Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

June 27, 2008

The following R code downloads the microarray data from GEO and processes it to create the AffyBatch objects.

1 Getting started

```
> library(affy)
```

The following functions are hacks based on an older version of the GEOquery package:

```
> parseGeoMeta <- function(txt) {
+     leader <- strsplit(grep("!\\w*_", txt, perl = TRUE, value = TRUE)[1],
+         "_")[[1]][1]
+     tmp <- txt[grep(leader, txt)]
+     tmp <- gsub(paste(leader, "_", sep = ""), "", tmp)
+     first.eq <- regexpr(" = ", tmp)
+     tmp <- cbind(substring(tmp, first = 1, last = first.eq -
+         1), substring(tmp, first = first.eq + 3))
+     header <- split(tmp[, 2], tmp[, 1])
+     return(header)
+ }
> getMetaGEO <- function(GEOAccNum, drop = TRUE) {
+     pre <- "http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc="
+     post <- "&targ=gsm&form=text&view=brief"
+     txt <- readLines(paste(pre, GEOAccNum, post, sep = ""))
+     meta <- parseGeoMeta(txt)
+     lng <- sapply(meta, length)
+     meta2 <- meta[lng == median(lng)]
+     if (drop) {
+         n.levels <- sapply(meta2, function(x) length(levels(factor(x))))
+         meta2 <- meta2[n.levels > 1]
+     }
+     as.data.frame(meta2)
+ }
```

2 GSE11812: cell lines

```
> system(paste("curl -O ", "ftp://ftp.ncbi.nih.gov/pub/geo/DATA/supplementary/",
+     "series/GSE11812/GSE11812_RAW.tar", sep = ""))
> system("tar xf GSE11812_RAW.tar")

> gse11812.meta <- getMetaGEO("GSE11812", drop = TRUE)
> rownames(gse11812.meta) <- paste(gse11812.meta$geo_accession,
+     ".CEL.gz", sep = "")
> all(rownames(gse11812.meta) %in% dir())

[1] TRUE

> gse11812.p <- gse11812.meta[, c(1, 2, 5)]
> gse11812.batch <- ReadAffy(filenames = rownames(gse11812.p),
+     phenoData = as(gse11812.p, "AnnotatedDataFrame"), notes = "GSE11812, cell lines from
> save(gse11812.batch, file = "gse11812.batch.RData")
> gse11812.batch

AffyBatch object
size of arrays=712x712 features (24 kb)
cdf=HG-U133A (22283 affyids)
number of samples=30
number of genes=22283
annotation=hgu133a
notes= GSE11812, cell lines from B. Gyorffy et al
```

3 GSE4183: colon biopsies

```
> system(paste("curl -O ", "ftp://ftp.ncbi.nih.gov/pub/geo/DATA/supplementary/",
+     "series/GSE4183/GSE4183_RAW.tar", sep = ""))
> system("tar xf GSE4183_RAW.tar")

> gse4183.meta <- getMetaGEO("GSE4183", drop = TRUE)
> rownames(gse4183.meta) <- paste(gse4183.meta$geo_accession, ".CEL.gz",
+     sep = "")
> all(rownames(gse4183.meta) %in% dir())

[1] TRUE

> gse4183.p <- gse4183.meta[, c(1, 2, 4)]
> gse4183.batch <- ReadAffy(filenames = rownames(gse4183.p), phenoData = as(gse4183.p,
+     "AnnotatedDataFrame"), notes = "GSE4183, colon biopsies from B. Gyorffy et al")
> save(gse4183.batch, file = "gse4183.batch.RData")
> gse4183.batch

AffyBatch object
size of arrays=1164x1164 features (29 kb)
```

```
cdf=HG-U133_Plus_2 (54675 affyids)
number of samples=53
number of genes=54675
annotation=hgu133plus2
notes= GSE4183, colon biopsies from B. Gyorffy et al
```

4 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()

R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e

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

other attached packages:
[1] hgu133plus2cdf_2.0.0 hgu133acdf_2.0.0     affy_1.16.0
[4] preprocessCore_1.0.0 affyio_1.6.1         Biobase_1.16.3
```

Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

June 29, 2008

The following R code processes the raw microarray probes intensities to generate expression values, using various normalization algorithms.

1 Getting started

```
> library(affy)
> library(farms)
> library(gcrma)
> library(plier)
> library(bgx)
> library(vsn)
```

Define some functions for those algorithms that have not been implemented as a simple R function.

```
> mbei <- function(x) expresso(x, normalize.method = "invariantset",
+     bg.correct = FALSE, pmcorrect.method = "pmonly", summary.method = "liwong")
> plierPlus16 <- function(x) {
+     e <- justPlier(x, normalize = TRUE)
+     exprs(e) <- log2((2^exprs(e)) + 16)
+     e
+ }
> source("dfwcode.R")
> dfw <- function(x) {
+     expresso(x, bgcorrect.method = "none", normalize.method = "quantiles",
+         pmcorrect.method = "pmonly", summary.method = "dfw")
+ }
```

Since it may be interesting to compare computation times for the various normalization algorithms, we will try to make the memory situations similar for each algorithm. Thus, we will perform garbage collection beforehand, and remove the results from memory afterwards.

```
> timeNormalize <- function(abatch, fn, name) {
+     gc(verbose = FALSE)
+     out <- system.time(assign(name, fn(abatch)))
+     save(name, file = paste(name, ".RData", sep = ""))
+     rm(name)
+     out
+ }
```

2 GSE11812: cell lines

2.1 raw data preparation

Load the raw data as available on the GEO site.

```
> load("gse11812.batch.RData")
```

Retain only those samples for which we have RT-PCR data.

