1. Introduction

Transcriptomic deconvolution in cancer and other heterogeneous tissues remains challenging. Available methods lack the ability to estimate both component-specific proportions and expression profiles for individual samples. We develop a three-component deconvolution model, DeMixT, for expression data from a mixture of cancerous tissues, infiltrating immune cells and tumor microenvironment. DeMixT is a software package that performs deconvolution on transcriptome data from a mixture of two or three components.

DeMixT is a frequentist-based method and fast in yielding accurate estimates of cell proportions and compart-ment-specific expression profiles for two-component \and three-component deconvolution problem. Our method promises to provide deeper insight into cancer biomarkers and assist in the development of novel prognostic markers and therapeutic strategies.

The function DeMixT is designed to finish the whole pipeline of deconvolution for two or three components. DeMixT.S1 function is designed to estimate the proportions of all mixed samples for each mixing component. DeMixT.S2 function is designed to estimate the component-specific deconvolved expressions of individual mixed samples for a given set of genes.

2 Feature Description

The DeMixT R-package builds the transcriptomic deconvolution with a couple of novel features into R-based standard analysis pipeline through Bioconductor. DeMixT showed high accuracy and efficiency from our designed experiment. Hence, DeMixT can be considered as an important step towards linking tumor transcriptomic data with clinical outcomes.

Different from most previous computational deconvolution methods, DeMixT has integrated new features for the deconvolution with more than 2 components.

Joint estimation: jointly estimate component proportions and expression profiles for individual samples by requiring reference samples instead of reference genes; For the three-component deconvolution considering immune infiltration, it provides a comprehensive view of tumor-stroma-immune transcriptional dynamics, as compared to methods that address only immune subtypes within the immune component, in each tumor sample.

Efficient estimation: DeMixT adopts an approach of iterated conditional modes (ICM) to guarantee a rapid convergence to a local maximum. We also design a novel gene-set-based component merging approach to reduce the bias of proportion estimation for three-component deconvolutionthe.

parallel computing: OpenMP enables parallel computing on single computer by taking advantage of the multiple cores shipped on modern CPUs. The ICM framework further enables parallel computing, which helps compensate for the expensive computing time used in the repeated numerical double integrations.

3. Installation

3.1 Source file

DeMixT source files are compatible with Windows, Linux and macOS.

DeMixT_0.99.0 is the latest version, which is for a computer that has OpenMP. To install DeMixT_0.99.0, start R and enter:

```
# devtools::install_github("wwylab/DeMixT")
```

For more information, please visit: http://bioinformatics.mdanderson.org/main/DeMixT

3.2 Functions

The following table shows the functions included in DeMixT.

Table Header	Second Header
DeMixT	Deconvolution of tumor samples with two or three components
DeMixT_S1	Estimates the proportions of mixed samples for each mixing component
DeMixT_S2	Deconvolves expressions of each sample for unknown component
Optimum_KernelC	Call the C function used for parameter estimation in DeMixT

4. Methods

4.1 Model

Let Y_{ig} be the observed expression levels of the raw measured data from clinically derived malignant tumor samples for gene $g, g = 1, \cdots, G$ and sample $i, i = 1, \cdots, S$. G denotes the total number of probes/genes and S denotes the number of samples. The observed expression levels for solid tumors can be modeled as a linear combination of raw expression levels from three components:

$$Y_{ig} = \pi_{1,i} N_{1,ig} + \pi_{2,i} N_{2,ig} + (1 - \pi_{1,i} - \pi_{2,i}) T_{ig}$$

Here $N_{1,ig}$, $N_{2,ig}$ and T_{ig} are the unobserved raw expression levels from each of the three components. We call the two components for which we require reference samples the N_1-component and the N_2-component. We call the unknown component the T-component. We let $\pi_{1,i}$ denote the proportion of the N_1-component, $\pi_{2,i}$ denote the proportion of the N_2-component, and $1-\pi_{1,i}-\pi_{2,i}$ denote the proportion of the T-component. We assume that the mixing proportions of one specific sample remain the same across all genes.

Our model allows for one component to be unknown, and therefore does not require reference profiles from all components. A set of samples for $N_{1,ig}$ and $N_{2,ig}$, respectively, needs to be provided as input data. This three-component deconvolution model is applicable to the linear combination of any three components in any type of material. It can also be simplified to a two-component model, assuming there is just one N-component. For application in this paper, we consider tumor (T), stromal (N_1) and immune components (N_2) in an admixed sample (Y).

