Supplementary Material

Melanoma antigen family A identified by the bimodality index defines a subset of triple negative breast cancers as candidates for immune response augmentation

Thomas Karn[1], Lajos Pusztai[2], Eugen Ruckhäberle[1], Cornelia Liedtke[3], Volkmar Müller[4], Marcus Schmidt[5], Dirk Metzler[6], Jing Wang[7], Kevin R. Coombes[7], Regine Gätje[1], Lars Hanker[1], Christine Solbach[1], Andre Ahr[1], Uwe Holtrich[1], Achim Rody[1], Manfred Kaufmann[1]

[1] Department of Obstetrics and Gynecology, J. W. Goethe-University, Frankfurt, Germany
[2] Department of Breast Medical Oncology, University of Texas M.D. Anderson Cancer Center, Houston, Texas, USA
[3] Department of Obstetrics and Gynecology, University of Muenster, Muenster, Germany
[4] Department of Obstetrics and Gynecology, University Hospital Hamburg-Eppendorf, Hamburg, Germany
[5] Department of Obstetrics and Gynecology, Gutenberg-University, Mainz, Germany
[6] Department of Biology II, Ludwig-Maximilians-University, Munich, Germany
[7] Department of Bioinformatics and Computational Biology, The University of Texas MD Anderson Cancer Center, Houston, Texas, USA

Corresponding Author:
Thomas Karn, PhD
Department of Obstetrics and Gynecology
J. W. Goethe-University
Theodor-Stern-Kai 7
D-60590 Frankfurt
Germany
Phone: +49-69-6301-4120
Fax: +49-69-6301-7025
e-mail: t.karn@em.uni-frankfurt.de

Stability analysis through different thresholds for correlation to metagenes and dataset bias

To analyze the stability of the methods and for sensitivity analysis we also used alternative cutoffs in control experiments. Therefore we increased the correlation coefficient threshold to $\geq$ 0.3 thereby increasing the number of "unclassified" probe sets. Using this threshold 91 (68.4 %) of 133 bimodal probe sets would be considered uncorrelated or "unclassified" (Supplementary Figure S6A and Supplementary Table S2).

In addition a more relaxed Kruskal-Wallis filter for dataset bias of 140 was derived from the distribution of the Kruskal-Wallis statistic as shown in Supplementary Figure S2. This filter excluded only 20 of the 222 bimodally expressed probesets with strongest bias (see Supplementary Table S2) and was also used in control experiments. We performed similar analyses using the 202 bimodally expressed probe sets that were identified using the more relaxed Kruskal-Wallis filter for dataset bias. This analysis resulted in 92 and 141 "unclassified" bimodal probe sets corresponding for correlation thresholds 0.2 and 0.3, respectively (Supplementary Figures S6B and S6C, and Supplementary Table S2).

Definition of cutoff values for dichotomizing of patient cohorts

In order to plot Kaplan-Meier curves, patients were dichotomized into low or high expression groups. Thresholds for CT-X metagenes were derived from their bimodal distribution as shown in Supplementary Figure S4. A cutoff for the continous distribution of the B-Cell metagene was optimized stepwise (0.001) as shown in Supplementary Figure S8. The cutoff of 0.005 was selected which displayed (i) high significance in univariate Cox regression concurrently with (ii) mostly equally sized sample groups.

Expression of CT-X metagenes among different molecular subtypes of breast cancer

The distribution of the expression values of the MAGE-A, MAGE-C3, and CTAG metagenes in different molecular subtypes was analyzed in the complete dataset of 3488 primary breast cancer samples as shown in Supplementary Figure S7. Samples were either stratified by ER status of the tumor or by molecular subtype according to the method of Hugh et al.[A].

In addition we used the centroid method based on the intrinsic gene set of Hu et al.[B]. For the centroid subtype definition we applied a recently published implementation of different variants of this method to assign breast cancer samples to a molecular subtype[C]. Detailed information and corresponding R-code can be downloaded from the authors of this study at:

http://rock.icr.ac.uk/collaborations/Mackay/centroid.correlations.Eset/ExpressionSet%20Nearest%20Centroid%20Correlations.pdf. For the results presented in Supplementary Figure S7 we performed spearman rank correlations on all probes with centering using the centroids according to Hu et al.[B] downloaded from

http://rock.icr.ac.uk/collaborations/Mackay/centroid.correlations.Eset/Hu306.centroids.txt. The analyses were performed independently in seven larger datasets (Frankfurt, Mainz, NewYork, Stockholm, Transbig, Uppsala, Rotterdam) to assign a total of 1364 breast cancer samples to a molecular subtype.

Supplementary Figures S7A and S7C, respectively, demonstrate that high expression of MAGE-A and CTAG metagenes is mainly confined to the subtypes of triple negative or basal-like breast cancer. Similar results were observed in a recent study by Grigoriadis et al.[D] when performing immunohistochemical analysis of tissue microarrays (TMA) and antibodies directed againgst CTAG1B and the MAGE-A

Supplementary References:

[A] Hugh J, Hanson J, Cheang MC, Nielsen TO, Perou CM, Dumontet C, Reed J, Krajewska M, Treilleux I, Rupin M, Magherini E, Mackey J, Martin M, Vogel C. Breast cancer subtypes and response to docetaxel in node-positive breast cancer: use of an immunohistochemical definition in the BCIRG 001 trial. J Clin Oncol. 2009 Mar 10;27(8):1168-76.