```
> cells.pcr <- as.matrix(read.delim("Supplemental table 1_cell line rtpcr raw data.txt",
+     row.names = 1))
> dim(cells.pcr)

[1] 96 29

> dim(exprs(gse11812.batch))

[1] 506944     30

> cells.batch <- gse11812.batch[, match(colnames(cells.pcr), gse11812.batch$geo)]
> all(cells.batch$geo == colnames(cells.pcr))

[1] TRUE
```

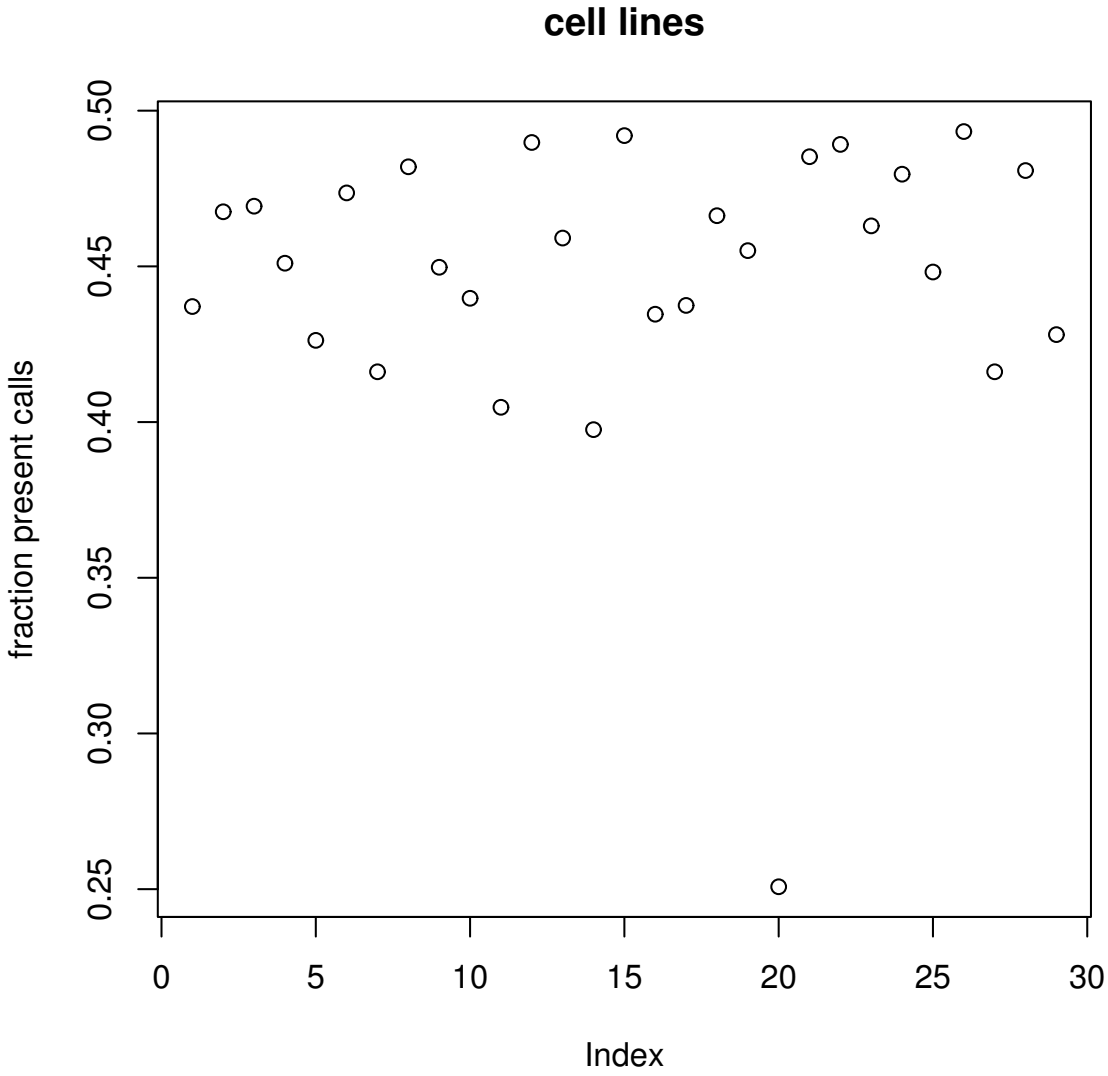
Check for low-quality samples (based on the fraction of present calls for each sample).

```
> cells.mas5calls <- mas5calls(cells.batch)

Getting probe level data...
Computing p-values
Making P/M/A Calls

> cells.present <- colMeans(exprs(cells.mas5calls) == "P")

> plot(cells.present, ylab = "fraction present calls", main = "cell lines")
```



Sample #20 is a clear outlier, so remove it.

```
> cells.batch <- cells.batch[, -20]
> dim(exprs(cells.batch))

[1] 506944     28

> save(cells.batch, file = "cells.batch.RData")
```

"cells.batch" is our working raw data set.
Remove clutter.

```
> rm(gse11812.batch, cells.mas5calls)
```

2.2 normalization

Now process the raw data with the various normalization algorithms.

```
> timeNormalize(cells.batch, mas5, "cells.mas5")

background correction: mas
PM/MM correction : mas
expression values: mas
background correcting...done.
22283 ids to be processed
|                    |
|####################|
    user   system  elapsed
1974.750    0.776 1976.244

> timeNormalize(cells.batch, rma, "cells.rma")

Background correcting
Normalizing
Calculating Expression
   user  system elapsed
 62.791   1.667  64.484

> timeNormalize(cells.batch, gcrma, "cells.gcrma")

Adjusting for optical effect.............................Done.
Computing affinities.Done.
Adjusting for non-specific binding.............................Done.
Normalizing
Calculating Expression
   user  system elapsed
519.584   8.661 529.349

> timeNormalize(cells.batch, mbei, "cells.mbei")

normalization: invariantset
PM/MM correction : pmonly
expression values: liwong
normalizing...done.
22283 ids to be processed
|                    |
|####################|
    user   system  elapsed
1577.821   15.627 1594.185

> timeNormalize(cells.batch, justPlier, "cells.plier")

   user  system elapsed
260.726   0.378 261.582
```

```
> timeNormalize(cells.batch, plierPlus16, "cells.plierPlus16")

Quantile normalizing...Done.
   user  system elapsed
317.979   0.486 318.801

> timeNormalize(cells.batch, q.farms, "cells.qfarms")

background correction: none
normalization: quantiles
PM/MM correction : pmonly
expression values: farms
background correcting...done.
normalizing...done.
22283 ids to be processed
|                    |
|####################|
   user  system elapsed
554.136   0.405 555.142

> timeNormalize(cells.batch, dfw, "cells.dfw")

background correction: none
normalization: quantiles
PM/MM correction : pmonly
expression values: dfw
background correcting...done.
normalizing...done.
22283 ids to be processed
|                    |
|####################|
   user  system elapsed
398.555   0.551 399.755

> timeNormalize(cells.batch, vsnrma, "cells.vsn")

Calculating Expression
   user  system elapsed
 77.587   0.713  78.456
```

3 GSE4183: colon biopsies

3.1 raw data preparation

```
> load("gse4183.batch.RData")
```

Retain only those samples for which we have RT-PCR data.

