RPA: analysis of probe reliability and gene expression on short oligonucleotide arrays

Leo Lahti*
leo.lahti@iki.fi

June 27, 2011

1 Introduction

RPA (Robust Probabilistic Averaging)[1] provides tools for probe reliability analysis and gene expression preprocessing for (Affymetrix) short oligonucleotide arrays. It can also be used more generally to summarize multivariate observations that target the same objects with varying degree of reliability.

RPA can be used for standard preprocessing tasks in gene expression studies; it has been shown to outperform other popular preprocessing methods in differential gene expression analysis. In addition, the method provides explicit, data-driven estimates of probe reliability; poorly performing probes are downweighted in the model, which yields more accurate estimates of gene expression and can reveal noisy probes independently of the error source. The noise estimates have been validated by comparisons to known probe-level error sources. The probabilistic formulation allows also incorporation of prior information concerning probe reliability into gene expression analysis [7].

2 Preprocessing gene expression data with RPA

RPA provides a wrapper ('rpa') for convenient preprocessing of Affymetrix arrays. Alternative CDF environments are also supported (see help(rpa) for details). Here is a preprocessing example with an example data set:

```
> library(affydata)
> data(Dilution)
> eset <- rpa(Dilution)
```

*http://www.iki.fi/Leo.Lahti

[1] http://bioconductor.org/packages/release/bioc/html/RPA.html

The input is an AffyBatch object. CEL files can be read in as affybatch with the ReadAffy function of the affy package. The output is an ExpressionSet object, which allows downstream analysis of the results using standard R/BioC tools for gene expression data.

3 Probe reliability analysis

RPA operates on affybatch objects [3]. An affybatch contains the probe-level data of Affymetrix arrays. Our toy examples use the `Dilution` dataset provided by *affydata* package. Load example data (the 'Dilution' affybatch):

```
> require(affy)
> require(affydata)
> data(Dilution)
```

RPA.pointestimate is the main function. Let us perform the analysis for particular probesets in the Dilution data (the whole data set will be analyzed by default if 'sets' is not given).

```
> require(RPA)
> sets <- geneNames(Dilution)[1:2]
> rpa.results <- RPA.pointestimate(Dilution, sets)
```

The 'rpa2eset' function can be used to coerce the probeset-level expression estimates into an ExpressionSet object.

The results for a particular probeset are visualized with

```
> plot(rpa.results, set = "1000_at")
```

The output is shown in Figure 1. See help('rpa.plot') for details.

3.1 Estimating probe-specific noise and probe reliability

RPA estimates the noise level of each individual probe through the probe-specific variance parameter (τ_j^2). These can be obtained with

```
> noise <- get.probe.noise.estimates(rpa.results)
```

The higher the variance, the more noisy the probe. Inverse of the variance, $\frac{1}{\tau_j^2}$, can be used to quantitate probe reliability. Note that the relative weight of a probe within probeset is determined by the relative noise of the probe with respect to the other probes in the same probeset. Comparison of probe-specific variances across probesets may benefit from normalization of this effect. The get.probe.noise.estimates function can optionally provide normalized versions of the noise estimates.

