The timecourse Package

Yu Chuan Tai

April 30, 2008

Institute for Human Genetics, University of California, San Francisco
`taiy@humgen.ucsf.edu`

Contents

1 Overview **1**

2 Longitudinal one-sample problem **2**
2.1 Data 2
2.2 Genes of interest 2

3 Longitudinal two-sample problem **3**
3.1 Data 3
3.2 Genes of interest: paired two-sample problem 4
3.3 Genes of interest: independent two-sample problem 4

4 Longitudinal multi-sample problem **5**
4.1 Data 5
4.2 Genes of interest 5

5 The amount of moderation **5**

6 Which Statistic? **6**

7 Missing data **6**

8 Outliers **6**

9 Acknowledgements **6**

10 Temporal profile plots **7**

1 Overview

The `timecourse` package aims to serve as a comprehensive library for the analysis of developmental microarray time course data. The current version includes functions for identifying genes of interest from longitudinal replicated developmental microarray time course

experiments with one or more biological conditions. The main functions are **mb.long()** and **mb.MANOVA()**. They calculate the MB-statistics and/or $\widetilde{T}^2$ statistics derived in Tai and Speed (2006) and Tai (2005), using multivariate empirical Bayes approaches. The input object for **mb.long()** and **mb.MANOVA()** can be a `matrix`, `MAList`, `marrayNorm`, or `ExpressionSet`, containing properly preprocessed log_2 expression values or ratios.
The multivariate empirical Bayes models proposed in Tai and Speed (2006) and Tai (2005) have the advantage over the traditional F-statistic in that they incorporate replicate variances, the correlations among gene expression time point samples from longitudinal data, and moderation borrowing the information across genes into the analysis to reduce the numbers of false positives and false negatives induced by those poorly estimated variance-covariance matrices. Please see Tai and Speed (2006), Tai and Speed (2005), and Tai (2005) for more details. The results from **mb.long()** and **mb.MANOVA()** are stored in a list object of class `MArrayTC`, which contains various useful information and is used as the input object for the plotting function **plotProfile()**. `MArrayTC` is a subclass of the virtual class `LargeDataObject` defined in package `limma` (Smyth, 2004, 2005) and inherits a show method there. For more information, type

```
> library(timecourse)
```

```
>help("MArrayTC-class") # look at the help files
>help(mb.long)
>help(mb.MANOVA)
```

2 Longitudinal one-sample problem

2.1 Data

The dataset used in the example was generated by Tomancak et al. (2002). Samples from *Drosophila* embryos were collected and hybridized onto Affymetrix chips at hours 1 to 12 post egg laying. The same procedure was conducted on 3 different days, yielding 3 replicates and 12 time points. Since mRNA samples at different times were extracted from different embryos, it is not a *true* longitudinal design. However, since parallel and identically treated embryos within each experiment were very similar, sampling these embryos within each experiment at different time points may approximate a longitudinal study. Therefore, we treat this dataset as longitudinal and proceed with the one-sample statistic calculated from **mb.long()**.
The dataset has been preprocessed using RMA algorithm (Irizarry et al., 2003; Bolstad et al., 2003) and log_2 expression values were extracted from the RMA output. To reduce the computational time, we only include the first 2000 probesets in our demonstration.

2.2 Genes of interest

Suppose we are interested in genes which change over time, the following codes give it a quick start:
1. Assign time and replicate group to each array:
The default of **mb.long()** assumes the columns (arrays) of the expression matrix from the input object are arranged in the order of biological conditions, replicates and times. For the one-sample problem, we only have 1 biological condition, so the function assumes the columns are arranged in the order of replicates and times. Hence, if the 3 replicates are named as `A`,

B, and C, the following column arrangement of the matrix fruitfly will be the same as the default.

```
> data(fruitfly)
> dim(fruitfly)

[1] 2000   36

> colnames(fruitfly)

 [1] "A_01.cel" "A_02.cel" "A_03.cel" "A_04.cel" "A_05.cel" "A_06.cel"
 [7] "A_07.cel" "A_08.cel" "A_09.cel" "A_10.cel" "A_11.cel" "A_12.cel"
[13] "B_01.cel" "B_02.cel" "B_03.cel" "B_04.cel" "B_05.cel" "B_06.cel"
[19] "B_07.cel" "B_08.cel" "B_09.cel" "B_10.cel" "B_11.cel" "B_12.cel"
[25] "C_01.cel" "C_02.cel" "C_03.cel" "C_04.cel" "C_05.cel" "C_06.cel"
[31] "C_07.cel" "C_08.cel" "C_09.cel" "C_10.cel" "C_11.cel" "C_12.cel"

> gnames <- rownames(fruitfly)
```