```
> colon.pcr <- as.matrix(read.delim("Supplemental table 2_colon rtpcr raw data.txt",
+     row.names = 1))
> dim(colon.pcr)

[1] 96 36

> dim(exprs(gse4183.batch))

[1] 1354896      53

> colon.batch <- gse4183.batch[, match(colnames(colon.pcr), gse4183.batch$geo)]
> all(colon.batch$geo == colnames(colon.pcr))

[1] TRUE
```

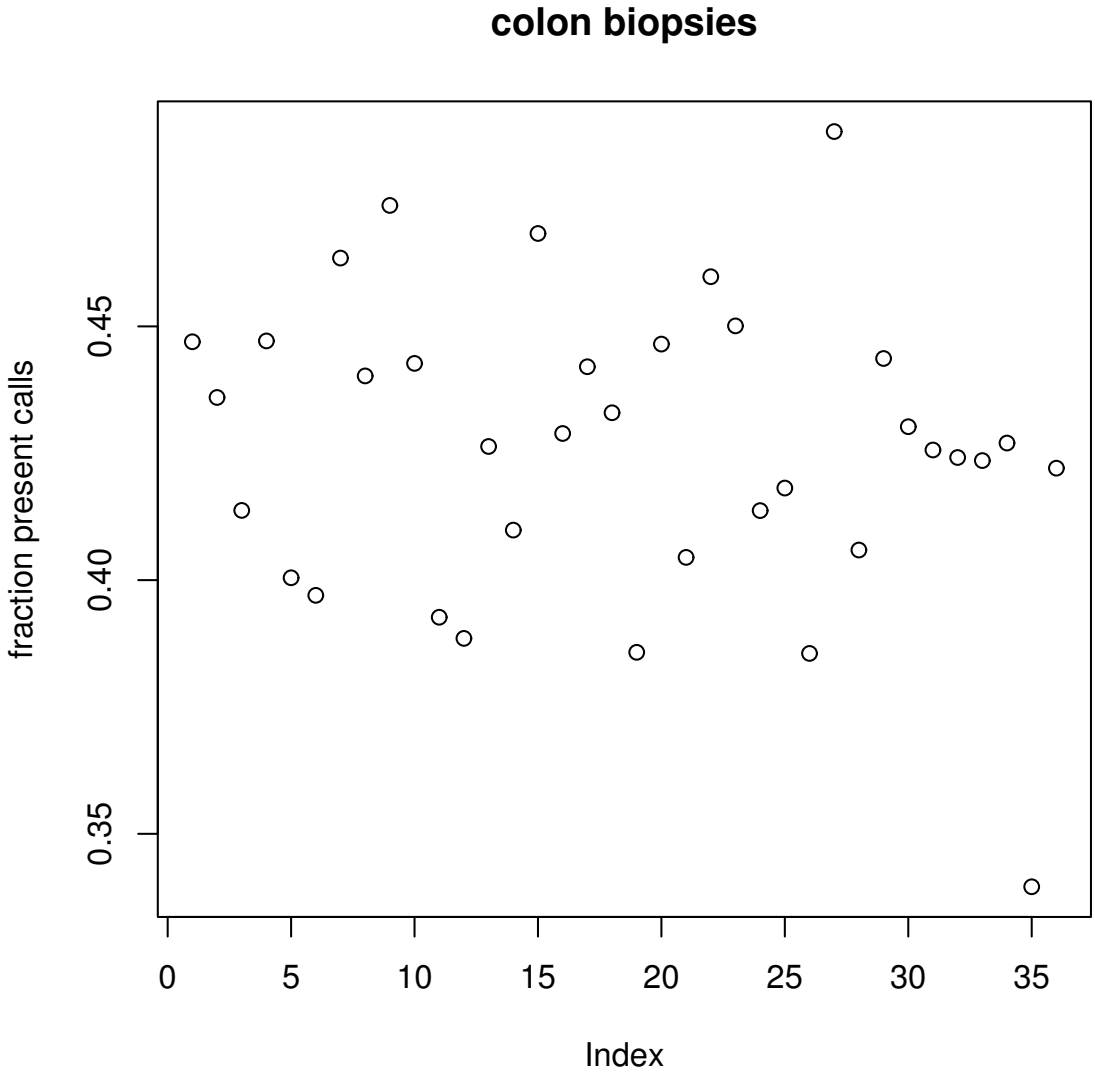
Check for low-quality samples (based on the fraction of present calls for each sample).

```
> colon.mas5calls <- mas5calls(colon.batch)

Getting probe level data...
Computing p-values
Making P/M/A Calls

> colon.present <- colMeans(exprs(colon.mas5calls) == "P")

> plot(colon.present, ylab = "fraction present calls", main = "colon biopsies")
```



Perhaps sample #35 is kind of borderline, but we'll leave it in.

```
> dim(exprs(colon.batch))
```

```
[1] 1354896      36
```

```
> save(colon.batch, file = "colon.batch.RData")
```

"colon.batch" is our working raw data set.
Remove clutter.

```
> rm(gse4183.batch, colon.mas5calls)
```

3.2 normalization

Now process the raw data with the various normalization algorithms.