[B] Hu Z, Fan C, Oh DS, Marron JS, He X, Qaqish BF, Livasy C, Carey LA, Reynolds E, Dressler L, Nobel A, Parker J, Ewend MG, Sawyer LR, Wu J, Liu Y, Nanda R, Tretiakova M, Ruiz Orrico A, Dreher D, Palazzo JP, Perreard L, Nelson E, Mone M, Hansen H, Mullins M, Quackenbush JF, Ellis MJ, Olopade OI, Bernard PS, Perou CM. The molecular portraits of breast tumors are conserved across microarray platforms. BMC Genomics. 2006 Apr 27;7:96.

[C] Weigelt B, Mackay A, A'hern R, Natrajan R, Tan DS, Dowsett M, Ashworth A, Reis-Filho JS. Breast cancer molecular profiling with single sample predictors: a retrospective analysis. Lancet Oncol. 2010 Apr;11(4):339-49.

[D] Grigoriadis A, Caballero OL, Hoek KS, da Silva L, Chen YT, Shin SJ, Jungbluth AA, Miller LD, Clouston D, Cebon J, Old LJ, Lakhani SR, Simpson AJ, Neville AM. CT-X antigen expression in human breast cancer. Proc Natl Acad Sci U S A. 2009 Aug 11;106(32):13493-8.

The Bimodality Index Identifies a Subset of Triple Negative Breast Cancers with High Expression of CT-X Antigens and a Worse Prognosis

Thomas Karn *et al*

November 2, 2010

Contents

List of Figures

1 R library Used

```
> library(oompaBase)
> library(ClassDiscovery)
```

2 Select the n=394 comparable TNBC samples from n=579 TNBC Affymetrix MAS5 data

Variables used

cd_compl_579 == chip-data complete (data.frame, 579 samples in columns, 22283 probesets in rows)
header_579 == sample infos including dataset allocation (data.frame, 579 samples in columns)
n_probes == number of ProbeSets (rows of cd_compl_579)
tcdm_579 == transposed chip data matrix
t_header_579 == transposed sample infos (data.frame)
ds_579 == dataset allocation of 579 samples from header_579
tds_579 == transposed ds_579 (dataset allocation as matrix)
tdsn_579 == as.numeric(tds_579) (vector)

Load data

```
> fPath <- "//data/bioinfo2/Lung-HN/Thomas-JingWang/"
> cd_compl_579 <- read.delim(file.path(fPath, "n579_TNBC_DATA_Sort_Datas_Combi_id.txt"),
+     row.names = "SampleNames", h = T)
> dim(cd_compl_579)

[1] 22283   579

> header_579 <- read.delim("n579_TNBC_Sample_Info_Sort_Datas_Combi_id.txt",
+     row.names = "SampleNames", h = T)

> n_col_cd = ncol(cd_compl_579)
> n_probes = nrow(cd_compl_579)

> ds_579 = header_579[2, ]
> tds_579 = t(ds_579)
> tdsn_579 = as.numeric(tds_579)

> tcdm_579 = t(cd_compl_579)
> t_header_579 = as.data.frame(t(header_579))
> ds_mean = by(tcdm_579, tdsn_579, mean)

> tcdm_mean = apply(tcdm_579, 2, mean)
> tcdm_stdev = apply(tcdm_579, 2, sd)
```

3 Calculate comparability metrics for the datasets

calculate sum of squared differences of dataset-mean from total-mean for all probesets
Define variables

```
> n_datas = length(ds_mean)
> diff_to_mean = matrix(0, nrow = n_probes, ncol = n_datas)
> nrm_diff_to_mean = matrix(0, nrow = n_probes, ncol = n_datas)

> for (probes in 1:n_probes) {
+     for (i in 1:n_datas) {
+         diff_to_mean[probes, i] = ds_mean[[i]][probes] - tcdm_mean[probes]
+         nrm_diff_to_mean[probes, i] = (ds_mean[[i]][probes] - tcdm_mean[probes])/tcdm_stdev[p
+     }
+ }
```

calculate squares of differences

```
> squ_diff = diff_to_mean^2
> squ_nrm_diff = nrm_diff_to_mean^2
```

sum of squared differences by column

```
> sum_squ_diff = apply(na.omit(squ_diff), 2, sum)
> sum_squ_nrm_diff = apply(na.omit(squ_nrm_diff), 2, sum)
```

summarize results

```
> comparab = data.frame(unique(tds_579), sum_squ_diff, sum_squ_nrm_diff)
> names(comparab) = c("dataset", "sum_squ_diff", "sum_squ_nrm_diff")
> sort.comparab = comparab[order(comparab$sum_squ_nrm_diff), ]
```

integrate normalized comparab data in sample info in t_header_579

```
> for (i in 1:n_col_cd) {
+     t_header_579$comparab_nrm[i] = comparab$sum_squ_nrm_diff[comparab$dataset ==
+         tdsn_579[i]]
+ }
```

remove temporary variables:

```
> rm(diff_to_mean, nrm_diff_to_mean, ds_mean, tcdm_mean, tcdm_stdev,
+     squ_diff, squ_nrm_diff, sum_squ_diff, sum_squ_nrm_diff)
```

