Robust Probabilistic Averaging for probe performance analysis and preprocessing on short oligonucleotide arrays

Leo Lahti*
University of Helsinki
leo.lahti@iki.fi

November 12, 2011

1 Introduction

RPA (Robust Probabilistic Averaging)[1] provides tools for probe reliability analysis and gene expression preprocessing for (Affymetrix) short oligonucleotide arrays, and more generally to summarize normally distributed multivariate observations that target the same object with varying degrees of reliability. RPA provides explicit data-driven estimates of probe performance, i.e. affinity and probe-specific noise level. Affinities are taken into account in summarizing the probes and noisy probes are downweighted, which yields more accurate estimates of gene expression [9]. The probabilistic formulation allows also incorporation of prior information concerning probe reliability into analysis.

2 RPA preprocessing

RPA provides a wrapper ('rpa') for convenient preprocessing of Affymetrix arrays and support for alternative CDF environments. RPA operates on affybatch objects [5]. Load the example data:

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

*http://www.iki.fi/Leo.Lahti
[1] http://bioconductor.org/packages/release/bioc/html/RPA.html

The toy example uses the `Dilution` dataset provided by *affydata* package. To preprocess the affybatch, use:

```
> eset <- rpa(Dilution)
```

Input is an AffyBatch object, obtained from CEL files 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 performance analysis

Use *RPA.pointestimate* to investigate particular probesets:

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

The rpa.results object contains probe-specific affinity and variance estimates (affinity, sigma2), the probeset-level summary estimate (d) and other information. The results can be visualized with

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

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

3.1 Estimating probe reliability

The noise level of individual probes can be quantified based on the probe-specific variance parameter (τ_j^2):

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

The precision $\frac{1}{\tau_j^2}$, can be used to quantitate probe reliability. Comparison of probe-specific variances across probesets may benefit from probeset-specific normalization; see help(get.probe.noise.estimates) for details.

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 conjugate prior for the variances. If the 'priors' parameter is not given, non-informative priors will be given for the other probesets:

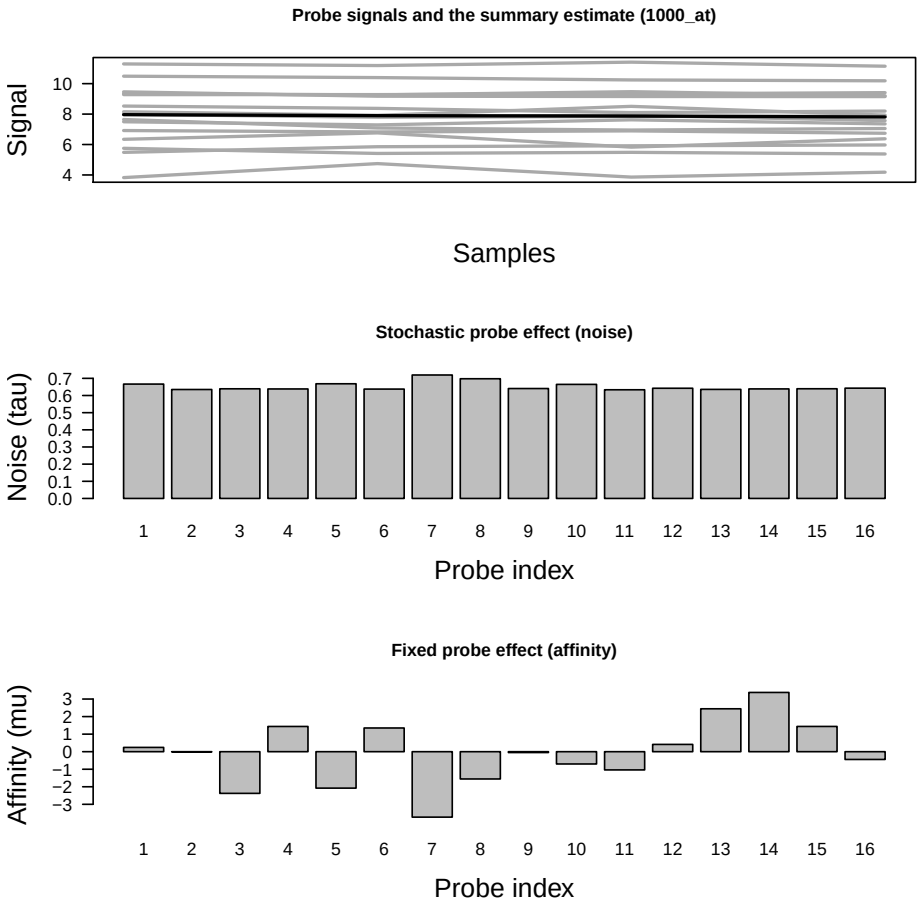

Figure 1: Estimated probe affinities, probe-specific noise level (standard deviation) and probeset-level summary estimate and probe-level signals for an example probe set.

```
> priors <- list(alpha = NULL, beta = NULL, d = NULL)
> set <- "1000_at"
> priors$alpha <- 2
> priors$beta[[set]] <- rep(1, 16)
> probe.index <- 5
> priors$beta[[set]][[probe.index]] <- 3
> rpa.results <- RPA.pointestimate(Dilution, sets, priors = priors)
```

3.3 General usage

Robust Probabilistic Averaging can be used to summarize any multivariate observations concerning the same object and having varying (Gaussian) affinities and noise levels:

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

4 The probabilistic model

4.1 Relation to other probe-level models

RPA differs from other popular preprocessing algorithms such as dChip's MBEI [10], RMA [7], or FARMS [6] in two key respects. It calculates probe-level estimates of differential expression; *before* probeset-level summarization, which will avoid the modeling of unidentifiable probe affinities in determining the signal shape. Second, RPA provides tools for investigating the performance of individual probes. This can be used in microarray design and to confirm the end results of a microarray study.

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 [8] and quantile normalization [2]. Our implementation utilizes the *affy* package [5] 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. 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 Probe affinity estimation

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} $N(0, \sigma_{probe}^2)$. 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} would sum to exactly zero, analogous to RMA. We suggest an alternative approach where probes are weighted during affinity estimation according to their noisiness σ^2. Then noisy probes that have little effect on the signal shape will contribute less to the absolute signal estimate while the expected sum of probe affinities remains at zero.

5 Validation

The model can reveal noisy probes independently of the error source noise; noise estimates have been validated by comparisons to known probe-level error sources [9]. RPA has been shown to outperform other popular preprocessing methods in cross-platform studies [4, 9] and spike-in data sets. RPA has also been compared to other preprocessing methods through the AffyCompII benchmarking website[2] [3]. RPA outperformed some widely-used preprocessing algorithms such as RMA, measured in terms of average rank over the different test statistics. Overall the results indicate that RPA has a comparable preprocessing performance with standard preprocessing algorithms, which supports the validity of the probe-level affinity and noise estimates in RPA. Note that while RPA can be used for preprocessing, its primary focus is on probe performance analysis.

6 Citing RPA

Please cite [9].

7 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=en_US.UTF-8
 [5] LC_MONETARY=C              LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=fi_FI.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_1.11.06       multicore_0.1-5   affydata_1.11.11
[5] affy_1.30.0       Biobase_2.12.1
```

[2] http://affycomp.jhsph.edu/AFFY2/TABLES.hgu/0.html as of March 13, 2011

```
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. M. Cope, R. A. Irizarry, H. A. Jaffee, Z. Wu, and T. P. Speed. A benchmark for Affymetrix GeneChip expression measures. *Bioinformatics*, 20:323–331, 2004.

[4] L. L. Elo, L. Lahti, H. Skottman, M. Kyläniemi, R. Lahesmaa, and T. Aittokallio. Integrating probe-level expression changes across generations of Affymetrix arrays. *Nucleic Acids Research*, 33(22):e193, 2005.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.