```
> timeNormalize(colon.batch, mas5, "colon.mas5")

background correction: mas
PM/MM correction : mas
expression values: mas
background correcting...done.
54675 ids to be processed
|                    |
|####################|
    user   system  elapsed
6431.496    2.375 6436.636

> timeNormalize(colon.batch, rma, "colon.rma")

Background correcting
Normalizing
Calculating Expression
   user  system elapsed
177.630   4.882 182.736

> timeNormalize(colon.batch, gcrma, "colon.gcrma")

Adjusting for optical effect.....................................Done.
Computing affinities.Done.
Adjusting for non-specific binding.....................................Done.
Normalizing
Calculating Expression
    user   system  elapsed
1874.221   28.706 1904.747

> timeNormalize(colon.batch, mbei, "colon.mbei")

normalization: invariantset
PM/MM correction : pmonly
expression values: liwong
normalizing...done.
54675 ids to be processed
|                    |
|####################|
    user   system  elapsed
5201.858   63.830 5267.499

> timeNormalize(colon.batch, justPlier, "colon.plier")

   user  system elapsed
837.825   0.772 838.665

> timeNormalize(colon.batch, plierPlus16, "colon.plierPlus16")
```

```
Quantile normalizing...Done.
    user   system  elapsed
1001.807    2.235 1004.080

> timeNormalize(colon.batch, q.farms, "colon.qfarms")

background correction: none
normalization: quantiles
PM/MM correction : pmonly
expression values: farms
background correcting...done.
normalizing...done.
54675 ids to be processed
|                    |
|####################|
    user   system  elapsed
1231.954    1.993 1235.199

> timeNormalize(colon.batch, dfw, "colon.dfw")

background correction: none
normalization: quantiles
PM/MM correction : pmonly
expression values: dfw
background correcting...done.
normalizing...done.
54675 ids to be processed
|                    |
|####################|
   user  system elapsed
992.720   1.843 995.737

> timeNormalize(colon.batch, vsnrma, "colon.vsn")

Calculating Expression
   user  system elapsed
280.609   2.270 282.906
```

4 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()

R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
```

```
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e

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

other attached packages:
 [1] hgu133plus2probe_2.0.0 hgu133plus2cdf_2.0.0   hgu133aprobe_2.0.0
 [4] hgu133acdf_2.0.0       vsn_3.2.1              limma_2.12.0
 [7] bgx_1.2.2              plier_1.8.0            gcrma_2.10.0
[10] matchprobes_1.10.0     farms_1.3              MASS_7.2-38
[13] affy_1.16.0            preprocessCore_1.0.0   affyio_1.6.1
[16] Biobase_1.16.3

loaded via a namespace (and not attached):
[1] grid_2.6.1     lattice_0.17-2

> system("uname -a", intern = TRUE)

[1] "Linux sbiology 2.6.5-7.282-sn2 #1 SMP Tue Aug 29 10:40:40 UTC 2006 ia64 ia64 ia64 GNU
```

Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

June 30, 2008

This R script generates Figure 1, which displays the distribution of three bias metrics in various microarray data sets.

1 Calculate bias metrics

Load raw data and calculate bias metrics for each data set.

```
> library(affy)
> datasets <- unlist(list(LatinSquare133 = "SpikeIn133", LatinSquare95 = "SpikeIn95",
+     Hess_breast = "hess", Wang_breast = "wang", Sotiriou_breast = "sotiriou",
+     Pawitan_breast = "gse1456.u133a", Bild_lung = "bild.lung",
+     Gyorffy_colon = "colon", Gyorffy_cells = "cells"))
> getMetrics <- function(x) {
+     data.frame(degradation = AffyRNAdeg(x)$slope, pm.median = apply(log2(pm(x)),
+         2, median), present = colMeans(exprs(mas5calls(x)) ==
+         "P"))
+ }
> allMetrics <- lapply(datasets, function(x) {
+     batchname <- paste(x, ".batch", sep = "")
+     load(paste(batchname, ".RData", sep = ""))
+     m <- getMetrics(get(batchname))
+     rm(list = batchname)
+     m
+ })
> save(allMetrics, file = "allMetrics.RData")
```

Here is a quick summary of the data sets: number of samples, and the standard deviation of each bias metric:

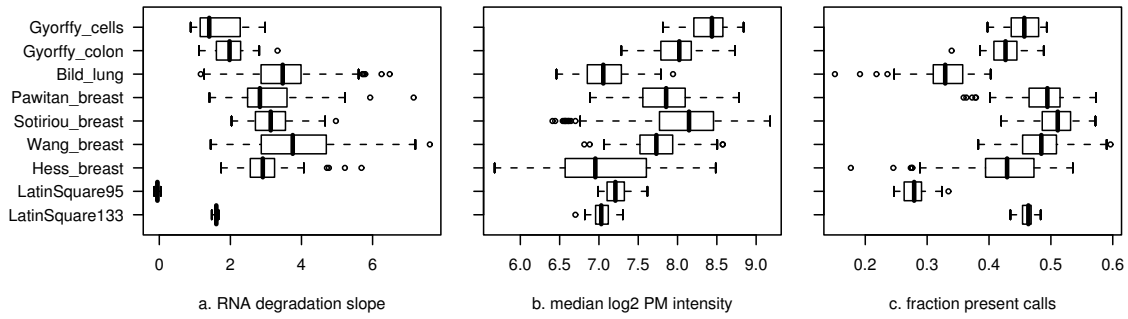
```
> data.frame(n = sapply(allMetrics, nrow), t(sapply(allMetrics,
+     apply, 2, sd)))
```

```
                 n degradation pm.median    present
LatinSquare133  42  0.04856287 0.1288504 0.01241301
LatinSquare95   59  0.04902640 0.1423922 0.02044941
Hess_breast    133  0.59908528 0.6336858 0.06537147
Wang_breast    286  1.22407545 0.3191447 0.03899426
Sotiriou_breast 189  0.61529253 0.6374836 0.03282777
Pawitan_breast  159  0.88055092 0.3874275 0.04037465
Bild_lung       111  0.98951944 0.3067100 0.04298324
Gyorffy_colon    36  0.47342339 0.3467839 0.02921767
Gyorffy_cells    28  0.67439105 0.2501737 0.02775232
```

2 Draw Figure 1

```
> all.degradation <- lapply(allMetrics, function(x) x$degradation)
> all.pm.median <- lapply(allMetrics, function(x) x$pm.median)
> all.present <- lapply(allMetrics, function(x) x$present)

> par(mar = c(5, 1, 2, 1) + 0.1, oma = c(0, 7, 0, 1), las = 1,
+     mfrow = c(1, 3))
> boxplot(all.degradation, horizontal = T, names = names(datasets),
+     xlab = "a. RNA degradation slope")
> boxplot(all.pm.median, horizontal = T, names = NA, xlab = "b. median log2 PM intensity")
> boxplot(all.present, horizontal = T, names = NA, xlab = "c. fraction present calls")
```



3 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()

R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e
```

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

> system("uname -a", intern = TRUE)

[1] "Linux sbiology 2.6.5-7.282-sn2 #1 SMP Tue Aug 29 10:40:40 UTC 2006 ia64 ia64 ia64 GNU
```

Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

June 30, 2008

Generate Figure 2, with which we argue that log-ratio discrepancy is a more intuitive measure of concordance/discordance than is (Pearson/Spearman) correlation.