Following the convention that $\log_2$-transformed microarray gene expression data follow a normal distribution, we assume that the raw measures $N_{1,ig} \sim LN(\mu_{N_1g}, \sigma^2_{N_1g})$, $N_{2,ig} \sim LN(\mu_{N_2g}, \sigma^2_{N_2g})$ and $T_{ig} \sim LN(\mu_{Tg}, \sigma^2_{Tg})$, where LN denotes a $\log_2$-normal distribution and $\sigma^2_{N_1g}$,$\sigma^2_{N_2g}$, σ^2_{Tg} reflect the variations under $\log_2$-transformed data. Consequently, our model can be expressed as the convolution of the density function for three $\log_2$-normal distributions. Because there is no closed form of this convolution, we use numerical integration to evaluate the complete likelihood function (see the full likelihood in the Supplementary Materials).

4.2 The DeMixT algorithm for deconvolution

DeMixT estimates all distribution parameters and cellular proportions and reconstitutes the expression profiles for all three components for each gene and each sample. The estimation procedure (summarized in Figure 1b) has two main steps as follows.

1. Obtain a set of parameters $\{\pi_{1,i}, \pi_{2,i}\}_{i=1}^{S}$, $\{\mu_T, \sigma_T\}_{g=1}^{G}$ to maximize the complete likelihood function, for which $\{\mu_{N_{1,g}}, \sigma_{N_{1,g}}, \mu_{N_{2,g}}, \sigma_{N_{2,g}}\}_{g=1}^{G}$ were already estimated from the available unmatched samples of the N_1 and N_2 component tissues. (See further details in our paper.)

2. Reconstitute the expression profiles by searching each set of $\{n_{1,ig}, n_{2,ig}\}$ that maximizes the joint density of $N_{1,ig}$, $N_{2,ig}$ and T_{ig}. The value of t_{ig} is solved as $y_{ig} - \hat{\pi}_{1,i} n_{1,ig} - \hat{\pi}_{2,i} n_{2,ig}$.

These two steps can be separately implemented using the function DeMixT.S1 and DeMixT.S2, which are combined in the function DeMixT.

5. Examples

5.1 Simulated two-component data

```
library(DeMixT)
data(test.data1.y)
data(test.data1.comp1)
res <- DeMixT(data.Y = test.data1.y,
    data.comp1 = test.data1.comp1,
    if.filter = FALSE,
    output.more.info = TRUE)
```

```
res$pi
```

```
##             1         2         3         4        5       6         7
## pi1 0.1070489 0.2207103 0.2929484 0.3372995 0.426243 0.50345 0.6210632
##             8         9        10
## pi1 0.661861 0.7457062 0.7707635
```

```
head(res$ExprT, 3)
```

```
##          1         2         3        4        5        6        7
## 1 96.79585  83.41835  80.77222 80.60414 84.20071 83.03622 87.01594
## 2 77.09334 112.40671 136.32347 85.98078 81.23200 68.42844 87.29900
## 3 38.68587  35.18014  38.51009 37.91703 32.56513 74.14903 84.34022
##          8        9       10
## 1 86.22003 84.86463 85.45999
## 2 96.40083 92.02591 72.51438
## 3 46.25410 44.50538 56.71183
```

```
head(res$ExprN1, 3)