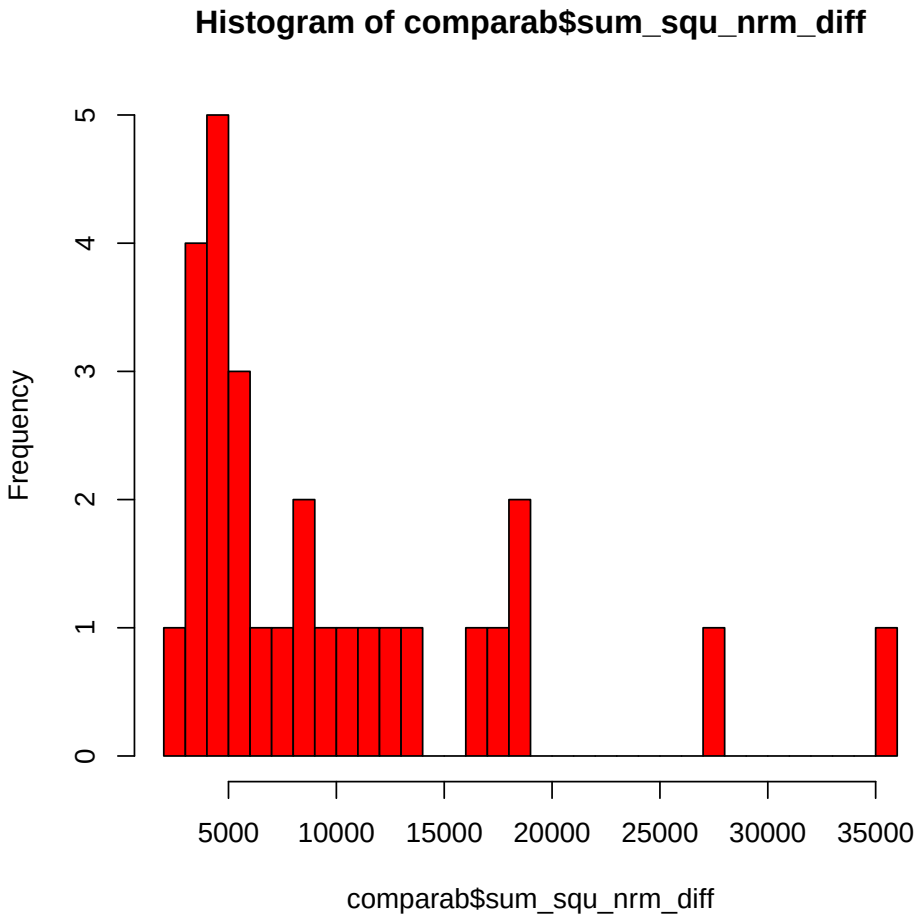

Figure 1: Histogram-sum-squ-nrm-diff

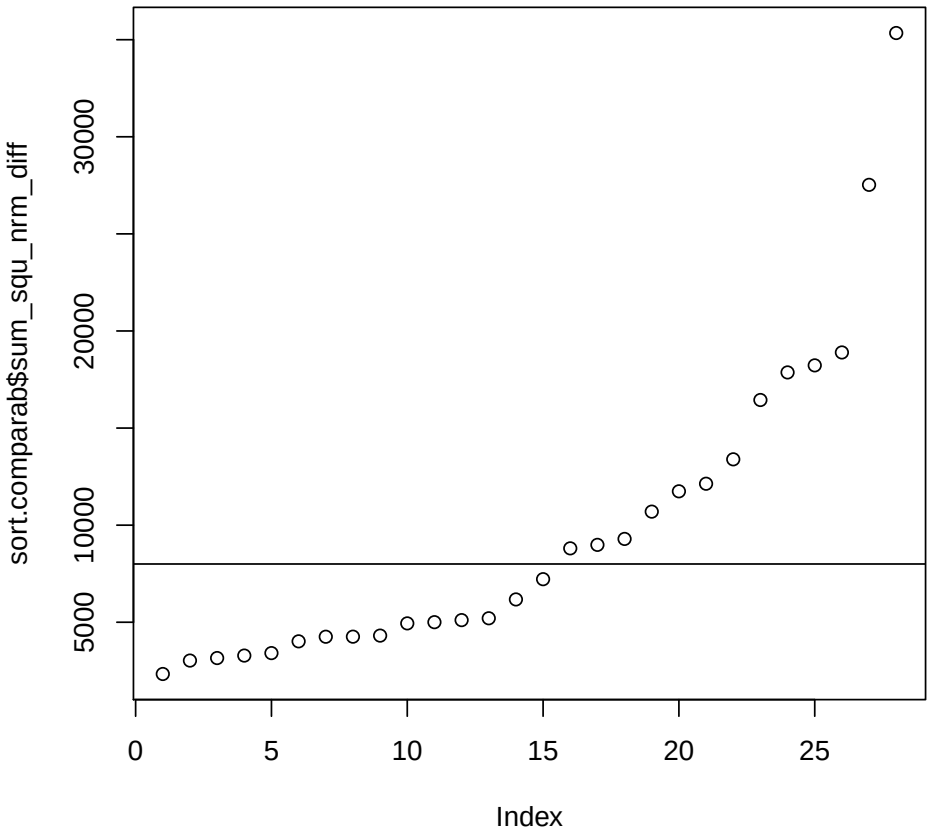

Figure 2: Plot-sum-squ-nrm-diff

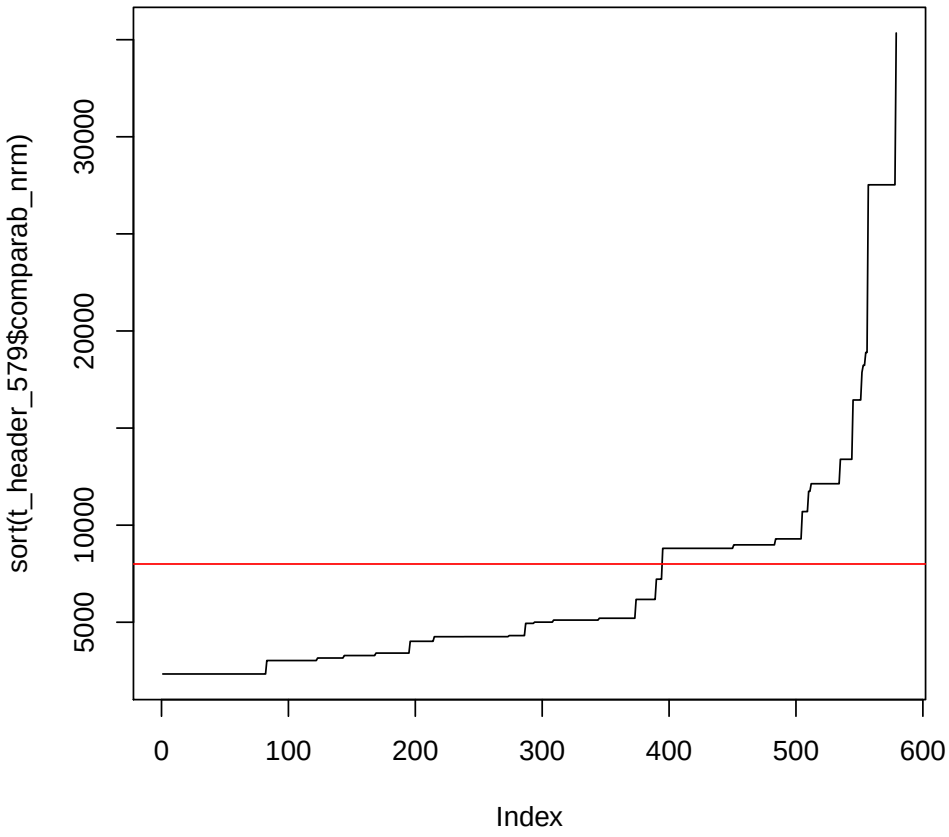

Figure 3: Plot-comparab-nrm

4 Select a subset of datasets with lowest comparability metric

Select a subset of comparab by defining criteria:

```
> compar_subset = subset(comparab, subset = sum_squ_nrm_diff < 8000)
```

vector of corresponding datasets:
Generate logical vector FALSE/TRUE for the complete dataset of 579 TNBC:

```
> datas_subset = compar_subset$dataset
> subset_index_vector = (tdsn_579 %in% datas_subset)
```

Query selected samples from transposed chipdata matrix:

```
> tcdm_394 = tcdm_579[subset_index_vector, ]
```

Query corresponding transposed dataset vector:

```
