plasma: Interpretation

Kevin R. Coombes

Contents

Introduction

In the previous vignette (`plasma.Rmd`), we showed that the `plasma` algorithm was able to generate a joint (or "composite") model to predict overall survival of esophageal cancer patients in an independent test set.

Methods

Our computational method is implemented in the `plasma` package.

```
suppressWarnings( library(plasma) )
packageVersion("plasma")
```

```
## [1] '0.9.9'
```

Data

The results included here are in whole or part based upon data generated by the TCGA Research Network. We downloaded the entire esophageal cancer Level 3 data set (Cancer Genome Atlas Research Network, 2017)

from the FireBrowse web site (http://firebrowse.org/) (Deng *et al.*, 2017) in August 2018. They are also available from the Genomics Data Commons (GDC) (Jensen *et al.*, 2017). We filtered the data sets so that only the most variable, and presumably the most informative, features were retained. Here, we load the model produced in the previous vignette by analyzing this sample data set.

```
U <- url("http://silicovore.com/data/plESCA.Rda")
created <- load(U, .GlobalEnv)
close(U)
rm(U, created)
```

Interpretation of the Model

At this point, our model appears to be a fairly complex black box. We have constructed a matrix of components, based on linear combinations of actual features in different omics data sets. These components can then be combined in yet another linear model that predicts the time-to-event outcome through Cox regression. In this section, we want to explore how the individual features from different omics data sets contribute to different model components.

Our first act toward opening the black box is to realize that not all of the components discovered from the individual omics data sets survived into the final composite model. Some components were eliminated because they appeared to be nearly linearly related to components found in other omics data sets. So, we can examine the final composite model more closely.

```
pl@fullModel
```

```
## Call:
## coxph(formula = Surv(Days, vital_status == "dead") ~ ClinicalBin1 +
##     MAF1 + Meth4501 + Meth4502 + Meth4503 + Meth4504 + mRNASeq1 +
##     RPPA2 + RPPA3, data = riskDF)
##
##                 coef exp(coef) se(coef)      z        p
## ClinicalBin1  0.7458    2.1082   0.4767  1.564 0.117715
## MAF1          2.2183    9.1914   0.8789  2.524 0.011602
## Meth4501     -0.1429    0.8668   0.0712 -2.007 0.044727
## Meth4502      1.6820    5.3761   0.8465  1.987 0.046930
## Meth4503      1.1703    3.2228   0.3373  3.469 0.000522
## Meth4504     -1.8067    0.1642   0.7077 -2.553 0.010683
## mRNASeq1      0.2624    1.3001   0.1003  2.615 0.008911
## RPPA2         0.9545    2.5975   0.2887  3.306 0.000947
## RPPA3         0.8029    2.2321   0.3239  2.479 0.013169
##
## Likelihood ratio test=60.41  on 9 df, p=1.12e-09
## n= 185, number of events= 77
```

```
temp <- terms(pl@fullModel)
mainterms <- attr(temp, "term.labels")
rm(temp)
mainterms
```

```
## [1] "ClinicalBin1" "MAF1"         "Meth4501"     "Meth4502"     "Meth4503"     "Meth4504"
## [7] "mRNASeq1"     "RPPA2"        "RPPA3"
```

We see that at least one component discovered from four of the five "true" omics data sets survived in the final model; only the miR components failed to make the cut. In addition, one component from the binary clinical data was retained in the final model.

Final (Composite) Weights

A key point to note is that the core of the `plasma` model consists of a composition of two levels of linear models. One model starts with individual omics data sets and predicts the "*components*" that we have learned across all data sets. The second model uses the components as predictors in a Cox proportional hazards model of time-to-event (i.e., overall survival) data. The composite of these two steps is yet another linear model. In particular, we should be able to compute the matrix that links the features from the omics-level data sets directly to the final survival ourtcome.

```
FW <- getFinalWeights(pl)
summary(FW)
```

```
##      Weight              Source            Feature             Standard
##  Min.   :-2.2797033   Length:5716        Length:5716        Min.   :-6.12730
##  1st Qu.:-0.0061276   Class :character   Class :character   1st Qu.:-0.63986
##  Median :-0.0001781   Mode  :character   Mode  :character   Median :-0.06496
##  Mean   :-0.0028050                                        Mean   : 0.00000
##  3rd Qu.: 0.0011486                                        3rd Qu.: 0.61131
##  Max.   : 2.4060943                                        Max.   : 5.69114
```

```
library(beanplot)
library(Polychrome)
data(palette36)
foo <- computeDistances(palette36[3:36])
colist <- as.list(palette36[names(foo)[1:7]])
rm(foo, palette36)
beanplot(Standard ~ Source, data = FW, what = c(1, 1, 1, 0), col = colist,
         main = "Standardized Feature Weights")
abline(h=c(1.96, -1.96), col = "gray")
```

