 22, 2008

In this document a brief tutorial to the **Power Law Global Error Model (PLGEM)** analysis method is provided [1]. A **wrapper** function (called *run.plgem*) is provided in the package, which performs all the necessary steps to obtain a list of differentially expressed genes or proteins (DEG), starting from a dataset of class *ExpressionSet*. This input dataset is expected to contain either normalized gene expression values obtained from one-channel microarrays (such as Affymetrix GeneChip [1]), or normalized spectral counts obtained from mass spectrometry-based proteomics methods (such as MudPIT [2]). The wrapper automatically attempts to find the best solution at each step, and requires only modest or no input decisions by the user. For didactic purposes, we will use here a subset of the microarray dataset used in the original publication about **PLGEM**, containing two replicates of LPS-stimulated dendritic cells ('LPS') and four replicates of untreated dendritic cells ('C'):

```
> library(plgem)
> data(LPSeset)
> set.seed(123)
> LPSdegList <- run.plgem(esdata = LPSeset)
```

The above obtained object *LPSdegList* will contain the list of genes or proteins, selected as significantly changing between experimental condition 'LPS' and baseline condition 'C', at the default significance level 0.001.

To provide advanced users with a higher control on the inner workings of the **PLGEM** pipeline, the individual functions called by *run.plgem* are also described in this tutorial. We will use them now, step by step.

The first step is to **fit the model** to the microarray or proteomics dataset. By fitting the model we will obtain a mathematical relationship that will allow us to determine the expected standard deviation associated to a given average gene expression value or average protein abundance level.

**PLGEM** can only be fitted on a set of replicates of a same experimental condition, therefore we first need to choose which condition to use for the fitting step. In our dataset two conditions are provided: C and LPS. Usually the most replicated one is chosen, therefore we choose the first condition C (i.e. *fit.condition* 1), because it contains 4 replicates.

Moreover, we can change the default values of parameters  $p$  and  $q$ . Briefly,  $p$  represents the number of intervals (or bins) used to partition the expression value range of the dataset. In the original publication we observed that  $p$  can be modified over a wide range of values, without any major effects on the final results, except when it was chosen to close to the total number of

genes or proteins in the dataset [1]. As a rule of thumb,  $p$  should be no more than one tenth of the number of genes or proteins. The default of 10 should therefore be OK for most microarray experiments, but could be set lower for proteomics data where less than 100 proteins were identified.

$q$  is the quantile of the location-dependent spread used to fit the model. The default of  $q$  is set to 0.5, because this represents the median value, which is what you are looking for when modeling the variability. We recommend to only modify this parameter for very special purposes, e.g. for the determination of empirical confidence intervals of standard deviation.

Finally, setting *plot.file*=TRUE saves a png image to a file instead of plotting it, therefore we won't change its default.

Moreover it is possible to evaluate the fitting of the model setting the option *fittingEval*. If this argument is set to TRUE, a multi-panel plot is produced where the residuals of the model are evaluated. A good fit is characterized by a near-normal distribution and an horizontal symmetric ranking-plot of the residuals.

```
> LPSfit <- plgem.fit(data = LPSeset, covariateNumb = 1, fit.condition = 1,  
+   p = 10, q = 0.5, plot.file = FALSE, fittingEval = TRUE, verbose = TRUE)
```

```
fitting PLGEM..
```

```
samples extracted for fitting:
```

```
  conditionName
```

```
C1             C
```

```
C2             C
```

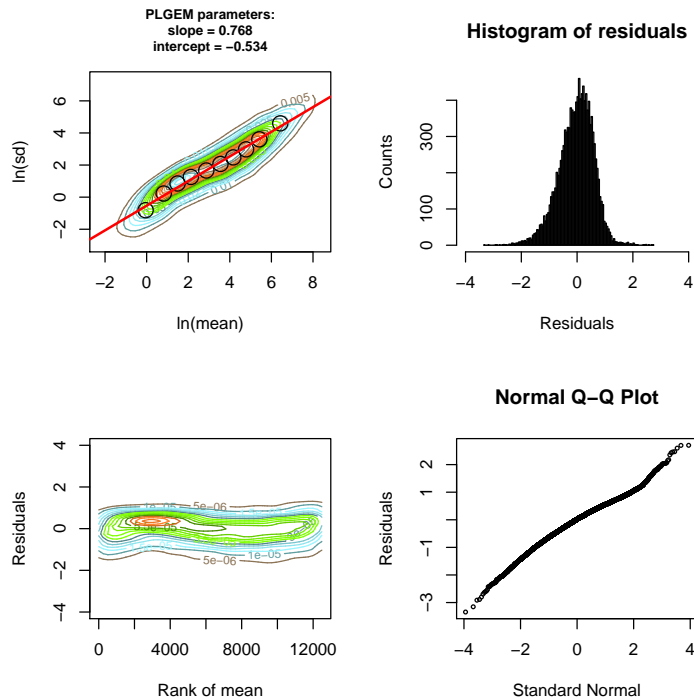
```
C3             C
```

```
C4             C
```

```
determining modelling points...
```

```
fitting data and modelling points...
```

```
done with fitting PLGEM.
```