> tdsn_394 = tdsn_579[subset_index_vector]
```

Query selcted samples from NOT-transposed dataset (and corresponding header and t_header):

```
> cd_compl_394 = cd_compl_579[, subset_index_vector]
> header_394 = header_579[, subset_index_vector]
> t_header_394 = t_header_579[subset_index_vector, ]
```

5 Run the Bimodality-Index on the dataset of 394 TNBC with comparable arrays

```
> bi <- bimodalIndex(cd_compl_394)
```

```
1 ..........
2 ..........
3 ..........
4 ..........
5 ..........
6 ..........
7 ..........
8 ..........
9 ..........
10 ..........
11 ..........
12 ..........
13 ..........
14 ..........
```

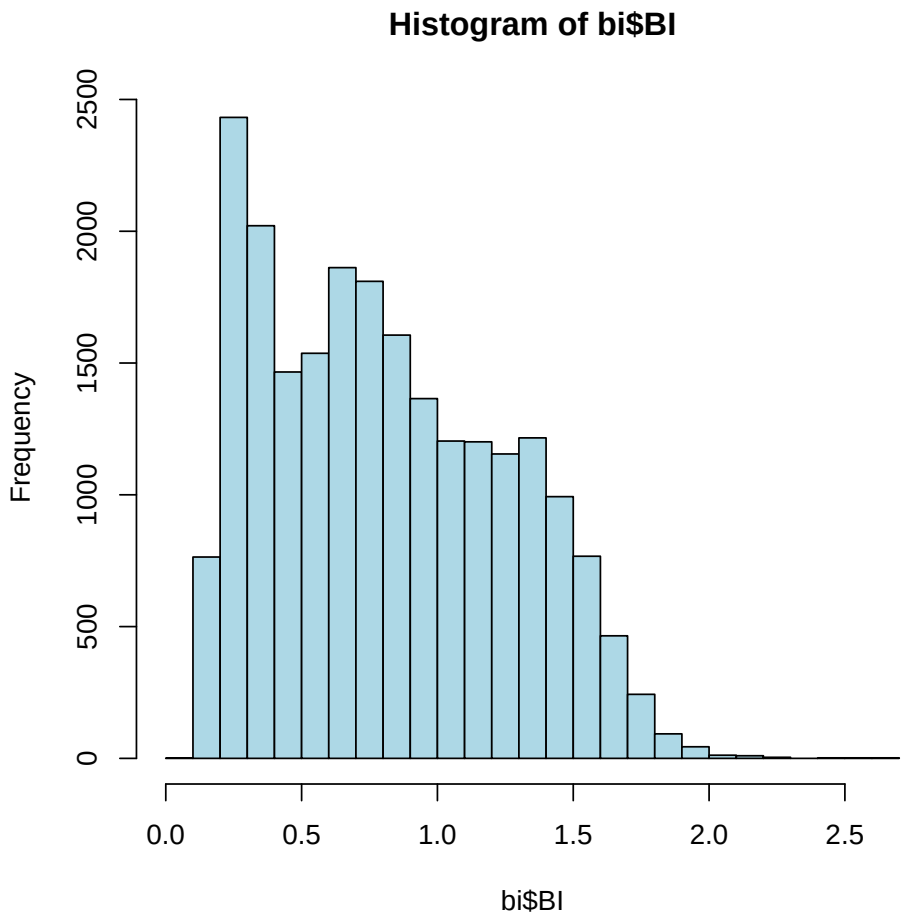

Figure 4: Histogram-BI

```
15 ..........
16 ..........
17 ..........
18 ..........
19 ..........
20 ..........
21 ..........
22 ..........
23 ..
```