In order to interpret the model, we would like to convert as many features as possible into gene names. This is easy with the mutation, mRNASeq, and RPPA data sets. We can also get genes (possibly more than one per locus) out of the methylation data set by referring to the Illumina annotations.

```
# Convert feature names into gene names, based on the source.
features2Genes <- function(df, translators) {
  genes <- rep("", nrow(df))
  for (N in unique(df$Source)) {
    FUN <- translators[[N]]
    if(is.null(FUN)) FUN = function(X) ""
    where <- which(df$Source == N)
    genes[where] <- FUN(df$Feature[where])
  }
  genes
}

illumina <- read.csv(file.path("https://webdata.illumina.com/downloads/productfiles",
                               "humanmethylation450/humanmethylation450_15017482_v1-2.csv"),
                     skip=7)
illumina <- illumina[, c("IlmnID", "UCSC_RefGene_Name")] # only care about two columns
illumina2genes <- function(Y) {
  illuminaFeatures <- illumina[which(illumina$IlmnID %in% Y),] # which did we use?
  temp <- strsplit(illuminaFeatures$UCSC_RefGene_Name, ";") # look at duplicates
  temp <- sapply(sapply(temp, unique), paste, collapse=";") # many dups have the same gene name
  illuminaFeatures$UCSC_RefGene_Name <- temp
  temp
```

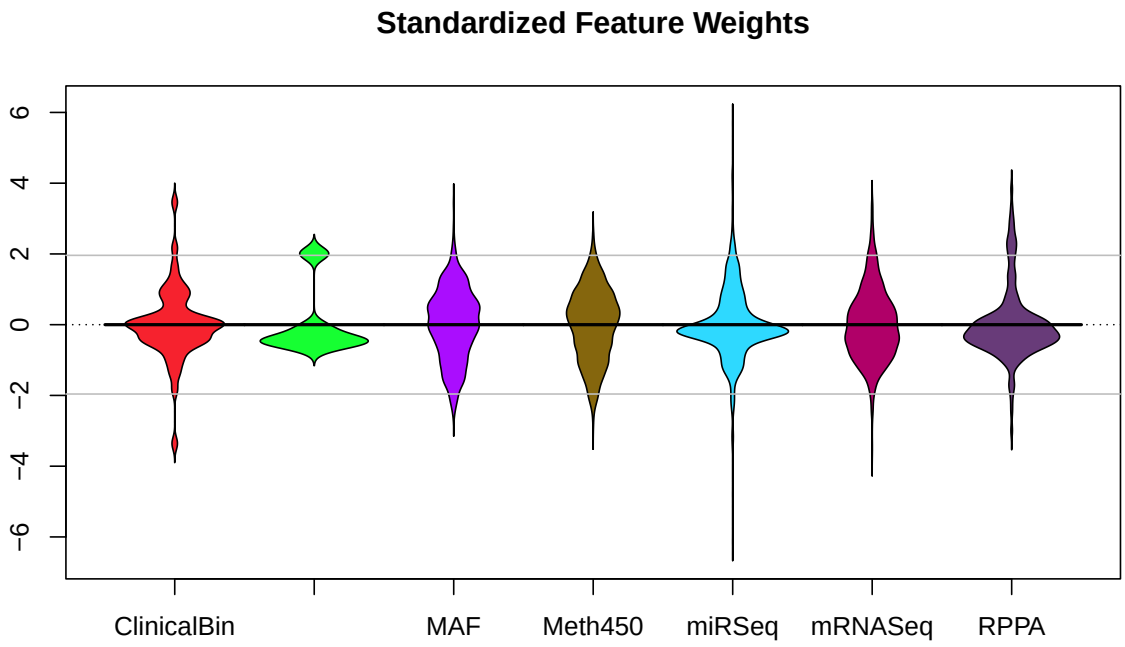

Figure 1: Distribution of standardized feature weights in the final model, by data set.

```
}

xlate <- list(RPPA = function(Y) { sapply(strsplit(Y, "\\."), function(x) x[1]) },
              MAF = function(Y) {Y},
              mRNASeq = function(Y) { sapply(strsplit(Y, "\\."), function(x) x[1]) },
              Meth450 = illumina2genes)

FW$Gene <- features2Genes(FW, xlate)
```

