Quick start guide for CALIB package

Hui Zhao
CMPG, K.U.Leuven, Belgium
Kristof Engelen
CMPG, K.U.Leuven, Belgium
Bart DeMoor
ESAT-SCD, K.U.Leuven, Belgium
Kathleen Marchal
CMPG, K.U.Leuven, Belgium

July, 2006

Contents

1 Overview 1
2 Classes 2
3 Work flow 2

1 Overview

The CALIB package provides a novel normalization method for normalizing spotted microarray data. The methodology is based on a physically motivated model, consisting of two major components:

- hybridization reaction.
- dye saturation function.

Spike-based curves are used to estimate absolute transcript levels for each combination of a gene and a tested biological condition, irrespective of the number of microarray slides or replicate spots on one slide. The CALIB package allows normalizing spotted microarray data, using the method methods mentioned above and also provides different visualization functions that allow quality control and data exploration. This document provides a brief introduction of data classes used in this package and a simple work flow of this package. The work flow contains the following procedure:

- Read in microarray data.
- Perform simple diagnostic functions to access quality of the spikes.
- Estimate parameters of the calibration model.
- Normalization by using the calibration model.
More detailed explanation is available in the other online document of the package called readme.pdf. To reach this readme file, you need to install the CALIB package. If you’ve installed the package, you can type

```r
> library(CALIB)
> calibReadMe()
```

## 2 Classes

Three data classes are used for storing data in the CALIB package.

**RGList_CALIB**: A list used to store raw measurement data after they are read in from an image analysis output file, usually by `read.rg()`. The RGList_CALIB in this package is an extended limma::RGList from the Limma package. As compared to the limma::RGList it contains two additional fields, RArea and GArea. These two additional fields are meant to store the spot areas, which in some cases are needed to calculate measured intensities.

**SpikeList**: A list used to store raw measurement data of all external control spikes spotted on the arrays. An object of this class is created by `read.spike()`. It is a subset of the object of RGList_CALIB plus two fields, RConc and GConc to indicate known concentration for the control spikes’ targets added to the hybridization solution and labeled in red and green respectively.

**ParameterList**: A list used to store parameters of the calibration model for each array. An object of this class is created by `estimateParameter()`.

## 3 Work flow

To load the CALIB package in your R session, type `library(CALIB)`. In order to illustrate the workings and principles of the method and the usage of the functions in the package, we use a test set containing two out of fourteen hybridizations of a publicly available benchmark data set. The experiment design of these two arrays consists of a color-flip of two conditions. The usage of the package is illustrated in this document by means of this test example.

1. To begin, users will create a directory and move all the relevant files to that directory including:

   - The image processing output files (e.g. .txt files).
   - A file contains target (or samples) descriptions (e.g. targets.txt file).
   - A file contains the IDs and other annotation information associated with each probe (e.g. annotation.txt file).
   - A file specifies spot type for each of the different spots on the array (e.g. Spot-Type.txt file).
   - A file contains concentration of each spike (e.g. conc.txt file).
For this illustration, the data has been gathered in the data directory /arraydata.

2. Start R in the desired working directory and load the `CALIB` package.

```r
> library(CALIB)
> path<-system.file("arraydata", package="CALIB")
> dir(system.file("arraydata", package="CALIB"))

[1] "3000177542.txt" "3000177543.txt" "SpotTypes.txt" "annotation.txt"
[5] "conc.txt" "targets.txt"
```

3. **Data input**: Read in the target file containing information about the hybridization.

```r
> datapath <- system.file("arraydata", package="CALIB")
> targets <- readTargets("targets.txt",path=datapath)
> targets

    Name   Cy5   Cy3 FileName
3000177542 array1 Cond1 Cond2 3000177542.txt
3000177543 array2 Cond2 Cond1 3000177543.txt
```

4. Read in the raw fluorescent intensities data, by default we assume that the file names are provided in the *first* column of the target file with the column name of `FileName`.

```r
> RG <- read.rg(targets$FileName,columns=list(Rf="CH1_NBC_INT",Gf="CH2_NBC_INT",Rb="CH1_SPOT_BKGD",Gb="CH2_SPOT_BKGD",RArea="CH1_SPOT_AREA",GArea="CH2_SPOT_AREA"),path=datapath)

Read /tmp/Rtmpmf07CQ/Rinst144e643f2c1b/CALIB/arraydata/3000177542.txt
Read /tmp/Rtmpmf07CQ/Rinst144e643f2c1b/CALIB/arraydata/3000177543.txt
```

5. Read in the probe annotation information.

```r
> filename <- "annotation.txt"
> fullname <- file.path(datapath,filename)
> annotation <- read.table(file=fullname,header=T,fill=T,quote="",sep="\t")
> RG$genes <- annotation
```

6. Read in the spot type information.

```r
> types<-readSpotTypes(path=datapath)
> types

    SpotType SOURCE_CLONE_ID ORIGIN Color
1      cDNA      CATMA*   * black
2        Ratio   rYIR*   APB orange
3  Calibration   cYIR*   APB   red
4    Negative   nYIR*   APB   blue
5      Utility   uYIR*   APB   green

> spotstatus<-controlStatus(types,RG$genes)
```
Matching patterns for: SOURCE_CLONE_ID ORIGIN
Found 18981 cDNA
Found 192 Ratio
Found 480 Calibration
Found 24 Negative
Found 72 Utility
Setting attributes: values Color

> RG$genes$Status<-spotstatus

7. Read in concentration of spikes.

> concfile="conc.txt"
> spike<-read.spike(RG,file=concfile,path=datapath)

8. **Spike quality assessment:** the following command generates diagnostic plots for a assessment of spike quality.

> arraynum <- 1
> plotSpikeCI(spike,array=arraynum)

From Figure 1 a sigmoidal relationship between the measured intensities and added concentrations is to be expected. Indeed, in a certain range the relationship will be linear, but at the highest and lowest concentration levels saturation effects will occur, which might be different for the red and green channel.

9. **Parameter Estimation:** estimate calibration model parameters array by array.
Figure 2: Estimated calibration model parameters

```r
> parameter<-estimateParameter(spike, RG, bc=F, area=T, errormodel="M")
```

10. Generate diagnostics and visualization for the calibration models.

```r
> plotSpikeHI(spike, parameter, array=arraynum)
```

In Figure 2 the red and green curves represent the estimated calibration models for the red and green channel respectively. In general, the more tight and smooth (no visible artifacts) the black dots fit the model curves, the more suitable the model is for further normalization.

11. **Normalization:** Once the calibration models for the red and green channels have been estimated for each array, they can be used to normalize the data. Absolute expression levels for each combination of a gene and condition in the experiment design, regardless of the number of replicates. Experimental design of arrays is specified by three equal length vectors `array, condition` and `dye`.

```r
> array<-c(1,1,2,2)
> condition<-c(1,2,2,1)
> dye<-c(1,2,1,2)
> idcol<"CLONE_ID"
> ## here, we normalize the first ten genes as example.
> cloneid<-RG$genes[1:10, idcol]
> normdata<-normalizeData(RG, parameter, array, condition, dye, idcol=idcol, cloneid=cloneid)
> normdata
```

```
   1  2
210496 7.243015 4.4040895
```
References


Note: This document was generated using the Sweave function from the R tools package. The source file is in the /doc directory of the package CALIB.