The next step is the computation of the signal-to-noise ratio (STN) statistic for the detection of differential expression. The STN is determined using the model-derived spread estimates instead of the data-derived ones. Therefore it is necessary to give to the `plgem.obsStn` function the model parameters `slope` and `intercept` determined during the model fitting that are contained in the value returned by `plgem.fit` function. By default, all experimental conditions (according to the values of the `covariateNumb` covariate defined in the `phenoData` slot of the `ExpressionSet`) are compared to the first condition (baseline). If the condition to be treated as the baseline is not the first one, you can change this by modifying the argument `baseline.condition`. A matrix of observed STN is determined, where the number of rows are the number of genes or proteins in the dataset and the number of columns are the number of comparisons that can be performed in the dataset (i.e. the total number of experimental conditions minus one). In this case, the dataset contains only one condition to be compared to the baseline, therefore the matrix will be a one-dimensional array of observed PLGEM-STN values.

```
> LPSobsStn <- plgem.obsStn(data = LPSeset, covariateNumb = 1,
+   baseline.condition = 1, plgemFit = LPSfit, verbose = TRUE)
```

```
calculating observed PLGEM-STN statistics:found 1 condition(s) to compare to the baseline.
working on baseline C ...
C1 C2 C3 C4
working on condition LPS ...
LPS1 LPS2
done with calculating PLGEM-STN statistics.
```

In order to get an estimate of the distribution of the test statistic under the null-hypothesis of not differential expression, a `resampled statistic` is determined using the method described in the paper [1]. The number of iterations of the resampling step should be correlated with the total number of replicates that are present in the data set. If this argument is set to `automatic`, the number of iterations is automatically determined based on the total number of possible combinations. In this case, an upper threshold of 500 iterations is set to avoid excessive computation time. This should be fine for most purposes.

```
> LPSresampledStn <- plgem.resampledStn(data = LPSeset, plgemFit = LPSfit,
+   iterations = "automatic", verbose = TRUE)
```

```
calculating resampled PLGEM-STN statistics:found 1 condition(s) to compare to the baseline
baseline samples:
C1 C2 C3 C4
resampling on samples:
C1 C2 C3 C4
Using 500 iterations...
working on cases with 2 replicates...
Iterations: 20 40 60 80 100 120 140 160 180 200 220 240 260 280 300 320
done with calculating resampled PLGEM-STN statistics.
```

Next, p-values are calculated for each observed STN value via a call to function `plgem.pValue`. Resampled STN values are also required by this function, because they will be used to build empirical cumulative distribution functions of the STN values that can be observed under the null hypothesis:

```
> LPSpValues <- plgem.pValue(observedStn = LPSobsStn, plgemResampledStn = LPSresampledStn,
+   verbose = TRUE)
```

```
calculating PLGEM p-values... done.
```

```
> head(LPSpValues)
```

```

                LPS
100001_at 0.163648302
100002_at 0.154047406
100003_at 0.001344971
100004_at 0.222065343
100005_at 0.352898783
100006_at 0.509934017
```

Finally, DEG are selected at the given significance level `delta` via a call to function `plgem.deg`. The chosen value of `delta` can be seen as an estimate of the False Positive Rate (FPR). Therefore, in case of a microarray dataset with 10000 genes of which not a single one is truly differentially expressed, choosing a `delta` of 0.001 will select roughly 10 genes by chance:

```
> LPSdegList <- plgem.deg(observedStn = LPSobsStn, plgemPval = LPSpValues,
+   delta = 0.001, verbose = TRUE)
```

```

selecting significant DEG:found 1 condition(s) compared to the baseline.
Delta = 0.001
      Condition = LPS
delta: 0.001 condition: LPS found 340 DEG
done with selecting significant DEG.

> head(LPSdegList[["0.001"]][["LPS"]])

100012_at 100213_f_at 100277_at 100278_at 100342_i_at 100379_f_at
-4.782786 3.729672 6.607890 3.643951 3.784421 7.204393

```

Above function returns a list with a number of items that is equal to the number of different significance levels *delta* used as input. In this case the default single value of 0.001 was used, so the list will contain only one item at this level. This item is again a list, whose number of items correspond to the number of performed comparisons, i.e. the number of conditions in the starting ExpressionSet minus the baseline, in this case again only one. In each list-item the values are the observed STN and the names are the IDs of the significantly changing genes or proteins, as defined in the ExpressionSet.

Finally, the obtained list of DEG can be written to the current working directory using the *plgem.write.summary* function.

## References

- [1] Pavelka, N., Pelizzola, M., Vizzardelli, C., Capozzoli, M., Splendiani, A., Granucci, F. and P. Ricciardi-Castagnoli (2004). A power law global error model for the identification of differentially expressed genes in microarray data. *BMC Bioinformatics*, 5, 203.
- [2] Pavelka, N., Fournier, M., L., Swanson, S., K., Pelizzola, M., Ricciardi-Castagnoli, P., Florens, L. and M. P. Washburn (2007). Statistical similarities between transcriptomics and quantitative shotgun proteomics data. *Molecular and Cellular Proteomics*, 19, in press.