Now we collect together all the significant genes, from whatever data set.

```
library(tidyr)
expanded <- as.data.frame(separate_rows(FW, Gene, sep=";"))
collapsed <- expanded[abs(expanded$Standard) > 1.96, ] # standard normal significance
collapsed$Source = factor(collapsed$Source)
geneset <- unique(collapsed$Gene)
length(geneset)
```

```
## [1] 227
```

We write the gene set into a text file, then call a perl script., 'toppgene.pl', to annotate it at the ToppGene web site (https://toppgene.cchmc.org/) (Chen *et al.*, 2009).

```
##
topp <- system.file('perl', 'toppgene.pl', package = 'plasma')
input <- file.path(system.file('GeneLists', package = 'plasma'), "finalGenes.txt")
if (!file.exists(input)) {
```

```
  write.table(geneset, file = input,
              quote = FALSE, row.names = FALSE, col.names = FALSE)
}
output <- sub("GeneLists", "TGOUT", sub("txt", "tsv", input))
if (!file.exists(output)) {
  cmd <- paste("C:/Strawberry/perl/bin/perl.exe", topp, input, output)
  sysout <- system(cmd)
}
```

Lists of Significant Genes by Component

Here we combine the weight matrices for features from all data sets into all components ionto a single data frame.

```
CW <- combineAllWeights(pl)
contra <- CW@combined
datasrc <- CW@dataSource
```

Next, we pick out the significant features for each component, using the following "`pickers`" function:

```
pickers <- function(dsname, Q = 0.05) {
  pickA <- interpret(CW, dsname, Q)
  rap <- pickA$Feature[pickA$Source == "RPPA"]
  rap <- sapply(strsplit(rap, "\\."), function(x) x[1])
  map <- pickA$Feature[pickA$Source == "MAF"]
  nap <- pickA$Feature[pickA$Source == "mRNASeq"]
  nap <- sapply(strsplit(nap, "\\."), function(x) x[1])
  nap <- nap[-1]
  nick <- rep("", nrow(pickA))
  nick[pickA$Source == "RPPA"] <- rap
  nick[pickA$Source == "MAF"] <- map
  nick[pickA$Source == "mRNASeq"] <- c("", nap)
  pickA$Nickname <- nick

  illuminaFeatures <- illumina[which(illumina$IlmnID %in% pickA$Feature),] # which did we use?
  temp <- strsplit(illuminaFeatures$UCSC_RefGene_Name, ";") # look at duplicates
  temp <- sapply(sapply(temp, unique), paste, collapse=";") # many dups have the same gene name
  illuminaFeatures$UCSC_RefGene_Name <- temp
  expanded <- as.data.frame(separate_rows(illuminaFeatures, UCSC_RefGene_Name, sep=";"))
  M <- merge(pickA, expanded, by.x = "Feature", by.y = "IlmnID", all.x = TRUE, sort = FALSE)
  M$Nickname[M$Nickname == ""] <- M$UCSC_RefGene_Name[M$Nickname == ""]
  M$Nickname[is.na(M$Nickname)] <- ""
  M$UCSC_RefGene_Name <- NULL
  M <- M[order(M$Source, M$Feature), ]
  M
}
```

```
allPicks <- lapply(mainterms, pickers, Q = 0.05)
names(allPicks) <- mainterms
sapply(allPicks, dim)
```

```
##      ClinicalBin1 MAF1 Meth4501 Meth4502 Meth4503 Meth4504 mRNASeq1 RPPA2 RPPA3
## [1,]          297  286      332      304      330      316      321   251   318
## [2,]            4    4        4        4        4        4        4     4     4
```