```

```
##           1         2         3         4         5         6         7
## 1 129.54542 116.09035 107.12345 104.37623 116.49396 107.61791 172.50324
## 2  90.46185  96.29083 102.25923  92.28136  89.61648  76.85337  96.72302
## 3  53.12755  52.61841  52.91753  52.74160  50.77739  58.11003  62.69202
##           8         9        10
## 1 156.53262 126.00456 152.83366
## 2 117.71219 120.03571  65.79434
## 3  56.37106  56.72734  68.31615
```

```
head(res$Mu, 3)
```

```
##       MuN1      MuT
## 1 6.940477 6.414444
## 2 6.582707 6.480817
## 3 5.767947 5.528961
```

```
head(res$Sigma, 3)
```

```
##     SigmaN1    SigmaT
## 1 0.2675802 0.1329832
## 2 0.3201497 0.3694953
## 3 0.2231435 0.5203534
```

```
res$pi.iter
```

```
## , , 1
##
##             [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
##  [1,] 0.09043013 0.2139893 0.2937994 0.3356124 0.4489045 0.5187335
##  [2,] 0.09201892 0.2256735 0.3003955 0.3433736 0.4443035 0.5215731
##  [3,] 0.09219042 0.2259280 0.3000349 0.3437348 0.4438505 0.5214386
##  [4,] 0.10617607 0.2253010 0.3015574 0.3608395 0.4454596 0.5067287
##  [5,] 0.10978012 0.2275883 0.3089822 0.3610623 0.4456584 0.5040961
##  [6,] 0.11123264 0.2424746 0.3081649 0.3580802 0.4229328 0.5039371
##  [7,] 0.11160111 0.2420850 0.3059384 0.3530211 0.4239054 0.4875370
##  [8,] 0.11057467 0.2216661 0.3054044 0.3515506 0.4250920 0.4873542
##  [9,] 0.10694393 0.2195627 0.3066021 0.3535477 0.4291094 0.4865487
## [10,] 0.10669682 0.2225809 0.2923172 0.3508721 0.4274588 0.4871565
##             [,7]      [,8]      [,9]     [,10]
##  [1,] 0.6544423 0.7281499 0.8230019 0.8744931
##  [2,] 0.6532637 0.7273787 0.8232729 0.8754061
##  [3,] 0.6528720 0.7269826 0.8231035 0.8336202
##  [4,] 0.6526201 0.7278005 0.7440686 0.8335697
##  [5,] 0.5813934 0.6121608 0.7436701 0.7410146
##  [6,] 0.5811786 0.6119601 0.7448591 0.7415347
##  [7,] 0.5805850 0.6121504 0.7455018 0.7418385
##  [8,] 0.5808233 0.6130303 0.7456994 0.7715662
##  [9,] 0.5962759 0.6133934 0.7461179 0.7943069
## [10,] 0.6197418 0.6905161 0.7460135 0.7512246
```

```
res$gene.name
```

```
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
##  [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
##  [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
##  [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
##  [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
##  [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
##  [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
##  [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
##  [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
## [100] "100" "101" "102" "103" "104" "105" "106" "107" "108" "109" "110"
## [111] "111" "112" "113" "114" "115" "116" "117" "118" "119" "120"
```