The number of time points and sample sizes are always required as inputs. If the arrays are not in the default order, one needs to assign the time and replicate groups for each array. In our example, they are in the default order, so we make the assignments just for demo purpose.

```
> assay <- rep(c("A", "B", "C"), each = 12)
> time.grp <- rep(c(1:12), 3)
> size <- rep(3, 2000)
```

2. Calculate the MB-statistics and $\widetilde{T}^2$ statistics, and plot individual genes of interest.

```
> out1 <- mb.long(fruitfly, times = 12, reps = size)

Time group assignments are set to default.
Replicate group assignments are set to default.

> out2 <- mb.long(fruitfly, times = 12, reps = size, rep.grp = assay,
+     time.grp = time.grp)
```

3 Longitudinal two-sample problem

3.1 Data

Now we simulate a dataset with two biological conditions. There are 1000 genes in total, and 20 of them have the same expected time course profiles between these two biological conditions. Each gene has 3 time course replicates and 5 time points for each condition. Note that this simulation is for demonstration purpose only, and does not necessarily reflect the real features of longitudinal time course data.

```
> SS <- matrix(c(0.01, -8e-04, -0.003, 0.007, 0.002, -8e-04, 0.02,
+     0.002, -4e-04, -0.001, -0.003, 0.002, 0.03, -0.005, -0.009,
+     0.007, -4e-04, -0.005, 0.02, 8e-04, 0.002, -0.001, -0.009,
+     8e-04, 0.07), ncol = 5)
> sim.Sigma <- function() {
+     S <- matrix(rep(0, 25), ncol = 5)
+     x <- mvrnorm(n = 10, mu = rep(0, 5), Sigma = 10 * SS)
+     for (i in 1:10) S <- S + crossprod(t(x[i, ]))
+     solve(S)
+ }
> sim.data2 <- function(x, indx = 1) {
+     mu <- rep(runif(1, 8, x[1]), 5)
+     if (indx == 1)
+         res <- c(as.numeric(t(mvrnorm(n = 3, mu = mu + rnorm(5,
+             sd = 5), Sigma = sim.Sigma()))), as.numeric(t(mvrnorm(n = 3,
+             mu = mu + rnorm(5, sd = 3.2), Sigma = sim.Sigma()))))
+     if (indx == 0)
+         res <- as.numeric(t(mvrnorm(n = 6, mu = mu + rnorm(5,
+             sd = 3), Sigma = sim.Sigma())))
+     res
+ }
> M2 <- matrix(rep(14, 1000 * 30), ncol = 30)
> M2[1:20, ] <- t(apply(M2[1:20, ], 1, sim.data2))
> M2[21:1000, ] <- t(apply(M2[21:1000, ], 1, sim.data2, 0))
```

3.2 Genes of interest: paired two-sample problem

Assume it is a paired two-sample problem, we can do the following. Note that, R is case-sensitive and since `mt` < `wt` , the first column of the input argument `reps` should be the sample sizes of all genes for `mt`, while the second column is those for `wt`. The same rule applies for the input argument `size` for **mb.MANOVA()**.

```
> trt <- rep(c("wt", "mt"), each = 15)
> assay <- rep(rep(c("rep1", "rep2", "rep3"), each = 5), 2)
> size <- matrix(3, nrow = 1000, ncol = 2)
> MB.paired <- mb.long(M2, method = "paired", times = 5, reps = size,
+     condition.grp = trt, rep.grp = assay)
```

```
Time group assignments are set to default.
```

```
> genenames <- as.character(1:1000)
```

3.3 Genes of interest: independent two-sample problem

Now assume it is an independent two-sample problem

```
> MB.2D <- mb.long(M2, method = "2", times = 5, reps = size, condition.grp = trt,
+     rep.grp = assay)
```

```
Time group assignments are set to default.
```