Select 1 percent probes with highest BI

```
> sort.bi = bi[order(bi$BI, na.last = TRUE, decreasing = TRUE), ]
> bi_selection = sort.bi[1:222, ]
> bi_sel_probes = rownames(bi_selection)
> bi_sel_probes_data = cd_compl_394[rownames(cd_compl_394) %in% bi_sel_probes,
+     ]
```

6 Calculate Kruskal-Wallis statistics for all probesets according to their association with the dataset vector among the 394 TNBC samples

```
> kruskal_result = kruskal.test(tcdm_394[, 1], tdsn_394)
> kruskal_stat = matrix(0, nrow = n_probes)
> kruskal_p = matrix(0, nrow = n_probes)
```

loop perfoming kruskal-test for each probeset (chip-data vs. dataset-vector) the results of each test are variables of type LIST, which are combined by indexing with [[i]]

```
> for (i in 1:n_probes) {
+     kruskal_result[[i]] = kruskal.test(tcdm_394[, i], tdsn_394)
+     kruskal_stat[i] = kruskal_result[[i]]$statistic
+     kruskal_p[i] = kruskal_result[[i]]$p.value
+ }
```

summarize statistics and p-values and probeset names in one dataframe:

```
> kruskal_param = data.frame(row.names = rownames(cd_compl_394), kruskal_stat,
+     kruskal_p)
```

Remove temporary variables:

```
> rm(kruskal_result, kruskal_stat, kruskal_p)


> bi_sel_probes_kruskal = kruskal_param[rownames(kruskal_param) %in%
+     bi_sel_probes, ]
```

cutoffs 75 and 140 adapted from this histogram

```
> bi_sel_probes_kru75 = subset(bi_sel_probes_kruskal, subset = bi_sel_probes_kruskal$kruskal_st
+     75)
> bi_sel_probes_kru140 = subset(bi_sel_probes_kruskal, subset = bi_sel_probes_kruskal$kruskal_s
+     140)
> bi_sel_probes_kru75_data = cd_compl_394[rownames(cd_compl_394) %in%
+     rownames(bi_sel_probes_kru75), ]
> bi_sel_probes_kru140_data = cd_compl_394[rownames(cd_compl_394) %in%
+     rownames(bi_sel_probes_kru140), ]
```