1 Figure 2

```
> library(affy)
> load("colon.rma.RData")
> colon.rtpcr <- as.matrix(read.delim("Supplemental table 2_colon rtpcr raw data.txt",
+     row.names = 1))
```

18S rRNA normalization of RT-PCR data:

```
> colon.18S <- -scale(colon.rtpcr[-1, ], center = colon.rtpcr[1,
+     ], scale = FALSE)
```

Reality check:

```
> all(colon.rma$geo_accession == colnames(colon.18S))
```

```
[1] TRUE
```

Define the function for log-ratio discrepancy:

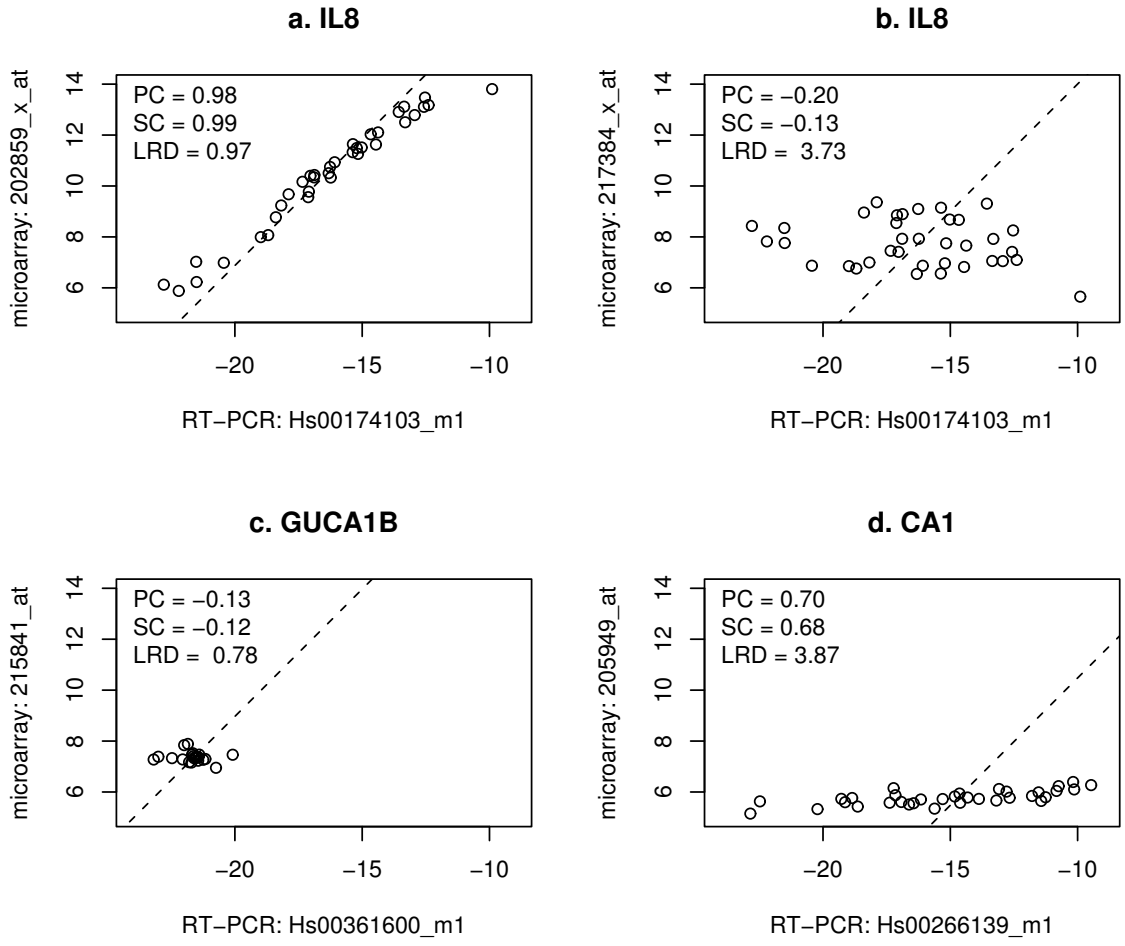
```
> disc <- function(x, y, na.rm = FALSE, fn = mean) {
+     stopifnot(length(x) == length(y))
+     ok <- is.finite(x) & is.finite(y)
+     stopifnot(all(ok) || na.rm)
+     x <- x[ok]
+     y <- y[ok]
+     x.d <- outer(x, x, "-")
+     y.d <- outer(y, y, "-")
+     d <- x.d[upper.tri(x.d)] - y.d[upper.tri(y.d)]
+     fn(abs(d))
+ }
```

Define a function used to draw figures:

```
> showDisc <- function(x, y, ...) {
+     stopifnot(length(x) == length(y))
+     ok <- is.finite(x) & is.finite(y)
+     x <- x[ok]
+     y <- y[ok]
+     plot(x, y, asp = 1, ...)
+     abline(a = median(y) - median(x), b = 1, lty = 2)
+     leg <- paste(c("PC =", "SC =", "LRD ="), format(c(cor(x,
+         y), cor(x, y, method = "spearman"), disc(x, y)), digits = 2))
+     text(-24, 12.5, paste(leg, collapse = "\n"), adj = c(0, 0.5))
+ }
> xlim <- c(-24, -9)
> ylim <- c(5, 14)
```

Now draw the figure.

```
> par(mfrow = c(2, 2))
> showDisc(colon.18S["Hs00174103_m1", ], exprs(colon.rma)["202859_x_at",
+     ], xlab = "RT-PCR: Hs00174103_m1", ylab = "microarray: 202859_x_at",
+     xlim = xlim, ylim = ylim, main = "a. IL8")
> showDisc(colon.18S["Hs00174103_m1", ], exprs(colon.rma)["217384_x_at",
+     ], xlab = "RT-PCR: Hs00174103_m1", ylab = "microarray: 217384_x_at",
+     xlim = xlim, ylim = ylim, main = "b. IL8")
> showDisc(colon.18S["Hs00361600_m1", ], exprs(colon.rma)["215841_at",
+     ], xlab = "RT-PCR: Hs00361600_m1", ylab = "microarray: 215841_at",
+     xlim = xlim, ylim = ylim, main = "c. GUCA1B")
> showDisc(colon.18S["Hs00266139_m1", ], exprs(colon.rma)["205949_at",
+     ], xlab = "RT-PCR: Hs00266139_m1", ylab = "microarray: 205949_at",
+     xlim = xlim, ylim = ylim, main = "d. CA1")
```