4 Longitudinal multi-sample problem

4.1 Data

Now let's simulate a dataset with three biological conditions 500 genes in total, 10 of them have different expected time course profiles across biological conditions: the first condition has 3 replicates, while the second condition has 4 replicates, and the third condition has 2 replicates. 5 time points for each condition.

```
> sim.data <- function(x, indx = 1) {
+     mu <- rep(runif(1, 8, x[1]), 5)
+     if (indx == 1)
+         res <- c(as.numeric(t(mvrnorm(n = 3, mu = mu + rnorm(5,
+             sd = 5), Sigma = sim.Sigma()))), as.numeric(t(mvrnorm(n = 4,
+             mu = mu + rnorm(5, sd = 3.2), Sigma = sim.Sigma()))),
+             as.numeric(t(mvrnorm(n = 2, mu = mu + rnorm(5, sd = 2),
+                 Sigma = sim.Sigma()))))
+     if (indx == 0)
+         res <- as.numeric(t(mvrnorm(n = 9, mu = mu + rnorm(5,
+             sd = 3), Sigma = sim.Sigma())))
+     res
+ }
> M <- matrix(rep(14, 500 * 45), ncol = 45)
> M[1:10, ] <- t(apply(M[1:10, ], 1, sim.data))
> M[11:500, ] <- t(apply(M[11:500, ], 1, sim.data, 0))
```

4.2 Genes of interest

The following codes show an example of identifying genes with different temporal profiles across biological conditions:

```
> assay <- rep(c("1.2.04", "2.4.04", "3.5.04", "5.21.04", "7.17.04",
+     "9.10.04", "12.1.04", "1.2.05", "4.1.05"), each = 5)
> trt <- c(rep(c("wildtype", "mutant1"), each = 15), rep("mutant1",
+     5), rep("mutant2", 10))
> size <- matrix(c(4, 2, 3), byrow = TRUE, nrow = 500, ncol = 3)
> MB.multi <- mb.MANOVA(M, times = 5, D = 3, size = size, rep.grp = assay,
+     condition.grp = trt)
```

```
Time group assignments are set to default.
```

5 The amount of moderation

It is always a good idea to check the amount of moderation, which is determined by the diagonal elements of the (transformed) gene-specific (pooled) variance-covariance matrices. The more homogenous these diagonal elements are across genes, the larger the amount of moderation will occur. Try different values of the input argument **prior.df** to see what happens.

6 Which Statistic?

In the one- and two- sample longitudinal problems, when the sample size(s) are *different* across genes, the MB-statistics are the ones you should use for ranking. Otherwise, the MB-statistics and $\widetilde{T}^2$ produce the same ranking.

7 Missing data

For version 1.0.0, missing time point samples are not allowed in **mb.long()** and **mb.MANOVA()**, while missing replicates are allowed for a subset of genes. In the latter case, the user still need to input a complete data matrix with missing replicates indicated by NAs.

```
> fruitfly[6, 13:24] <- NA
> size <- rep(3, 2000)
> size[6] <- 2
> MB.missing <- mb.long(fruitfly, times = 12, reps = size, HotellingT2.only = FALSE)
```

```
Time group assignments are set to default.
Replicate group assignments are set to default.
```

8 Outliers

When type=robust, the numerator of the $\widetilde{T}^2$ statistic is calculated using the weighted average time course vector(s), where the weight for each data point is determined by Huber's weight function (Huber, 2004; Fox, 2002) with the default tuning constant 1.345.

When there are only 2 replicates within conditions, type=robust produces the same rankings as type=none since there is no concensus on gene expression values. We recommend that at least 3 replicates should be performed when designing an experiment so that outliers can be determined by concensus. Check the output weights for these outliers.

9 Acknowledgements

The `timecourse` package has greatly benefited from many critical comments by Terry Speed. The *Drosophila* data were provided by Pavel Tomancak and his colleagues. Thanks are also due to Ben Bolstad for his advices on building up this package, and Karen Vranizan, Yun Zhou, Jun Li and their Pritzker colleagues, and Soyeon Ahn for using the test version of this package and their helpful feedbacks.