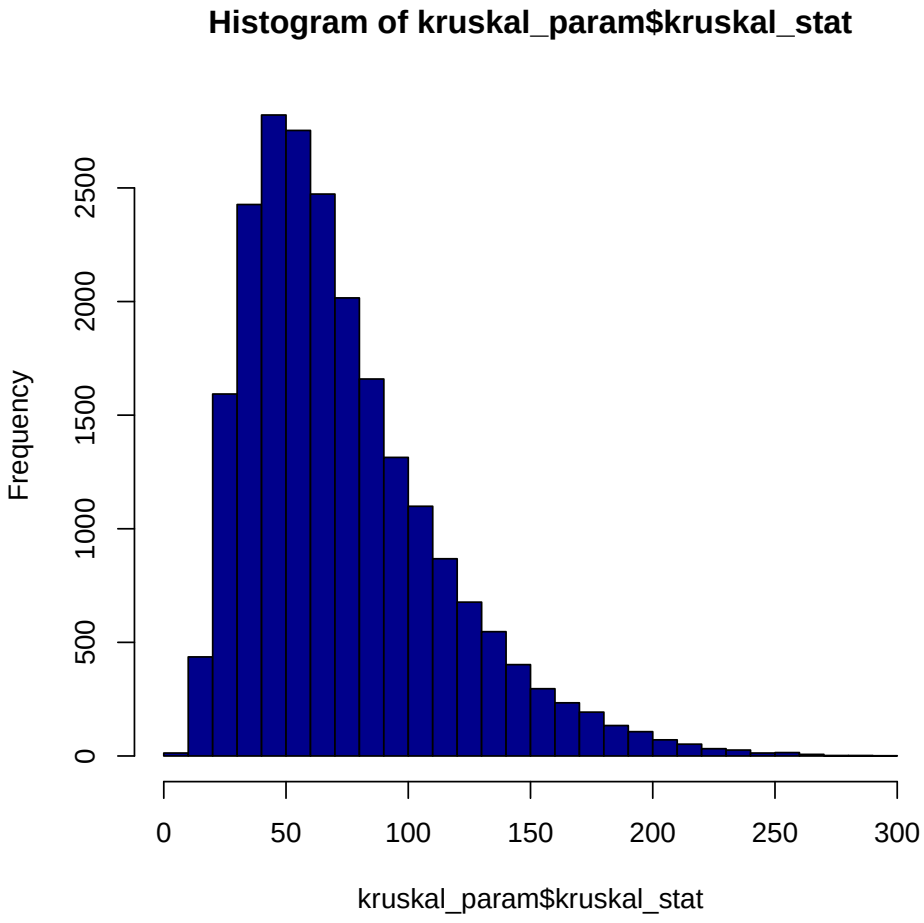

Figure 5: Histogram-kruskal-stat

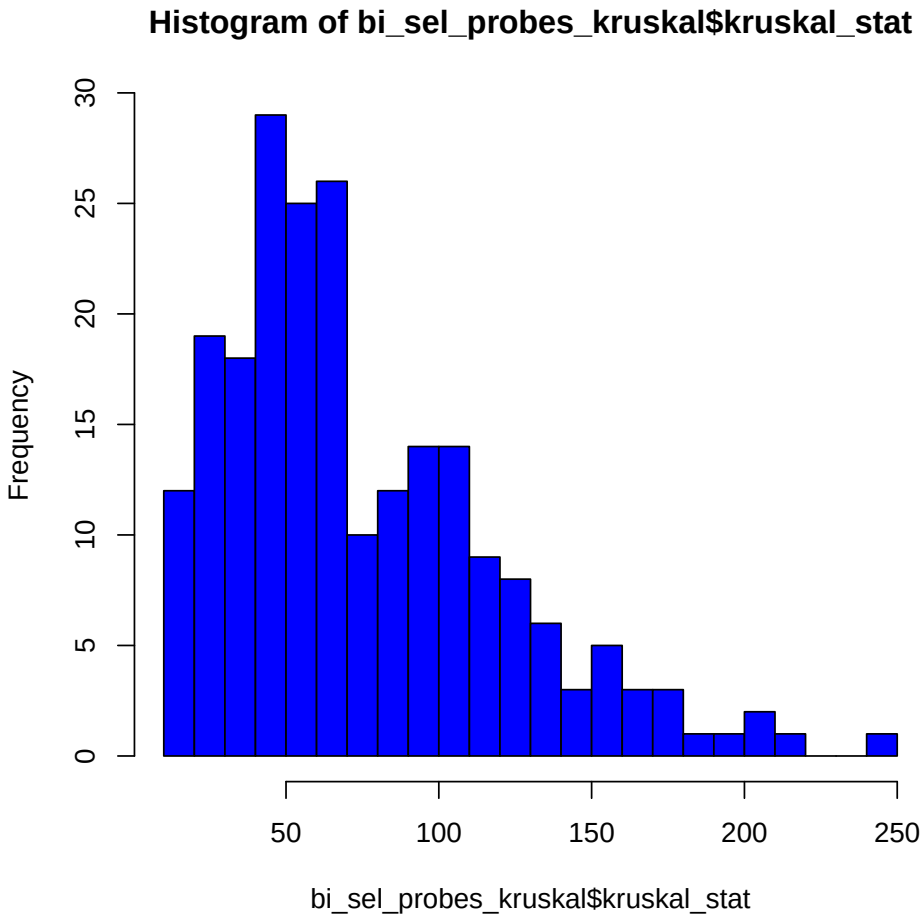

Figure 6: Histogram-probes-kruskal

```
> dim(bi_sel_probes_data)

[1] 222 394

> dim(bi_sel_probes_kru75_data)

[1] 133 394

> dim(bi_sel_probes_kru140_data)

[1] 202 394
```

7 Calculation of metagene-values for all 394 samples

Variables:

metagenes_probelists ==> lists of probesets representing the different metagenes
n_metagenes ==> number of different metagenes
metag_means ==> matrix for metagene means of all samples
cd_probes_c ==> probeset-names as character from rownames of cd_compl_394

Variable used from above: cd_compl_394

load probelists for metagenes and respective header-line

```
> metagenes_probelists <- read.delim("TNBC_Metag_ProbesLists.txt", h = T)
> n_metagenes = ncol(metagenes_probelists)
> metag_means = matrix(0, ncol = ncol(cd_compl_394), nrow = n_metagenes)
> cd_probes_c = rownames(cd_compl_394)

> for (i in 1:n_metagenes) {
+     probes_of_metag = as.character(metagenes_probelists[, i])
+     metag_subset_index_vector = (cd_probes_c %in% probes_of_metag)
+     metag_mean_TF = by(as.matrix(cd_compl_394), metag_subset_index_vector,
+         mean)
+     metag_means[i, ] = metag_mean_TF$"TRUE"
+ }
```

Summarize results:

```
> rownames(metag_means) = colnames(metagenes_probelists)
> colnames(metag_means) = colnames(cd_compl_394)
> metag_df = as.data.frame(t(metag_means))
> rm(n_metagenes, metag_means, cd_probes_c, probes_of_metag, metag_subset_index_vector,
+     metag_mean_TF)
```

8 calculate correlation of bimodal probesets to metagenes (for 3 different lists of bimodal probesets)

kruskal-cutoff<75 (n=133 probesets)

```
> chipdata = bi_sel_probes_kru75_data
> length_chipdata = length(chipdata)
> t_chipdata = as.data.frame(t(chipdata))
> Cor_metagenes = matrix(data = NA, nrow = length(t_chipdata), ncol = 0)
```

loop for selection of metagene

```
> for (SelMetag in c("IL8", "VEGF", "Prolif", "Basal", "CLDN3", "Apocrine",
+     "Histone", "Adipocyte", "Stroma", "IFN", "MHC1", "T.Cell", "MHC2",
+     "B.Cell", "Hemoglobin", "HOX")) {
+     SelectedMetag_data = metag_df[SelMetag]
+     temp_cor = cor(SelectedMetag_data, t_chipdata, use = "pairwise.complete.obs")
+     Cor_metagenes = cbind(Cor_metagenes, t(temp_cor))
+ }

> bi_sel_probes_kru75_Cor = Cor_metagenes
```

kruskal-cutoff<140 (n=202 probesets)

```
> chipdata = bi_sel_probes_kru140_data
> length_chipdata = length(chipdata)
```

transpose chipdata

```
> t_chipdata = as.data.frame(t(chipdata))
> Cor_metagenes = matrix(data = NA, nrow = length(t_chipdata), ncol = 0)
```

