

# affyPLM: the "threestep" function

Ben Bolstad

`bolstad@stat.berkeley.edu`

November 25, 2003

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Using the threestep function</b>	<b>1</b>
<b>3</b>	<b>Methods available in <i>threestep</i></b>	<b>2</b>

## 1 Introduction

This document describes the threestep function which is part of the affyPLM package. The threestep function is an alternative method of computing expression measures (which can also be generated using *expresso*, *rma()* and other functions from the *affy* package). In the threestep framework an expression measure consists of threesteps: background/signal adjustment, normalization and then summarization. One important thing to notice is that all expression measures returned by *threestep* are all in the  $\log_2$  scale.

After starting R, the package should be loaded using this will load affyPLM as well as the affy package and its dependencies.

## 2 Using the threestep function

The simplest method of calling the *threestep* function is to call it with no arguments beyond an AffyBatch.

```
> data(Dilution)
> eset <- threestep(Dilution)
```

this will compute the conventional RMA expression measure.

But we can go beyond this by altering any of the threesteps. This can be done using the three parameters *background.method*, *normalize.method*, *summary.method*. For example

```
> eset <- threestep(Dilution, background.method = "MASIM", normalize.method = "quantile",
+   summary.method = "tukey.biweight")
```

computes an expression measure where we background correct/signal adjust by breaking the array into grids doing a location dependent adjustment, then subtracting the "ideal mismatch" from the perfect match. Normalization is done using the quantile normalization method and then summarization using a 1-step Tukey Biweight.

An even more unconventional expression measure is to background correct/signal adjust by subtracting the ideal mismatch, carry out no normalization, then summarize by taking the 2nd largest PM (which in this case will really be the largest PM after subtracting the ideal mismatch).

```
> eset <- threestep(Dilution, background.method = "IdealMM", normalize = "quantile",
+   summary.method = "log.2nd.largest")
```

### 3 Methods available in *threestep*

The following tables outline the methods that are available for each of the threesteps.

Background methods		
Option	Name	Description
RMA.2 (default)	RMA convolution model background	
RMA.1	RMA convolution model background	
MAS	MAS 5.0 location dependent background	
IdealMM	Ideal Mismatch	
MASIM	MAS 5 and Ideal Mismatch	

Normalization methods		
Option	Name	Description
quantile (default)	Quantile normalization	
scaling	scaling normalization	

Summarization methods		
Option	Name	Description
<code>median.polish</code> (default)	Median polish	
<code>tukey.biweight</code>	1 step tukey biweight	
<code>average.log</code>	Average of Logs	
<code>log.average</code>	Log of Average	
<code>median.log</code>	Median of Logs	
<code>log.median</code>	Log of Median	
<code>log.2nd.largest</code>	2nd largest PM	
<code>lm</code>	Linear Model	
<code>rlm</code>	Robust Linear Model	