References

B. Bolstad, R. Irizarry, M. Astrand, and T. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003. URL `http://bioinformatics.oupjournals.org/cgi/content/abstract/19/2/185`.

J. Fox. Robust regression: an appendix to an R and S-PLUS companion to applied regression, 2002.

P. J. Huber. *Robust Statistics.* Wiley series in probability and mathematical statistics, New York, 2004.

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. URL `http://nar.oupjournals.org/cgi/content/abstract/31/4/e15`.

G. K. Smyth. Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3 (1):article 3, 2004.

G. K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions using R and Bioconductor*. Springer, New York, 2005.

Y. C. Tai. *Multivariate empirical Bayes models for replicated microarray time course data.* PhD thesis, Division of Biostatistics, University of California, Berkeley, May 2005.

Y. C. Tai and T. P. Speed. Statistical analysis of microarray time course data. In U. Nuber, editor, *DNA Microarrays.* BIOS Scientific Publishers Limited, Taylor & Francis, 4 Park Square, Milton Park, Abingdon OX14 4RN, 2005. Chapter 20.

Y. C. Tai and T. P. Speed. A multivariate empirical Bayes statistic for replicated microarray time course data. *Annals of Statistics*, 34(5):2387–2412, 2006.

P. Tomancak, A. Beaton, R. Weiszmann, E. Kwan, S. Shu, S. Lewis, S. Richards, M. Ashburner, V. Hartenstein, S. Celniker, and G. Rubin. Systematic determination of patterns of gene expression during drosophila embryogenesis. *Genome Biology*, 3(12):research0088.1–0088.14, 2002. ISSN 1465-6906. URL `http://genomebiology.com/2002/3/12/research/0088`. The microarray time course data can be downloaded from `www.fruitfly.org/cgi-bin/ex/insitu.pl`.

10 Temporal profile plots

The temporal profile(s) for any single gene can be plotted using the `plotProfile()` with the resulting `MArrayTC` object as the input, see below for examples. The user can either specify the rank or the gene ID of the gene to be plotted.

```
> plotProfile(out2, type = "b", gnames = gnames, legloc = c(2,
+     15), pch = c("A", "B", "C"), xlab = "Hour")
```