5.2 Simulated three-component data

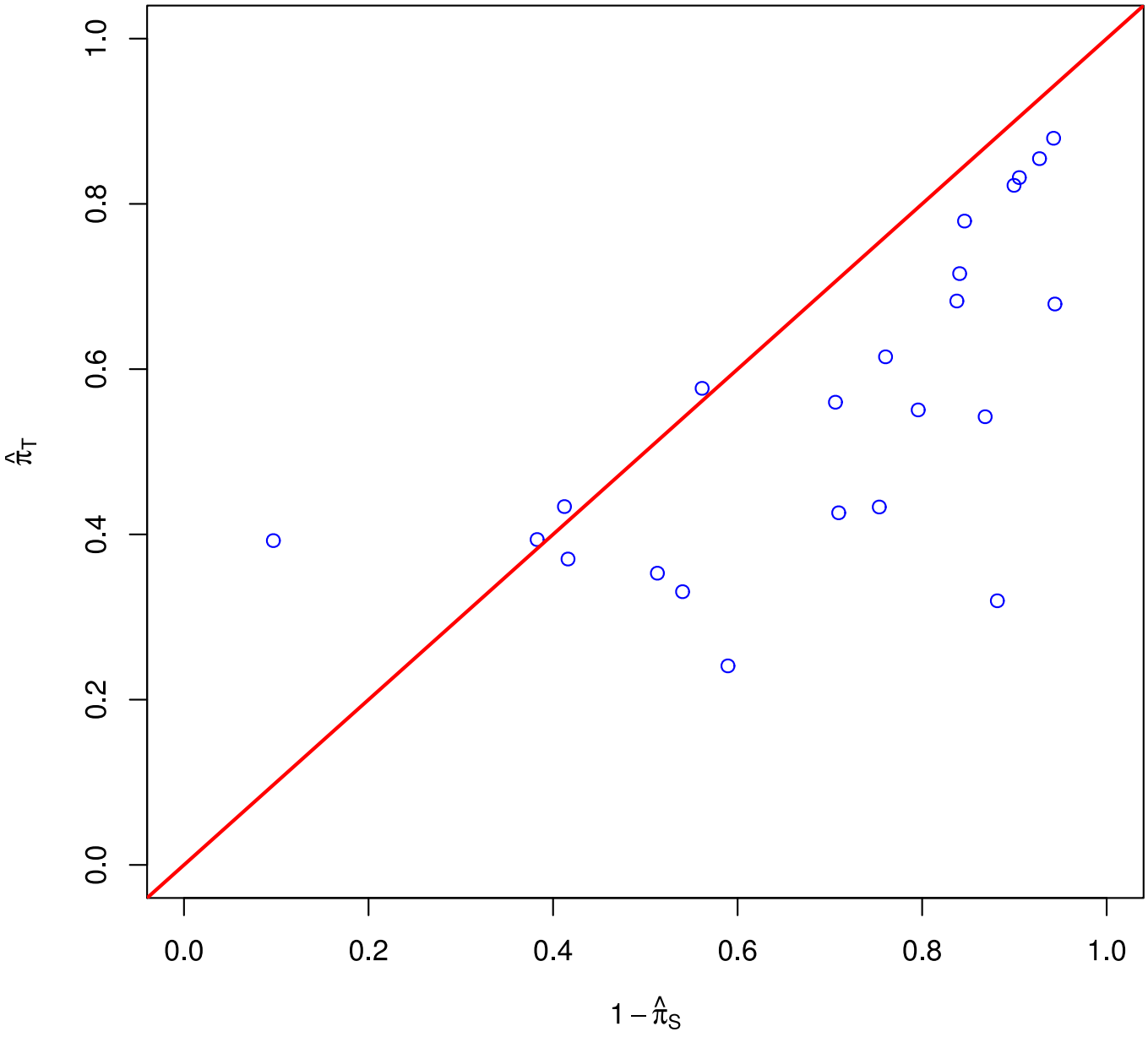
```
# data(test.data2.y)
# data(test.data2.comp1)
# data(test.data2.comp2)
# res <- DeMixT(data.Y = test.data2.y,
#     data.comp1 = test.data2.comp1,
#     data.comp2 = test.data2.comp2,
#     if.filter = FALSE)
```