loop for selection of metagene

```
> for (SelMetag in c("IL8", "VEGF", "Prolif", "Basal", "CLDN3", "Apocrine",
+     "Histone", "Adipocyte", "Stroma", "IFN", "MHC1", "T.Cell", "MHC2",
+     "B.Cell", "Hemoglobin", "HOX")) {
+     SelectedMetag_data = metag_df[SelMetag]
+     temp_cor = cor(SelectedMetag_data, t_chipdata, use = "pairwise.complete.obs")
+     Cor_metagenes = cbind(Cor_metagenes, t(temp_cor))
+ }

> bi_sel_probes_kru140_Cor = Cor_metagenes
```

```
> chipdata = bi_sel_probes_data
> length_chipdata = length(chipdata)
> t_chipdata = as.data.frame(t(chipdata))
> Cor_metagenes = matrix(data = NA, nrow = length(t_chipdata), ncol = 0)
```

loop or selection of metagene

```
> for (SelMetag in c("IL8", "VEGF", "Prolif", "Basal", "CLDN3", "Apocrine",
+     "Histone", "Adipocyte", "Stroma", "IFN", "MHC1", "T.Cell", "MHC2",
+     "B.Cell", "Hemoglobin", "HOX")) {
+     SelectedMetag_data = metag_df[SelMetag]
+     temp_cor = cor(SelectedMetag_data, t_chipdata, use = "pairwise.complete.obs")
+     Cor_metagenes = cbind(Cor_metagenes, t(temp_cor))
+ }

> bi_sel_probes_Cor = Cor_metagenes
```

remove temporary variables:

```
> rm(chipdata, length_chipdata, t_chipdata)
```

9 Assign each probeset to metagene with highest correlation

Analysis for 133 probeset list from Kruskal-75-cutoff

```
> CorrMatrix = bi_sel_probes_kru75_Cor
> MaxAbsCorr = as.numeric(by(abs(CorrMatrix), c(1:nrow(CorrMatrix)),
+     max))
> indexvect = vector(mode = "logical", length = ncol(CorrMatrix))
> names = colnames(CorrMatrix)
> namelist = vector(mode = "character", length = nrow(CorrMatrix))
> MaxCorr = vector(mode = "numeric", length = nrow(CorrMatrix))

> for (i in c(1:nrow(CorrMatrix))) {
+     indexvect = abs(CorrMatrix[i, ]) == MaxAbsCorr[i]
+     namelist[i] = names[indexvect]
+     MaxCorr[i] = CorrMatrix[i, indexvect]
+ }
```

Additional number-coded-metagene-list for sorting

```
> namelist_coded = namelist
> metag_sort = c("IL8", "Histone", "CLDN3", "HOX", "Stroma", "Adipocyte",
+     "Hemoglobin", "Prolif", "Basal", "Apocrine", "VEGF", "MHC1", "IFN",
+     "MHC2", "T.Cell", "B.Cell")
```

```
> for (i in c(1:16)) {
+     namelist_coded[grep(metag_sort[i], namelist)] = i
+ }
```

Replace each metagene as ordered by number

```
> bi_sel_probes_kru75_Cor = data.frame(bi_sel_probes_kru75_Cor, namelist,
+     namelist_coded, MaxCorr, MaxAbsCorr, stringsAsFactors = FALSE)
```

Analysis for probeset list from Kruskal-140-cutoff (n=202)

```
> CorrMatrix = bi_sel_probes_kru140_Cor
> MaxAbsCorr = as.numeric(by(abs(CorrMatrix), c(1:nrow(CorrMatrix)),
+     max))
> indexvect = vector(mode = "logical", length = ncol(CorrMatrix))
> names = colnames(CorrMatrix)
> namelist = vector(mode = "character", length = nrow(CorrMatrix))
> MaxCorr = vector(mode = "numeric", length = nrow(CorrMatrix))

> for (i in c(1:nrow(CorrMatrix))) {
+     indexvect = abs(CorrMatrix[i, ]) == MaxAbsCorr[i]
+     namelist[i] = names[indexvect]
+     MaxCorr[i] = CorrMatrix[i, indexvect]
+ }
> namelist_coded = namelist
> metag_sort = c("IL8", "Histone", "CLDN3", "HOX", "Stroma", "Adipocyte",
+     "Hemoglobin", "Prolif", "Basal", "Apocrine", "VEGF", "MHC1", "IFN",
+     "MHC2", "T.Cell", "B.Cell")

> for (i in c(1:16)) {
+     namelist_coded[grep(metag_sort[i], namelist)] = i
+ }

> bi_sel_probes_kru140_Cor = data.frame(bi_sel_probes_kru140_Cor, namelist,
+     namelist_coded, MaxCorr, MaxAbsCorr, stringsAsFactors = FALSE)
```

Analysis for probeset list without Kruskal-cutoff (n=222)

```
> CorrMatrix = bi_sel_probes_Cor
> MaxAbsCorr = as.numeric(by(abs(CorrMatrix), c(1:nrow(CorrMatrix)),
+     max))
> indexvect = vector(mode = "logical", length = ncol(CorrMatrix))
> names = colnames(CorrMatrix)
> namelist = vector(mode = "character", length = nrow(CorrMatrix))
> MaxCorr = vector(mode = "numeric", length = nrow(CorrMatrix))
```

```
> for (i in c(1:nrow(CorrMatrix))) {
+     indexvect = abs(CorrMatrix[i, ]) == MaxAbsCorr[i]
+     namelist[i] = names[indexvect]
+     MaxCorr[i] = CorrMatrix[i, indexvect]
+ }

> namelist_coded = namelist
> metag_sort = c("IL8", "Histone", "CLDN3", "HOX", "Stroma", "Adipocyte",
+     "Hemoglobin", "Prolif", "Basal", "Apocrine", "VEGF", "MHC1", "IFN",
+     "MHC2", "T.Cell", "B.Cell")

> for (i in c(1:16)) {
+     namelist_coded[grep(metag_sort[i], namelist)] = i
+ }

> bi_sel_probes_Cor = data.frame(bi_sel_probes_Cor, namelist, namelist_coded,
+     MaxCorr, MaxAbsCorr, stringsAsFactors = FALSE)
```