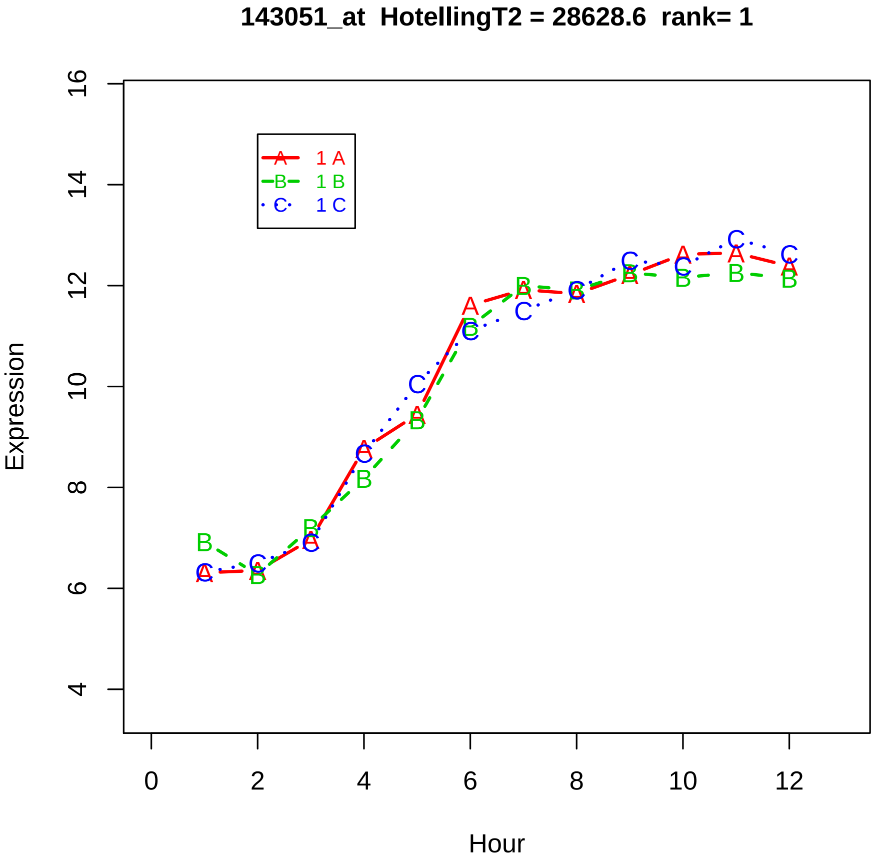

Figure 1: The no. 1 gene of *fruitfly*.

```
> plotProfile(out2, type = "b", gnames = gnames, pch = c("A", "B",
+     "C"), xlab = "Hour", ranking = 100)
```

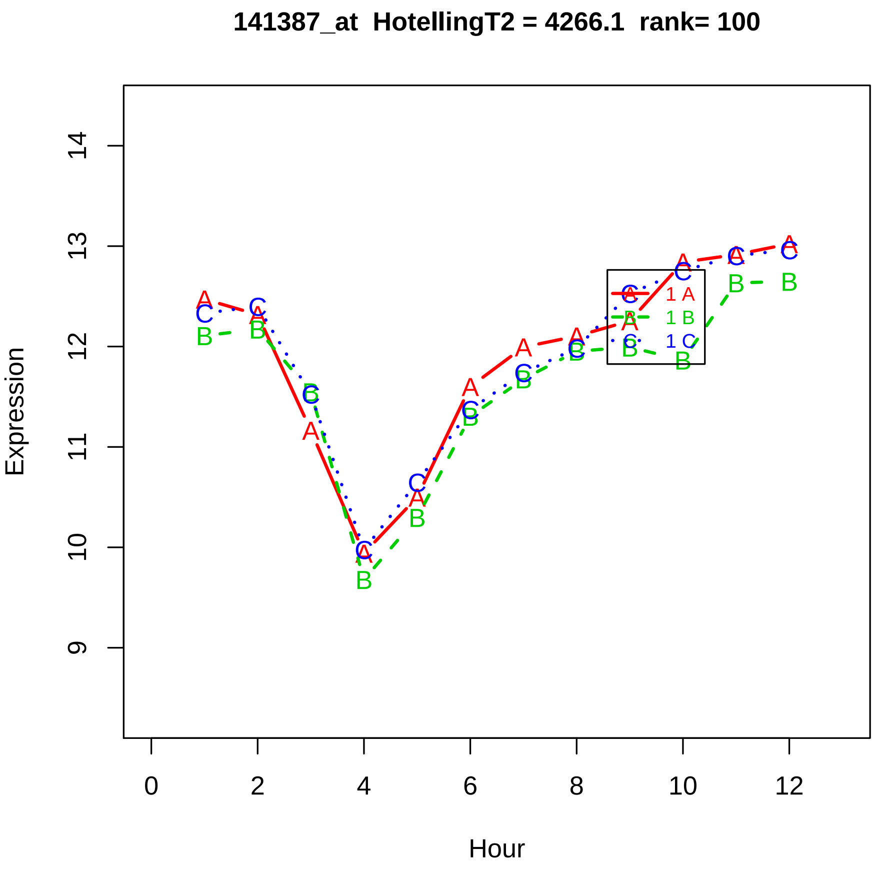

Figure 2: The no. 100 gene of *fruitfly*

```
> plotProfile(out2, type = "b", gnames = gnames, pch = c("A", "B",
+     "C"), xlab = "Hour", gid = "141404_at")
```