5.3 Laser-capture microdissection prostate cancer FFPE microarray dataset

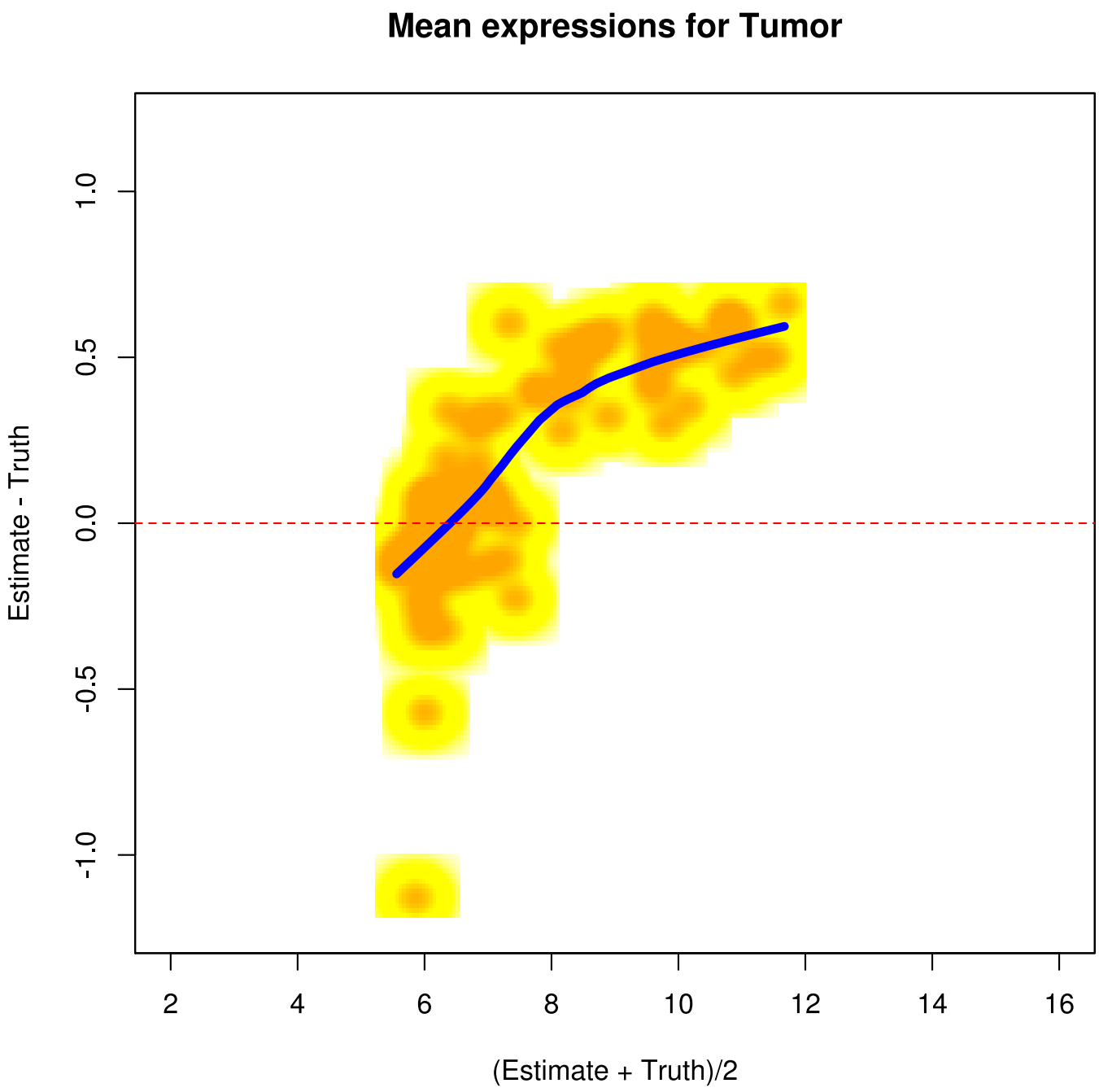
This dataset was generated at the Dana Farber Cancer Institute (GSE97284). Radical prostatectomy specimens were annotated in detail by pathologists, and regions of interest were identified that corresponded to benign epithelium, prostatic intraepithelial neoplasia (abnormal tissue that is possibly precancerous), and tumor, each with its surrounding stroma. FFPE samples are known to generate overall lower quality expression data than those from fresh frozen samples. We observed a small proportion of probesets that presented large differences in mean expression levels between the dissected tissues: tumor (T) and stroma (N) in this dataset. Only 53 probesets presented a mean difference $(|\bar{T} - \bar{N}|) > 1$, as compared to 10,397 probesets in GSE19830. We therefore chose the top 80 genes with the largest mean differences and ran DeMixT under two settings: tumor unknown and stroma unknown. DeMixT is able to obtain concordant estimates of the tumor proportions when the proportion of the stromal component was unknown and when the proportion of tumor tissue was unknown and also tended to provide accurate component-specific mean expression levels.

```
library(DeMixT)
data <- as.matrix(read.table("input.lcm.txt", header = FALSE))
normal <- data[, 1:25]
adm <- data[, 26:48]
tumor <- data[, 49:73]

nrows <- nrow(adm); ncols <- ncol(adm)
lcm.data.adm <- matrix(2^adm, nrows)
lcm.data.adm <- SummarizedExperiment(assays=list(counts=lcm.data.adm))

nrows <- nrow(tumor); ncols <- ncol(tumor)
lcm.data.tumor <- matrix(2^tumor, nrows)
lcm.data.tumor <- SummarizedExperiment(assays=list(counts=lcm.data.tumor))

nrows <- nrow(normal); ncols <- ncol(normal)
lcm.data.normal <- matrix(2^normal, nrows)
lcm.data.normal <- SummarizedExperiment(assays=list(counts=lcm.data.normal))

testr.TA <- DeMixT(data.Y = lcm.data.adm, data.comp1 = lcm.data.tumor,
    niter = 20, nbin = 60, if.filter = FALSE, tol = 10^-6)
testr.SA <- DeMixT(data.Y = lcm.data.adm, data.comp1 = lcm.data.normal,
    niter = 20, nbin = 60, if.filter = FALSE, tol = 10^-6)
```