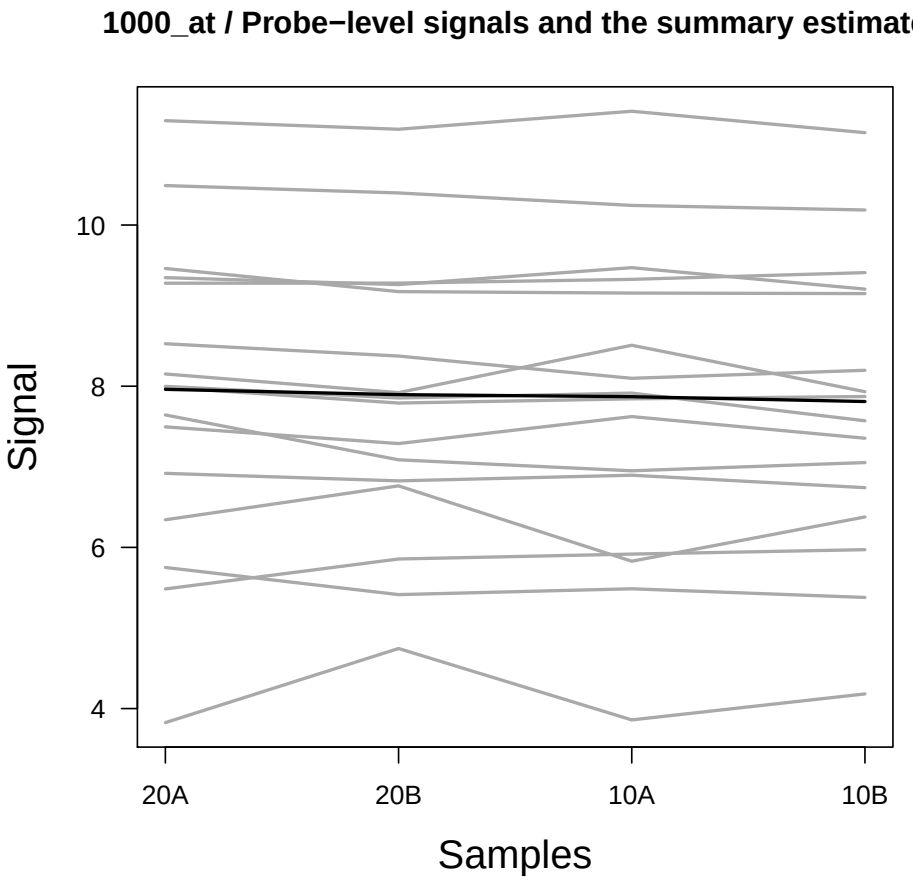

Figure 1: Estimated probe-specific variances and gene expression signal for an example probe set.

3.2 Setting probe-specific priors

Prior information of probe reliability can be set by tuning the shape (α) and scale (β) parameters of the inverse Gamma distribution, which is the conjugate prior for the variances. Set priors for a particular probeset. If the 'priors' parameter is not given, non-informative priors will be given for the other probesets:

```
> alpha <- beta <- rep(1, 16)
> probe.index <- 5
> alpha[[probe.index]] <- 3
> beta[[probe.index]] <- 1
> priors <- set.priors(Dilution, set = "1000_at", alpha, beta)
```

Run RPA with priors:

```
> rpa.results <- RPA.pointestimate(Dilution, sets, priors = priors)
```

3.3 General usage

RPA can be used more generally to summarize multivariate observations of the same object with varying noise levels, see function rpa.fit:

```
> res <- rpa.fit(S)
```

4 The probabilistic model

4.1 Relation to other probe-level models

RPA differs from other popular preprocessing algorithms in two key respects. First, it utilizes probe-level estimates of differential expression; these are calculated *before* probeset-level summarization, which avoids certain probe-level effects that obscure the results in other preprocessing methods where probes with various affinities and contamination levels are combined into a probeset-level summary prior to differential expression analyses. In particular, our procedure avoids the modeling of unidentifiable probe affinities, which is the key probe-specific parameter in many preprocessing methods. Second, RPA provides tools for investigating the reliability of individual probes in terms of a probe-specific variance. This can be used in microarray design and in confirming the end results of a microarray study. These properties distinguish RPA from other probe-level preprocessing methods such as dChip's MBEI [8], RMA [5], or FARMS [4].

4.2 Summary of RPA model

4.2.1 Background correction and normalization

The probe-level data is background corrected, normalized, and log2-transformed before the analysis. By default, RPA uses the background correction model of RMA [6] and quantile normalization [2]. Our implementation utilizes the *affy* package [3] to handle probe-level data. For details about short oligonucleotide arrays and the design of the Affymetrix GeneChip arrays, see the Affymetrix MAS manual [1].

4.2.2 Probe reliability estimation and summarization

The RPA algorithm is used to obtain probeset-level summaries for gene expression and to estimate probe-specific noise. RPA assumes a Gaussian model for probe effects. Let us consider a probe set targeted at measuring the expression level of target transcript g. Probe-level observation s_{ij} of probe j on array i is modeled as a sum of the true expression signal (common for all probes in the probeset), and probe-specific Gaussian noise: $s_{ij} = g_i + \mu_j + \varepsilon_{ij}$. The stochastic noise component is probe-specific, distributed as $\varepsilon_{ij} \sim N(0, \tau_j^2)$. The variance parameters $\{\tau_j^2\}$ are of interest in probe reliability analysis; the inverse variance $1/\tau_j^2$ can be used to measure of probe reliability (see get.probe.noise.estimates function).