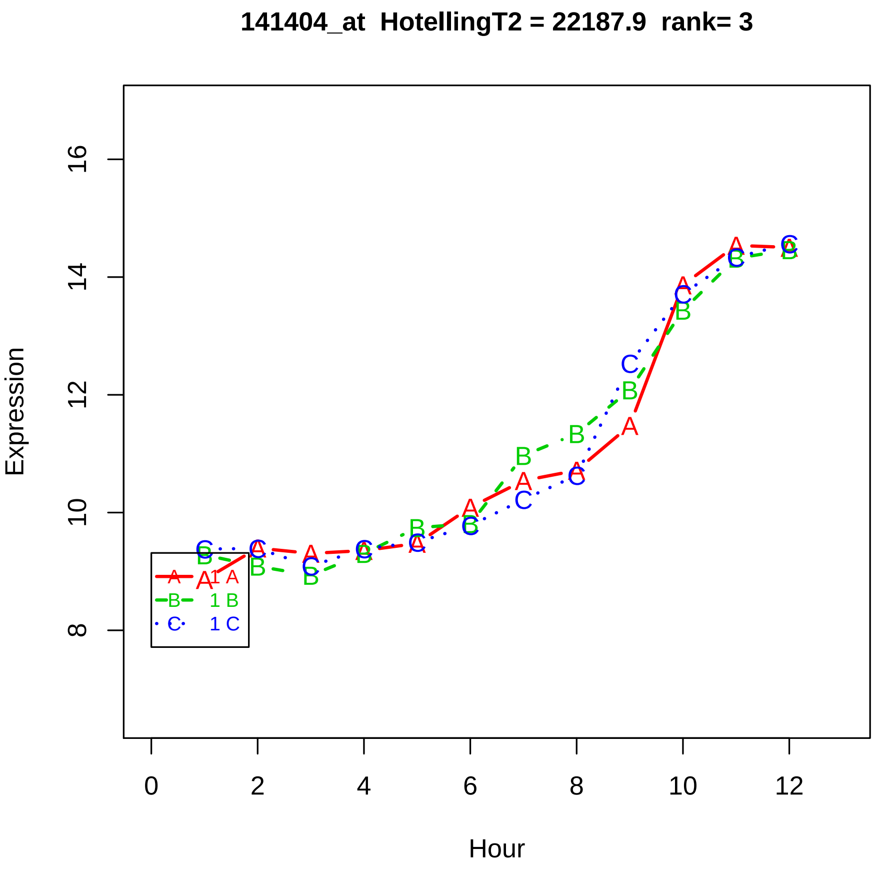

Figure 3: Gene 141404_at in *fruitfly*

```
> plotProfile(MB.paired, type = "b", gnames = genenames)
```

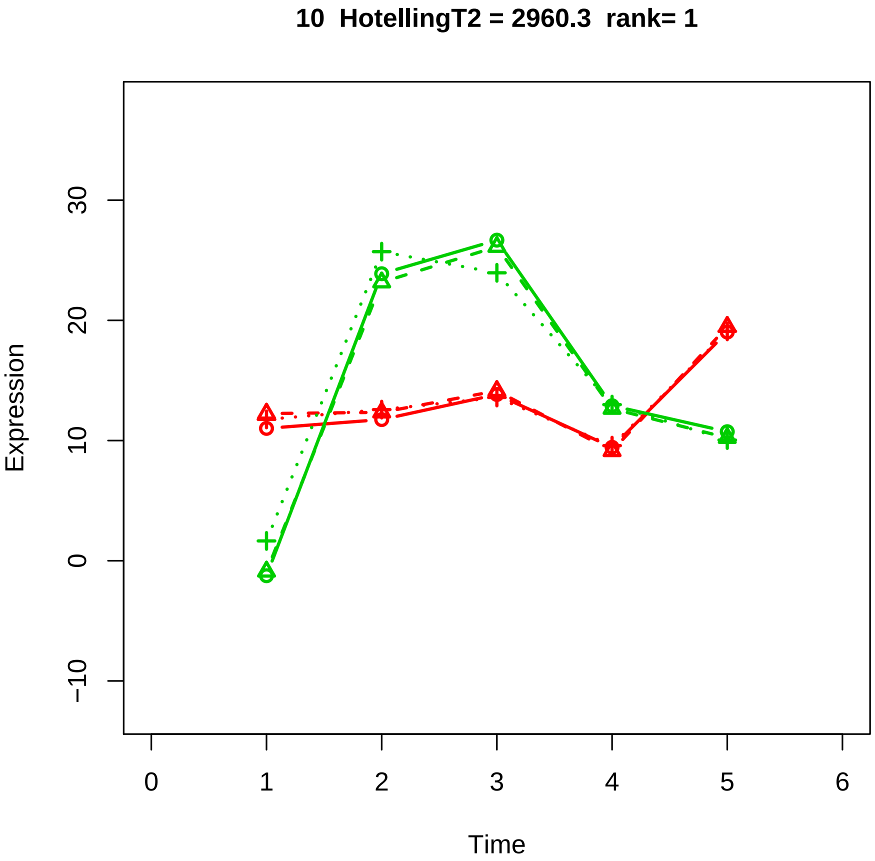

Figure 4: The no. 1 gene for paired two-sample problem.

```
> plotProfile(MB.2D, type = "b", gnames = genenames)
```

