# The AffyRNADegradation Package

Mario Fasold

October 30, 2024

Affymetrix 3' expression arrays employ a specific experimental protocol and a specific probe design that allows assessment of RNA integrity based on probe signal data. Problems of RNA integrity are primarily governed to the degradation of the target transcripts. We have shown in Fasold and Binder (2012) that

(i) degradation leads to a probe positional bias that needs to be corrected in order to compare expression of samples with varying degree of degradation, and

(ii) it is possible to estimate a robust and accurate measure of RNA integrity from the probe signals that, for example, can be used to study degradation within the large number of available microarray data.

The rationale and further analysis are described in the accompanying publication by Fasold and Binder. We here show how to utilize this package for both problems.

## 1 Basic RNA Degradation Analysis

We here show how to use the package for the analysis of RNA degradation. Let us first load exemplar data provided by the *AmpAffyExample* package into the environment.

```
> library(AffyRNADegradation)
> library(AmpAffyExample)
> data(AmpData)
> AmpData

AffyBatch object
size of arrays=712x712 features (22 kb)
cdf=HG-U133A (22283 affyids)
number of samples=6
number of genes=22283
annotation=hgu133a
notes=
```

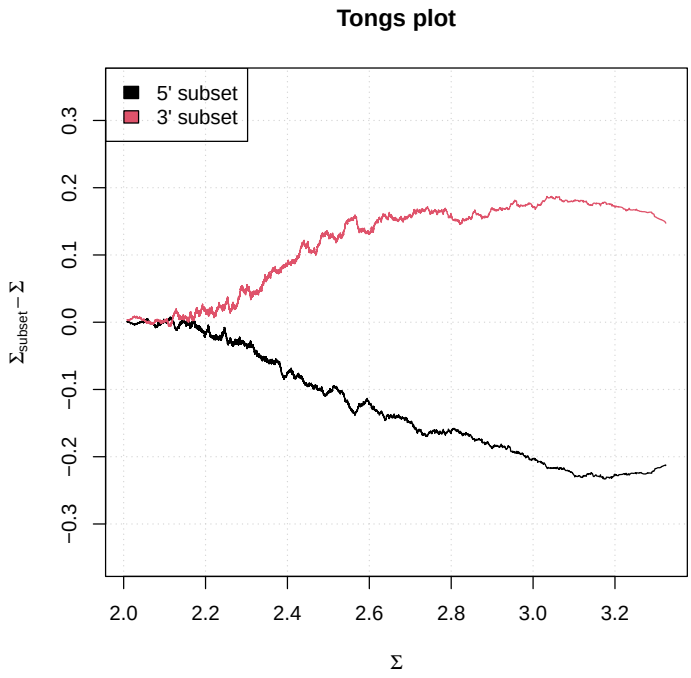


Figure 1: The tongs plot shows that the intensity difference between 3' and 5' probes increases with $\Sigma = \langle \log I \rangle$. $\langle\rangle$ here denotes either averaging over all probes within the probeset, or averaging over the 3' or 5' subset of probes in $\Sigma_{subset}$.

Every transcript is measured by a set of 11-16 probes. The log-average intensity difference between probes located closer to the 3' end of the target transcripts and those located further away constitutes the probe positional bias. It can be visualized using the *tongs plot*.

```
> tongs <- GetTongs(AmpData, chip.idx = 4)
> PlotTongs(tongs)
```

Figure 1 shows that the bias relates to the expression level of the transcripts. As this can vary from sample to sample, it must be considered in estimating of RNA degradation.