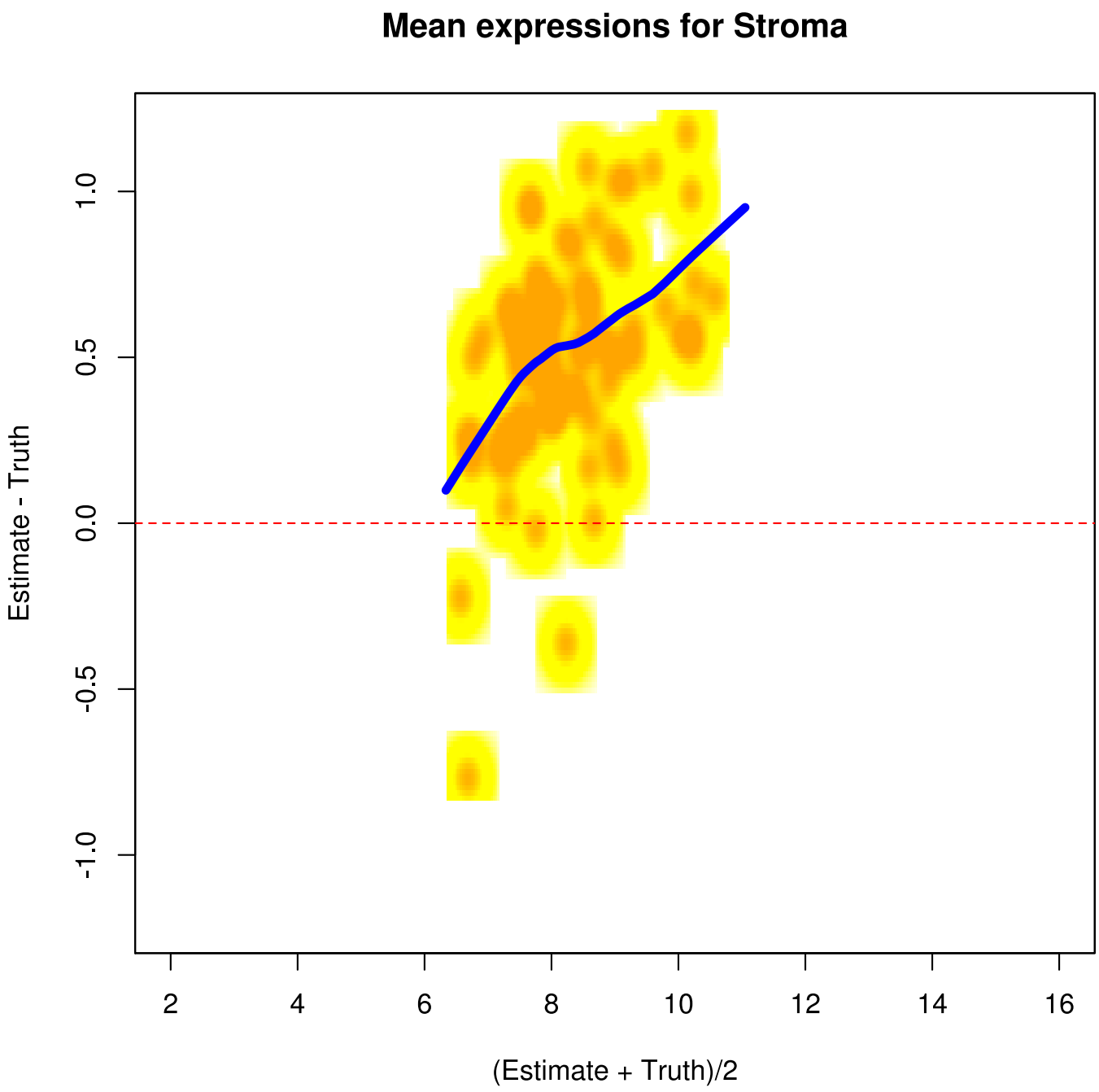
```
# plot A
dt_purT <- 1- as.numeric(testr.SA$pi)
dt_purS <- 1- as.numeric(testr.TA$pi)
plot(1 - dt_purS, dt_purT,
    col = "blue", pch = 1, xlim = c(0, 1), ylim = c(0, 1),
    xlab = expression(1 - hat(pi)[S]), ylab = expression(hat(pi)[T]))
abline(0, 1, col = "red", lwd = 2)
```