2 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()

R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e

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

other attached packages:
```

```
[1] affy_1.16.0          preprocessCore_1.0.0 affyio_1.6.1
[4] Biobase_1.16.3

> system("uname -a", intern = TRUE)

[1] "Linux interaction 2.6.5-7.282-sn2 #1 SMP Tue Aug 29 10:40:40 UTC 2006 ia64 ia64 ia64
```

Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

July 1, 2008

1 Collect all comparable data into one place.

```
> library(affy)
> library(gdata)
```

The lookup files tell us which probes are comparable between microarray and RT-PCR:

```
> colon.lookup <- read.xls("SuppFile1.xls", sheet = 2, stringsAsFactors = FALSE)
> cells.lookup <- read.xls("SuppFile1.xls", sheet = 1, stringsAsFactors = FALSE)
```

Load the raw RT-PCR data (CT values):

```
> colon.rtpcr <- as.matrix(read.delim("Supplemental table 2_colon rtpcr raw data.txt",
+     row.names = 1))
> cells.rtpcr <- as.matrix(read.delim("Supplemental table 1_cell line rtpcr raw data.txt",
+     row.names = 1))
```

Remove the RT-PCR sample corresponding to the bad microarray sample.

```
> cells.rtpcr <- cells.rtpcr[, -20]
```

Normalize RT-PCR by 18s rRNA:

```
> colon.n18 <- scale(colon.rtpcr[-1, ], center = colon.rtpcr[1,
+     ], scale = FALSE)
> cells.n18 <- scale(cells.rtpcr[-1, ], center = cells.rtpcr[1,
+     ], scale = FALSE)
```

Normalize RT-PCR by mean (global normalization):

```
> colon.nm <- scale(colon.rtpcr[-1, ], center = TRUE, scale = FALSE)
> cells.nm <- scale(cells.rtpcr[-1, ], center = TRUE, scale = FALSE)
```

Reality check:

```
> load("colon.rma.RData")
> load("cells.rma.RData")
> all(colon.rma$geo_accession == colnames(colon.n18))
```

```
[1] TRUE

> all(cells.rma$geo_accession == colnames(cells.n18))

[1] TRUE
```

Combine RT-PCR expression values (two types of normalization) and microarray expression values (several types of normalization). Only the probes comparable across platforms are retained.

```
> cells <- list(pcr.n18 = cells.n18[cells.lookup$taqman.probe,
+     ], pcr.nm = cells.nm[cells.lookup$taqman.probe, ])
> colon <- list(pcr.n18 = colon.n18[colon.lookup$taqman.probe,
+     ], pcr.nm = colon.nm[colon.lookup$taqman.probe, ])
> algorithms <- unlist(list(RMA = "rma", MAS5 = "mas5", GCRMA = "gcrma",
+     MBEI = "mbei", PLIER = "plier", "PLIER+16" = "plierPlus16",
+     DFW = "dfw", FARMS = "qfarms", VSN = "vsn"))
> for (i in algorithms) {
+     objectName <- paste("cells.", i, sep = "")
+     filename <- paste(objectName, ".RData", sep = "")
+     load(filename)
+     cells[[i]] <- exprs(get(objectName))[cells.lookup$affymetrix.probe,
+         ]
+     rm(list = objectName)
+ }
> for (i in algorithms) {
+     objectName <- paste("colon.", i, sep = "")
+     filename <- paste(objectName, ".RData", sep = "")
+     load(filename)
+     colon[[i]] <- exprs(get(objectName))[colon.lookup$affymetrix.probe,
+         ]
+     rm(list = objectName)
+ }
```

We need to log2-transform the MAS5 and MBEI expression values. (The other algorithms return log2-transformed values).

```
> cells$mas5 <- log2(cells$mas5)
> colon$mas5 <- log2(colon$mas5)
> cells$mbei <- log2(cells$mbei)
> colon$mbei <- log2(colon$mbei)
> save(cells, file = "cells.RData")
> save(colon, file = "colon.RData")
```

2 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()

R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e

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

other attached packages:
[1] gdata_2.3.1          affy_1.16.0          preprocessCore_1.0.0
[4] affyio_1.6.1         Biobase_1.16.3

loaded via a namespace (and not attached):
[1] gtools_2.4.0

> system("uname -a", intern = TRUE)

[1] "Linux sbiology 2.6.5-7.282-sn2 #1 SMP Tue Aug 29 10:40:40 UTC 2006 ia64 ia64 ia64 GNU
```

Supplemental Material: Choice of normalization algorithm affects consistency between microarray and RT-PCR in clinical samples

Balazs Gyorffy Bela Molnar Hermann Lage Zoltan Szallasi
Aron C. Eklund

July 1, 2008

Here we generate figures 3 and 4.

```
> load("cells.RData")
> load("colon.RData")
```

Define the function for log-ratio discrepancy:

```
> disc <- function(x, y, na.rm = FALSE, fn = mean) {
+     stopifnot(length(x) == length(y))
+     ok <- is.finite(x) & is.finite(y)
+     stopifnot(all(ok) || na.rm)
+     x <- x[ok]
+     y <- y[ok]
+     x.d <- outer(x, x, "-")
+     y.d <- outer(y, y, "-")
+     d <- x.d[upper.tri(x.d)] - y.d[upper.tri(y.d)]
+     fn(abs(d))
+ }
```

Miscellaneous other functions used to do many comparisons between the two platforms:

```
> go.n18 <- function(x, fn, ...) {
+     sapply(x, function(y) {
+         sapply(1:nrow(y), function(i) {
+             fn(y[i, ], -x$pcr.n18[i, ], ...)
+         })
+     })
+ }
> go.nm <- function(x, fn, ...) {
+     sapply(x, function(y) {
+         sapply(1:nrow(y), function(i) {
+             fn(y[i, ], -x$pcr.nm[i, ], ...)
+         })
+     })
+ }
```

Calculate log-ratio discrepancy (LRD), Pearson correlation (PC) and Spearman correlation (SC) between each combination of the two RT-PCR normalization algorithms and the 10 microarray normalization algorithms.