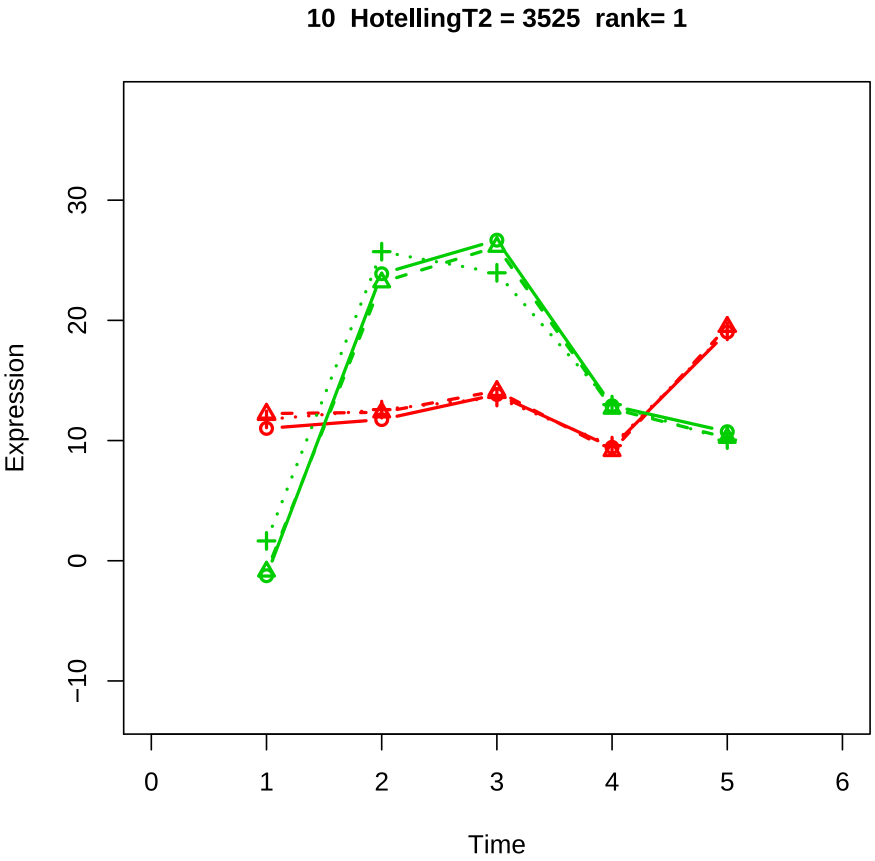

Figure 5: The no. 1 gene for independent two-sample problem.

```
> plotProfile(MB.multi, stats = "MB", type = "b")
```