1.0
0.8
0.6
0.4
0.2
0.0
$\hat{\pi}_T$
0.0
0.2
0.4
0.6
0.8
1.0
$1 - \hat{\pi}_S$

```
# Plot - Mean expressions for Tumor
OB_St <- log2(read.table("lcm_normal.txt", header = FALSE))
OB_Tu <- log2(read.table("lcm_tumor.txt", header = FALSE))
DT_Tu_mu <- as.numeric(testr.SA$Mu[, 1])
DT_St_mu <- as.numeric(testr.TA$Mu[, 1])
DT_Tu_sg <- as.numeric(testr.SA$Sigma[, 1])
DT_St_sg <- as.numeric(testr.TA$Sigma[, 1])
OB_St_m <- apply(OB_St, 1, mean)
OB_Tu_m <- apply(OB_Tu, 1, mean)

# filter out genes with large estimated standard deviations
condSt <- (DT_St_sg < 0.99)
condTu <- (DT_Tu_sg < 0.99)
DT_Tu_m <- as.numeric(apply(log2(testr.SA$ExprT), 1, mean))
DT_St_m <- as.numeric(apply(log2(testr.TA$ExprT), 1, mean))
OB_St_m <- OB_St_m[condSt]
OB_Tu_m <- OB_Tu_m[condTu]
DT_St_m <- DT_St_m[condSt]
DT_Tu_m <- DT_Tu_m[condTu]

# Plot - Mean expressions for Tumor
smoothScatter((DT_Tu_m + OB_Tu_m) / 2, DT_Tu_m - OB_Tu_m,
    ylab = "Estimate - Truth", xlab = "(Estimate + Truth)/2",
    xlim = c(2,16), ylim = c(-1.2,1.2),
    main = "Mean expressions for Tumor",
    pch = 1, nrpoints = 0, col = 'yellow',
    colramp=colorRampPalette(c("white","yellow",
    "yellow1","orange","orange1")))

tmp01 <- lowess((DT_Tu_m - OB_Tu_m) ~ ((DT_Tu_m + OB_Tu_m) / 2))
lines(tmp01$x, tmp01$y, col="blue", lwd = 5)
abline(h = 0, col = 'red', lty = 2)
```


Mean expressions for Tumor
Estimate - Truth
1.0
0.5
0.0
-0.5
-1.0
2
4
6
8
10
12
14
16
(Estimate + Truth)/2

```
# Plot - Mean expressions for Stroma
smoothScatter((DT_St_m + OB_St_m) / 2, DT_St_m - OB_St_m,
    ylab = "Estimate - Truth", xlab = "(Estimate + Truth)/2",
    xlim = c(2,16), ylim = c(-1.2,1.2),
    main = "Mean expressions for Stroma", pch = 1, nrpoints = 0,
    col = 'yellow',
    colramp=colorRampPalette(c("white","yellow",
    "yellow1","orange","orange1")))

tmp01 <- lowess((DT_St_m - OB_St_m) ~ ((DT_St_m + OB_St_m) / 2))
lines(tmp01$x, tmp01$y, col="blue", lwd = 5)
abline(h = 0, col = 'red', lty = 2)
```



6. Session Info

```
sessionInfo(package = "DeMixT")
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Red Hat Enterprise Linux
##
## Matrix products: default
## BLAS:   /software/x86_64/R/3.6.0/lib64/R/lib/libRblas.so
## LAPACK: /software/x86_64/R/3.6.0/lib64/R/lib/libRlapack.so
##
## 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
##
## attached base packages:
## character(0)
##
## other attached packages:
## [1] DeMixT_0.99.10
##
## loaded via a namespace (and not attached):
##  [1] knitr_1.23                  XVector_0.22.0
##  [3] magrittr_1.5                GenomicRanges_1.34.0
##  [5] BiocGenerics_0.28.0         zlibbioc_1.28.0
##  [7] IRanges_2.16.0              grDevices_3.6.0
##  [9] BiocParallel_1.16.6         lattice_0.20-38
## [11] highr_0.8                   stringr_1.4.0
## [13] GenomeInfoDb_1.18.2         tools_3.6.0
## [15] utils_3.6.0                 SummarizedExperiment_1.12.0
## [17] parallel_3.6.0              grid_3.6.0
## [19] Biobase_2.42.0              xfun_0.7
## [21] KernSmooth_2.23-15          stats_3.6.0
## [23] datasets_3.6.0              matrixStats_0.54.0
## [25] base_3.6.0                  Matrix_1.2-17
## [27] GenomeInfoDbData_1.2.0      graphics_3.6.0
## [29] S4Vectors_0.20.1            bitops_1.0-6
## [31] RCurl_1.95-4.12             evaluate_0.13
## [33] DelayedArray_0.8.0          stringi_1.4.3
## [35] compiler_3.6.0              methods_3.6.0
## [37] stats4_3.6.0
```