(nm = "normalization to mean", n18 = "normalization to 18S rRNA")

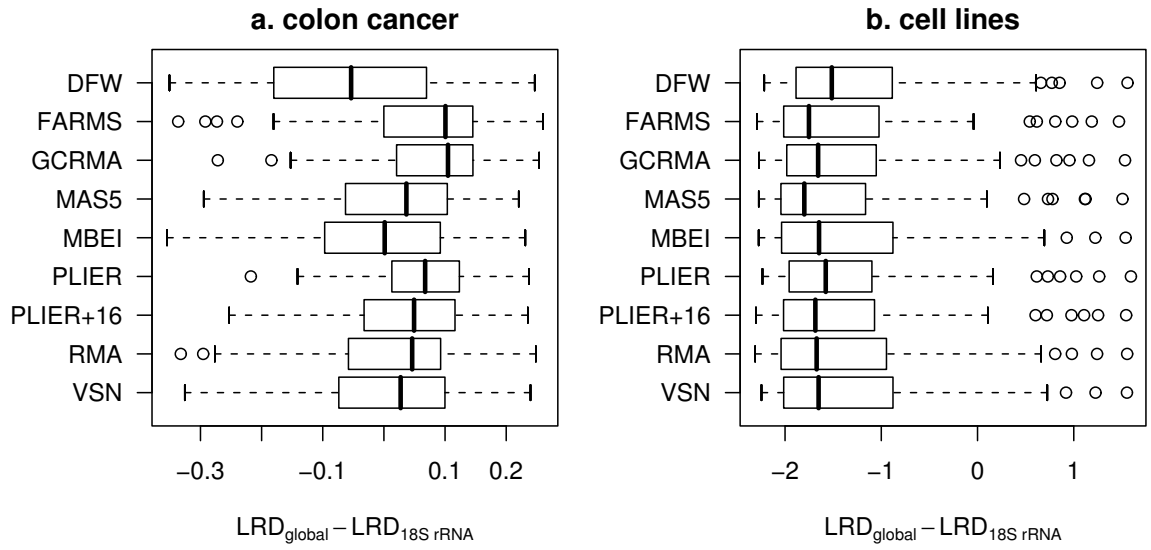
```
> colon.n18.LRD <- go.n18(colon, disc, na.rm = TRUE)
> colon.n18.PC <- go.n18(colon, cor, use = "pairwise")
> colon.n18.SC <- go.n18(colon, cor, method = "spearman", use = "complete")
> colon.nm.LRD <- go.nm(colon, disc, na.rm = TRUE)
> colon.nm.PC <- go.nm(colon, cor, use = "pairwise")
> colon.nm.SC <- go.nm(colon, cor, method = "spearman", use = "complete")
> cells.n18.LRD <- go.n18(cells, disc, na.rm = TRUE)
> cells.n18.PC <- go.n18(cells, cor, use = "pairwise")
> cells.n18.SC <- go.n18(cells, cor, method = "spearman", use = "complete")
> cells.nm.LRD <- go.nm(cells, disc, na.rm = TRUE)
> cells.nm.PC <- go.nm(cells, cor, use = "pairwise")
> cells.nm.SC <- go.nm(cells, cor, method = "spearman", use = "complete")
```

Misc. objects and functions for drawing figures:

```
> algorithms <- unlist(list(RMA = "rma", MAS5 = "mas5", GCRMA = "gcrma",
+     MBEI = "mbei", PLIER = "plier", "PLIER+16" = "plierPlus16",
+     DFW = "dfw", FARMS = "qfarms", VSN = "vsn"))
> nnn <- names(algorithms)
> names(nnn) <- algorithms
> pl3 <- function(x, ...) {
+     d <- as.data.frame(x[, c(-1, -2)])
+     names(d) <- nnn[names(d)]
+     d <- d[, rev(order(names(d)))]
+     boxplot(as.data.frame(d), horizontal = TRUE, las = 1, ...)
+ }
```

Now draw figure 3.

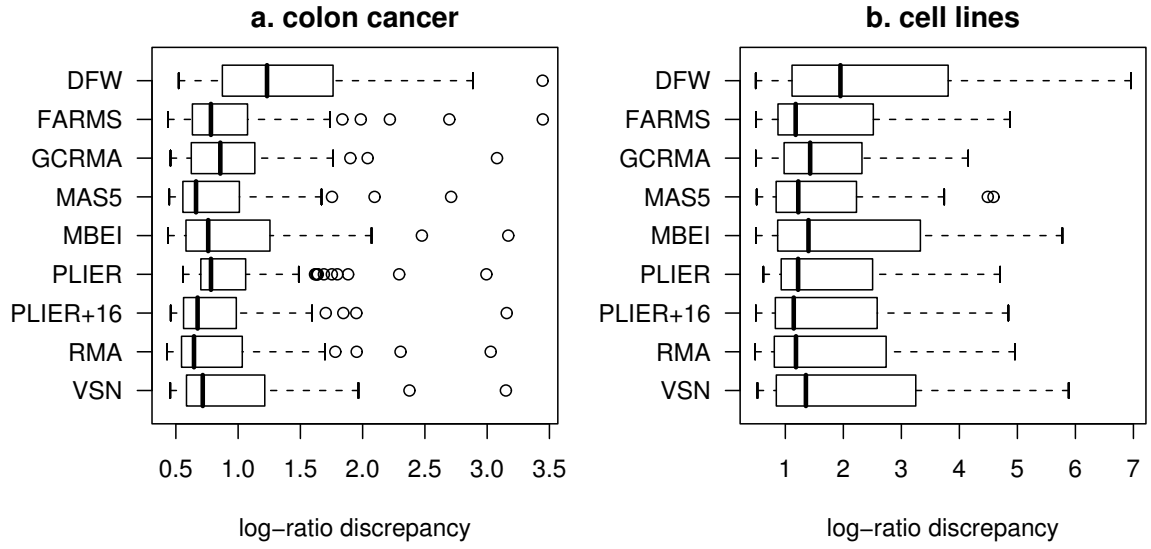
```
> par(mfrow = c(1, 2), mar = c(5, 5, 2, 1) + 0.1)
> pl3(colon.nm.LRD - colon.n18.LRD, xlab = expression(LRD[global] -
+     LRD["18S rRNA"]), main = "a. colon cancer")
> pl3(cells.nm.LRD - cells.n18.LRD, xlab = expression(LRD[global] -
+     LRD["18S rRNA"]), main = "b. cell lines")
```



Based on this, we will use 18S rRNA normalization for the colon biopsy data, and global-mean normalization normalization for the cell line data.