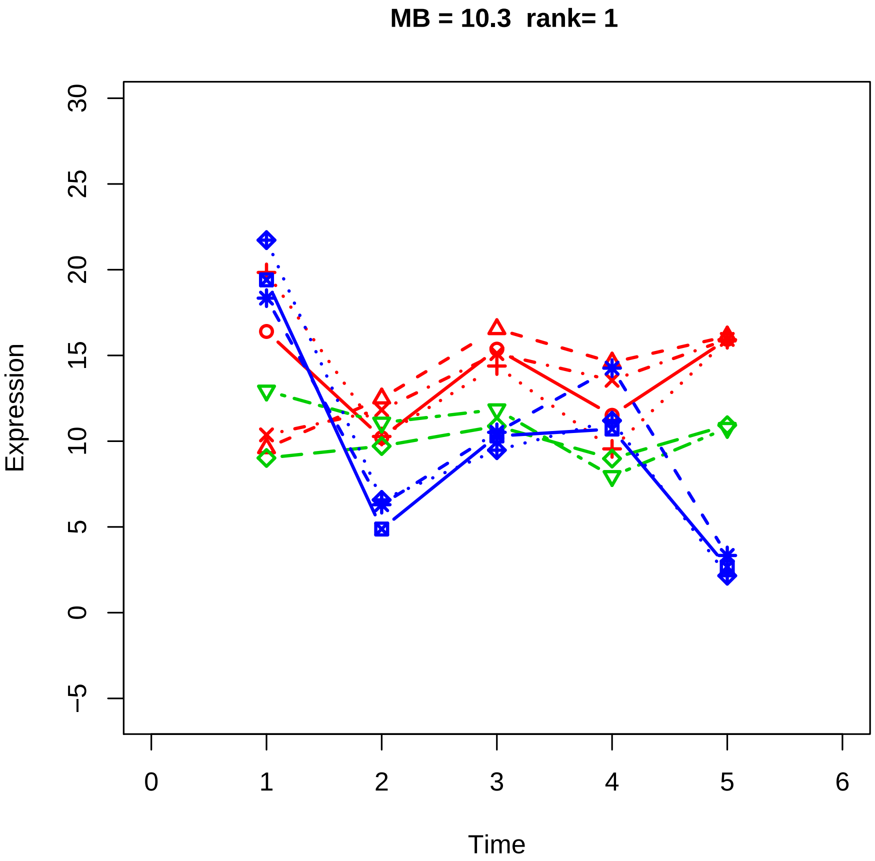

Figure 6: The no. 1 gene for multi-sample problem.

```
> plotProfile(MB.missing, stats = "MB", type = "b", gid = "141205_at",
+     gnames = gnames, pch = c("A", "B", "C"))
```

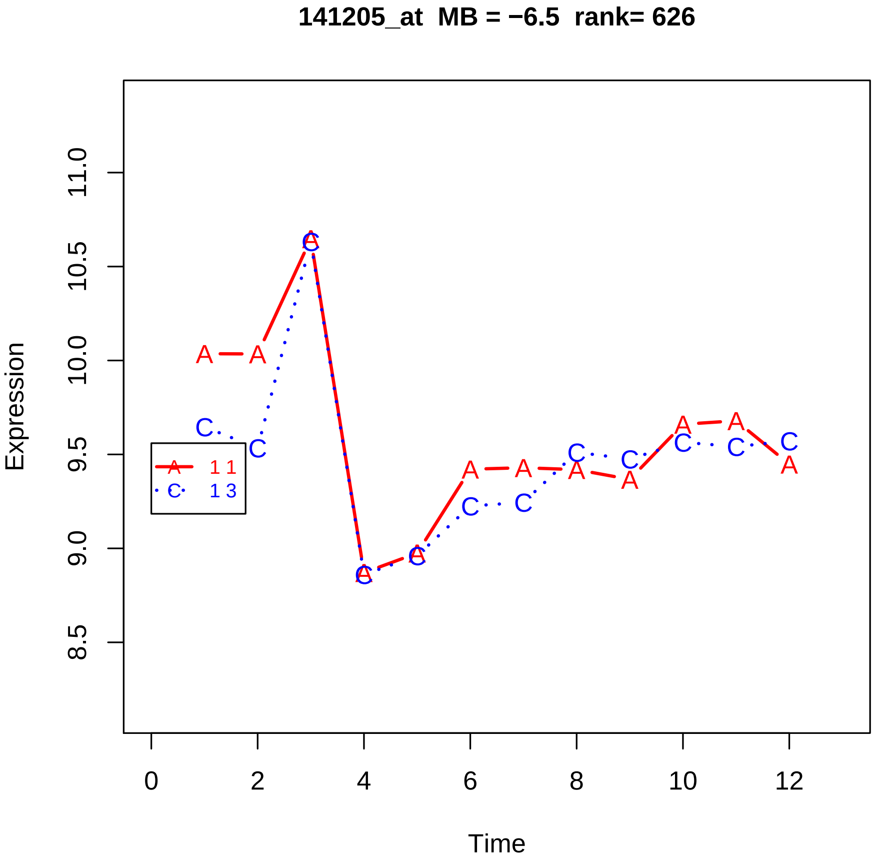

Figure 7: The gene 141205_at in fruitfly: one replicate is missing.