The function `RNADegradation` performs the basal analysis of RNA degradation based on raw probe intensities stored in an AffyBatch object. The result is an `AffyDegradationBatch` object that contains the corrected probe intensities as well as several statistical parameters.

```
> rna.deg <- RNADegradation(AmpData, location.type = "index")
```

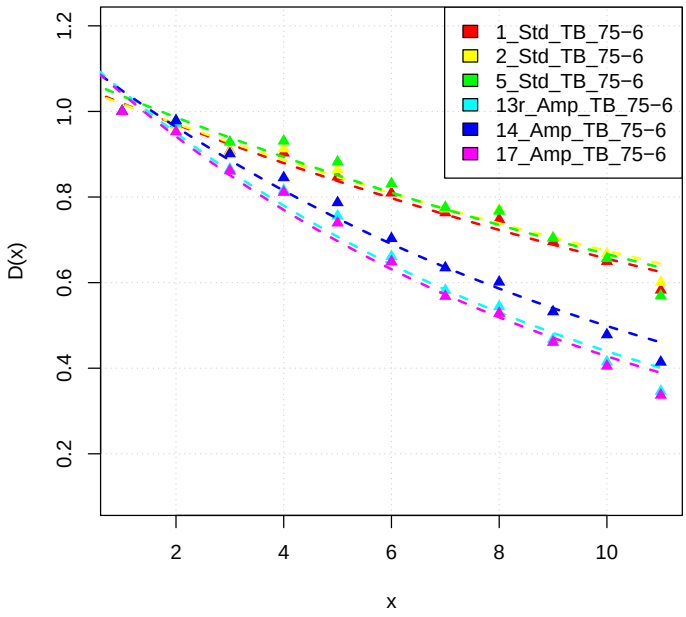


Figure 2: Probe degradation plot. The points show the average probe intensity of expressed genes for each index $x = 1, ..11$ relative to the average intensity at position $x = 1$. The lines are a fitted decay function.

We can visualize the probe positional bias using the `PlotDx` function.

```
> plotDx(rna.deg)
```

Figure 2 shows the results. Different degradation between different samples are observed.

To access the parameter $d$, which provides a robust, sample-wise measure for the degree of RNA degradation, one can use the function

```
> d(rna.deg)
```

```
  1_Std_TB_75-6   2_Std_TB_75-6   5_Std_TB_75-6 13r_Amp_TB_75-6  14_Amp_TB_75-6
      0.6161481       0.6339041       0.6129885       0.3804619       0.4462438
17_Amp_TB_75-6
      0.3710109
```

## 2 Using Absolute Probe Locations

Instead of using the probe index within the probeset as argument of the degradation degree, one can use the actual probe locations within the transcript. We have pre-computed the distance of each probe to the 3' end of its target transcript for all Affymetrix 3' expression arrays. These probe location files are available under the URL `http://www.izbi.uni-leipzig.de/downloads_links/programs/rna_integrity.php`.

In order to perform the analysis and correction using absolute probe locations, one must first download the probe location file for the used chip type. You can then start the analysis using `RNADegradation`, as above, but selecting `absolute` as `location.type`. The parameter `location.file.dir` must specify the download directory of the probe location file.

## 3 Creating Custom Probe Location Files

It is possible to use custom probe locations, for example if one wishes to analyze custom built microarrays or if one relies on alternative probe annotations. For this, one has to create a probe location file similar to the pre-built ones used in the previous section.

Here is how to generate such a file. First, create a data frame with the name `probeDists` containing the five columns `Probe.Set.Name`, `Probe.X`, `Probe.Y`, `Probe.Distance` and `Target.Length`. `Probe.Set.Name` is of class character and contains the Affymetrix probe set id. The remaining variables are of class integer.

`Probe.X` and `Probe.Y` denote the coordinates of the probe on the microarray. These are important because the coordinates are used to map the probes to the AffyBatch object using the `xy2indices` function from the *affy* package. This implies that the ordering of the table rows can be of any kind. It is however important that this information can be mapped to every probe pair in the AffyBatch intensity array (as for example shown in the `affy::pm` function).