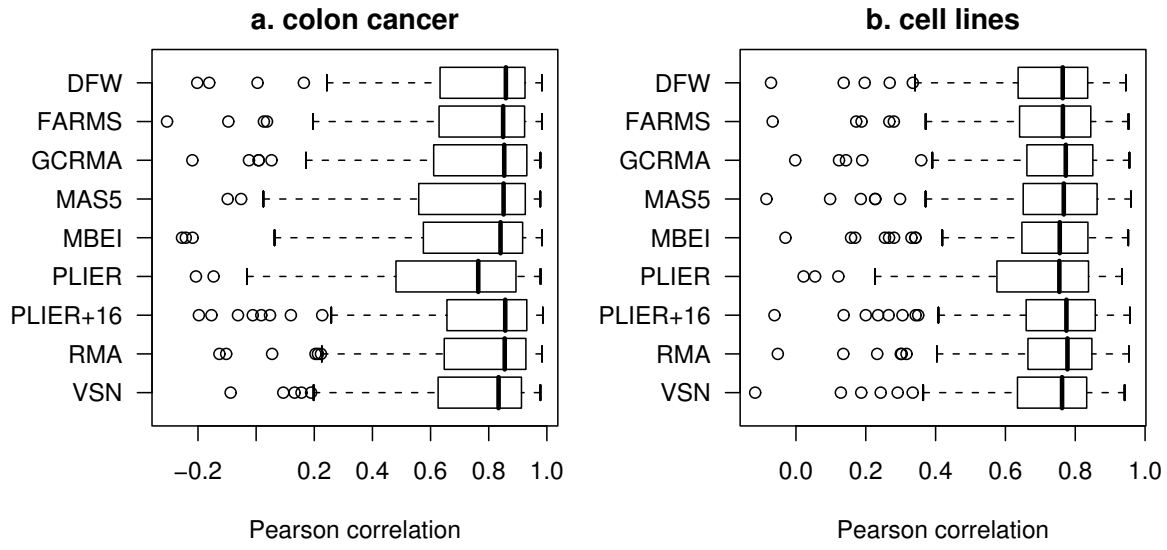
Now draw figure 4:

```
> par(mfrow = c(1, 2), mar = c(5, 5, 2, 1) + 0.1)
> pl3(colon.n18.LRD, xlab = "log-ratio discrepancy", main = "a. colon cancer")
> pl3(cells.nm.LRD, xlab = "log-ratio discrepancy", main = "b. cell lines")
```



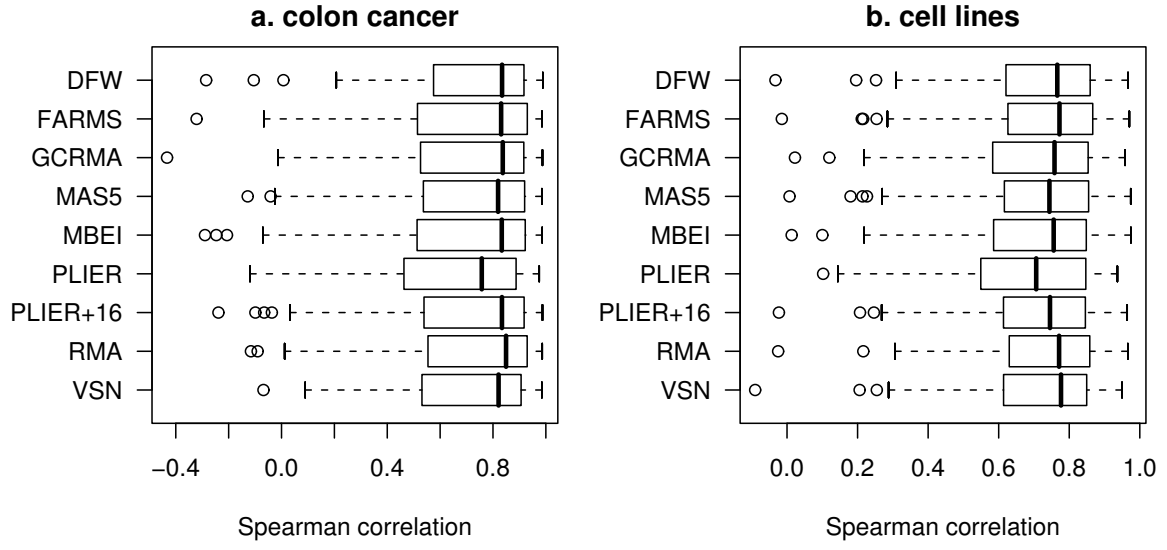
Here is the equivalent plot using Pearson correlation:

```
> par(mfrow = c(1, 2), mar = c(5, 5, 2, 1) + 0.1)
> pl3(colon.n18.PC, xlab = "Pearson correlation", main = "a. colon cancer")
> pl3(cells.nm.PC, xlab = "Pearson correlation", main = "b. cell lines")
```



Here is the equivalent plot using Spearman correlation:

```
> par(mfrow = c(1, 2), mar = c(5, 5, 2, 1) + 0.1)
> pl3(colon.n18.SC, xlab = "Spearman correlation", main = "a. colon cancer")
> pl3(cells.nm.SC, xlab = "Spearman correlation", main = "b. cell lines")
```



1 sessionInfo

The results in this file are generated using the following packages:

```
> sessionInfo()
```

```
R version 2.6.1 (2007-11-26)
ia64-unknown-linux-gnu

locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-8;LC_MONETARY=e

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

> system("uname -a", intern = TRUE)

[1] "Linux sbiology 2.6.5-7.282-sn2 #1 SMP Tue Aug 29 10:40:40 UTC 2006 ia64 ia64 ia64 GNU
```