Remove temporary variables

```
> rm(namelist, namelist_coded, metag_sort, names, indexvect, CorrMatrix,
+     MaxCorr, MaxAbsCorr)
```

10 Generate probeset output table

```
> corr_cutoff = 0.2
> corr_heatmap = bi_sel_probes_kru75_Cor
> corr_heatmap[corr_heatmap[, "MaxAbsCorr"] < corr_cutoff, ][, "namelist"] = "unclassified"
> corr_heatmap[corr_heatmap[, "MaxAbsCorr"] < corr_cutoff, ][, "namelist_coded"] = 0
> corr_heatmap = corr_heatmap[order(as.numeric(corr_heatmap[, "namelist_coded"]),
+     corr_heatmap[, "MaxAbsCorr"]), , ]
> out_table = corr_heatmap[, c("namelist", "namelist_coded", "MaxCorr",
+     "MaxAbsCorr")]
```

11 Generate correlation heatmap in R

Sort probesets ASCENDING according metagenes namelist_coded for heatmap

```
> sort.corr_heatmap = corr_heatmap[order(as.numeric(corr_heatmap[, "namelist_coded"]),
+     corr_heatmap[, "MaxAbsCorr"], decreasing = TRUE), ]
> sort.corr_heatmap = t(as.matrix(sort.corr_heatmap[, c("IL8", "Histone",
+     "CLDN3", "HOX", "Stroma", "Adipocyte", "Hemoglobin", "Prolif",
+     "Basal", "Apocrine", "VEGF", "MHC1", "IFN", "MHC2", "T.Cell", "B.Cell")]))
```

Generate heatmap-figure

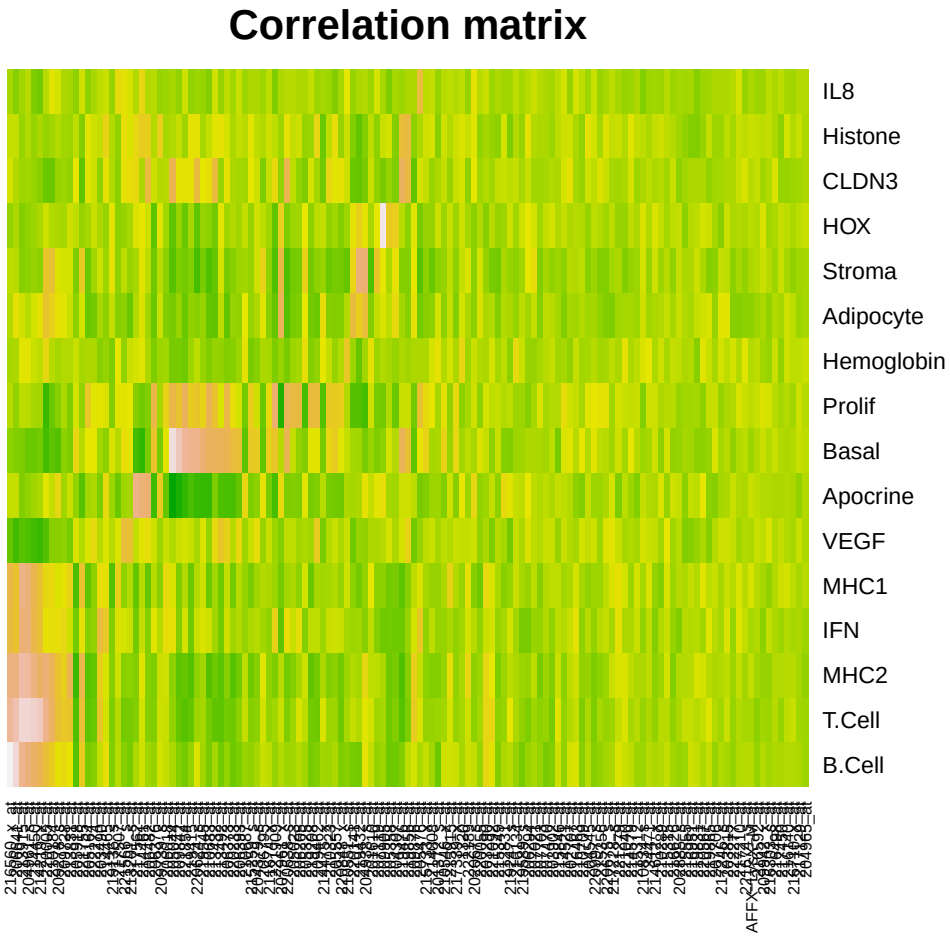

Figure 7: Heatmap

Retain variables:

cd_compl_579
cd_compl_394
remove unnecessary data which can be re-generated:
tcdm_579 used for comparab calculation
tcdm_394 used for kruskal calculation

```
> rm(tcdm_579, tcdm_394)
```

12 Saves

```
> save.image("Thomas.RData")
```

13 Session Information

```
> sessionInfo()

R version 2.10.1 (2009-12-14)
sparc-sun-solaris2.10

locale:
[1] /en_US.ISO8859-1/C/en_US.ISO8859-1/en_US.ISO8859-1/C/C

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

other attached packages:
[1] ClassDiscovery_2.10.1 mclust_3.4.3          cluster_1.12.1
[4] PreProcess_2.10.0     oompaBase_2.10.1
```