`Probe.Distance` contains the probe location: the number of nucleotides counted between the designated 3'-end of the transcript and the first (i.e. nearest) base of the 25meric probe sequence. The last column `Target.Length` contains the length of the target in base bairs - it is not used in this package and can be set to any value. The following table shows an example of the `probeDists` data frame:

| | Probe.Set.Name | Probe.X | Probe.Y | Probe.Distance | Target.Length |
|---|---|---|---|---|---|
| 1 | 1007_s_at | 467 | 181 | 608 | 3938 |
| 2 | 1007_s_at | 531 | 299 | 495 | 3938 |
| 3 | 1007_s_at | 86 | 557 | 426 | 3938 |
| ... | ... | ... | ... | ... | .. |

This table is then stored in an R binary object file. The filename must be set to the chip type identifier as given by the `affy::cdfName` function with the file ending `.Rd`:

```
> filename = paste(cdfName(AmpData), ".Rd", sep="")
> save(probeDists, file = filename)
```

To use the custom probe locations, start the analysis using `RNADegradation`, as above with `location.type=absolute` and `location.file.dir` set to the directory containing the custom probe location file.

# 4 Correction of the Bias and Integration into the Microarray Calibration Pipeline

The correction of the probe positional bias is performed within the `AffyRNADegradation` function. The result is a new `AffyBatch` object with corrected probe level intensities. It can be accessed using the `afbatch` function

```
> afbatch(rna.deg)
```

It is possible to replace the original raw data with this data corrected for probe positional bias, before performing further microarray normalization and summarization (e.g. using RMA).

Alternatively, the correction can be performed after probe-level normalization. The following example shows how to first apply the VSN normalization method, then correct for probe positional bias to finally get summarized expression measures

```
> library(vsn)
> affydata.vsn <- do.call(affy:::normalize, c(alist(AmpData, "vsn"), NULL))
> affydata.vsn <- afbatch(RNADegradation(affydata.vsn))
> expr <- computeExprSet(affydata.vsn, summary.method="medianpolish", pmcorrect.method="pmo
```

# 5 Citing AffyRNADegradation

Please cite (Fasold and Binder, 2013) when using the package.

# 6 Details

This document was written using:

```
> sessionInfo()
```

```
R version 4.4.1 (2024-06-14)
Platform: x86_64-pc-linux-gnu
```

```
Running under: Ubuntu 24.04.1 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.26.so;  LAPACK version

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    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

time zone: Etc/UTC
tzcode source: system (glibc)

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

other attached packages:
[1] hgu133acdf_2.18.0         AmpAffyExample_1.45.0
[3] AffyRNADegradation_1.53.0 affy_1.83.1
[5] Biobase_2.65.1            BiocGenerics_0.51.3

loaded via a namespace (and not attached):
 [1] bit_4.5.0               preprocessCore_1.67.1   jsonlite_1.8.9
 [4] compiler_4.4.1          BiocManager_1.30.25     crayon_1.5.3
 [7] blob_1.2.4              Biostrings_2.73.2       IRanges_2.39.2
[10] png_0.1-8               fastmap_1.2.0           R6_2.5.1
[13] XVector_0.45.0          GenomeInfoDb_1.41.2     knitr_1.48
[16] maketools_1.3.1         GenomeInfoDbData_1.2.13 AnnotationDbi_1.67.0
[19] DBI_1.2.3               affyio_1.75.1           rlang_1.1.4
[22] KEGGREST_1.45.1         cachem_1.1.0            xfun_0.48
[25] sys_3.4.3               bit64_4.5.2             RSQLite_2.3.7
[28] memoise_2.0.1           cli_3.6.3               zlibbioc_1.51.2
[31] S4Vectors_0.43.2        vctrs_0.6.5             buildtools_1.0.0
[34] stats4_4.4.1            httr_1.4.7              tools_4.4.1
[37] UCSC.utils_1.1.0
```

# References


Mario Fasold and Hans Binder. Estimating RNA-quality using GeneChip microarrays. *BMC Genomics*, 13:186, 2012.

Mario Fasold and Hans Binder. AffyRNADegradation: Control and correction of RNA quality effects in GeneChip expression data. *Bioinformatics*, 29(1):129–131, 2013. doi: 10.1093/bioinformatics/bts629. URL `http://bioinformatics.oxfordjournals.org/content/29/1/129.abstract`.