The mean parameter μ_j of the noise model describes systematic probe affinity effect, which is unidentifiable. These parameters cancel out in RPA when the signal log-ratio between a user-specified 'reference' array and the remaining arrays is calculated at probe level: the differential expression signal between arrays $t = \{1, \ldots, T\}$ and the reference array c for probe j is given by $m_{tj} = s_{tj} - s_{cj} = g_t - g_c + \varepsilon_{tj} - \varepsilon_{cj} = d_t + \varepsilon_{tj} - \varepsilon_{cj}$. In vector notation the differential expression profile of probe j across the arrays can be written as $\mathbf{m}_j = \mathbf{d} + \boldsymbol{\varepsilon}_j$. In practice, $\mathbf{d}$ and the probe-specific variances $\{\tau_j\}_{j=1}^P$ for the P probes within the probeset are estimated simultaneously based on the probabilistic model. With large sample sizes the solution will converge to estimating the mean of the probe-level observations weighted by probe reliability. Note that the algorithm is robust to choice of the reference array since the reference effect is marginalized out in the probabilistic treatment; our experiments confirm that the probe-level noise estimates are not affected by the choice of the reference array.

4.2.3 Estimation of probe affinity terms

Probe affinity terms and the original signal level are estimated after summarizing the probe-level differential gene expression estimates. First an estimate of the absolute signal level is calculated based on particular modeling assumptions. Then probe-specific affinities are calculated by comparing each probe to the probeset-level signal estimate.

Let us write the probe-level observation in terms of differential expression signal, absolute signal level, and stochastic noise as $\mathbf{s}_j = \mathbf{d} + \mu + \varepsilon$, where μ is a scalar (vector with identical elements). This will indicate how much probe-level observation deviates

from the estimated signal shape $\mathbf{d}$. This can be decomposed as $\mu = \mu_{real} + \mu_{probe}$, where μ_{real} describes the 'real' signal level, common for all probes and μ_{probe} describes probe affinity effect. Let us assume that μ_{probe} $N(0, \sigma^2_{probe})$. This encodes the assumption that in general the affinity effect of each probe tends to be close to zero. Then ML estimates of μ_{real} and μ_{probe} are calculated based on these particular assumptions. This part of the algorithm has not been defined in full probabilistic terms, we are only providing the point estimates.

If an identical prior is used for all probes in affinity estimation then μ_{real} is estimated as the average of the probe effects μ and the probe-specific affinities μ_{probe} will sum to exactly zero ('zeromean' option). This is analogous to the model used in RMA, which uses medianpolish algorithm to fit this assumption. In contrast to our model the stochastic probe effects are not probe-specific in RMA. We suggest an alternative approach where probes are weighted during affinity estimation ('rpa' option). While σ^2 estimated by RPA measures stochastic noise, not the affinity effect, we utilize them to give a heuristic weigh for the probes in affinity estimation according to how much they contribute to the overall signal shape. Intuitively, probes that have little effect on the signal shape (i.e. are very noisy and likely to be contaminated by many unrelated signals) should also contribute less to the absolute signal estimate. The probe affinities are expected to sum to zero but the model allows some flexibility.

5 Citing RPA

Please cite [7] when using the package.

6 Details

This document was written using:

```
> sessionInfo()

R version 2.13.0 (2011-04-13)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=C              LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
```

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

other attached packages:
[1] hgu95av2cdf_2.8.0 RPA_1.8.01        affydata_1.11.11  affy_1.30.0
[5] Biobase_2.12.1

loaded via a namespace (and not attached):
[1] affyio_1.20.0         preprocessCore_1.14.0 tools_2.13.0
```

References

[1] Affymetrix. *Affymetrix Microarray Suite User Guide*. Affymetrix, Santa Clara, CA, version 5 edition, 2001.

[2] B. M. Bolstad, R. A. Irizarry, M. Astrand, and T. P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.

[3] L. Gautier, L. Cope, B. M. Bolstad, and R. A. Irizarry. affy–analysis of Affymetrix GeneChip data at the probe level. *Bioinformatics*, 20(3):307–315, 2004.

[4] S. Hochreiter, D.-A. Clevert, and K. Obermayer. A new summarization method for affymetrix probe level data. *Bioinformatics*, 22(8):943–949, 2006.

[5] R. A. Irizarry, B. M. Bolstad, F. Collin, L. M. Cope, B. Hobbs, and T. P. Speed. Summaries of Affymetrix GeneChip probe level data. *Nucl. Acids Res.*, 31(4):e15, 2003.

[6] R. A. Irizarry, B. Hobbs, F. Collin, Y. D. Beazer-Barclay, K. J. Antonellis, U. Scherf, and T. P. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 2003.

[7] L. Lahti, L. L. Elo, T. Aittokallio, and S. Kaski. Probabilistic analysis of probe reliability in differential gene expression studies with short oligonucleotide arrays. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(1):217–225, 2011.

[8] C. Li and W. H. Wong. Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. *Proc. Natl. Acad. Sci.*, 98:31–36, 2001.