We extract the names of the significant genes for each component. These lists are based on the mutation, mRNASeq, RPPA, and methylation data. Each gene list is stored in a separate output file to be used later.

```
gldir <- system.file("GeneLists", package = "plasma")
if (!file.exists(gldir)) stop("Missing GeneLists directory")

allGenes <- NULL
for (I in 1:length(allPicks)) {
  N <- names(allPicks)[I]
  fname <- file.path(gldir, paste(N, "txt", sep = "."))
  genes <- allPicks[[I]]$Nickname
  genes <- sort(unique(genes[genes != ""]))
  allGenes <- c(allGenes, genes)
  if (file.exists(fname)) next
  write.table(genes, file = fname,
              sep = "\t", quote = FALSE, row.names = FALSE, col.names = FALSE)
}
length(allGenes)
allGenes <- unique(allGenes)
length(allGenes)
allout <- file.path(gldir, "allGenes.txt")
if (!file.exists(allout)) {
  write.table(allGenes, file = allout,
              sep = "\t", quote = FALSE, row.names = FALSE, col.names = FALSE)
}
```

ToppGene queries for each component

Now we call a perl script (`toppgene.pl`) to automate the process of retrieving the results of each of the enrichment analyses on each component, based on their individual gene lists. Each result is again written to a separate file, in tab-separated-values (TSV) format.

```
outdir <- system.file("TGOUT", package = "plasma")
if (!file.exists(outdir)) stop("Missing TGOUT directory")

fullnames <- file.path(gldir, dir(gldir, pattern = "*.txt"))
for (I in 1:length(fullnames)) {
  R <- file.path(outdir, sub("txt$", "tsv", basename(fullnames[I])))
  if(file.exists(R)) next
  topp <- system.file('perl', 'toppgene.pl', package = 'plasma')
  cmd <- paste("C:/Strawberry/perl/bin/perl.exe", topp, fullnames[[I]], R)
  sysout <- system(cmd)
  if (sysout) {
    stop("Received error code ", sysout, "\n")
  }
  Sys.sleep(10)
}
```

We can now use the database annotations from the ToppGene analyses to cluster the components (**Figure 16**). Interestingly, the clusters of components based on overlapping annotations are different from those defined by shared features.

```
outdir <- system.file("TGOUT", package = "plasma")
fl <- dir(outdir, pattern = "tsv$")
toppresults <- lapply(1:length(fl), function(J) {
```

```
  sheet <- read.table(file.path(outdir, fl[J]),
                      header = TRUE, sep = "\t", fill = TRUE, quote = "")
  sheet <- sheet[sheet$Category %in% c("Cytoband", "Disease", "GeneOntologyBiologicalProcess",
                                       "GeneOntologyCellularComponent", "HumanPhenotype", "Pathway") ,]
  sheet$Category <- factor(sheet$Category)
  sheet$Component <- sub(".txt", "", fl[J])
  sheet
})

enrichment <- do.call(rbind, toppresults)
enrichment$Component <- sub(".tsv", "", enrichment$Component)
enrichment <- enrichment[enrichment$Component %in% c("allGenes", "finalGenes", mainterms),]
enrichment$Component <- factor(enrichment$Component)
dim(enrichment)
```

```
## [1] 1623   15
```

```
usn <- unique(enrichment$Name)
shortnames <- substring(usn, 1, 38)
enrichment$Name <- substring(enrichment$Name, 1, 38)

library(ggplot2)
library(ClassDiscovery)
```

```
## Loading required package: cluster
```

```
## Loading required package: oompaBase
```

```
bubbly <- function(cat) {
  hark <- enrichment[enrichment$Category == cat, ]
  tap <- tapply(hark$PValue, list(hark$Name, hark$Component), mean)
  tap <- -log10(tap)
  tap[is.na(tap)] <- 0
  dim(tap)
  colnames(tap) <- sub(".tsv", "", colnames(tap))
  hc <- hclust(distanceMatrix(tap, "pearson"), "ward.D2")
  gc <- hclust(distanceMatrix(t(tap), "pearson"), "ward.D2")
  foo <- gc$labels[gc$order]
  hark$Name <- factor(hark$Name, levels = foo)
  boo <- hc$labels[hc$order]
  hark$Component <- factor(hark$Component, levels = boo)
  p <- ggplot(hark, aes(x = Name , y = Component, size = -log10(PValue), col = -log10(PValue))) +
    geom_point(alpha=0.5) +
    scale_size(range = c(0.1, 6), name="enrichment") +
    scale_color_gradientn(colours = viridisLite::plasma(16)) +
    theme(axis.text.x = element_text(angle = -45, vjust = 0.5, hjust=0))
  list(hc = hc, gc = gc, tap = tap, p = p)
}

bubbly("Disease")$p

bubbly("Pathway")$p

bubbly("GeneOntologyBiologicalProcess")$p

bubbly("GeneOntologyCellularComponent")$p
```

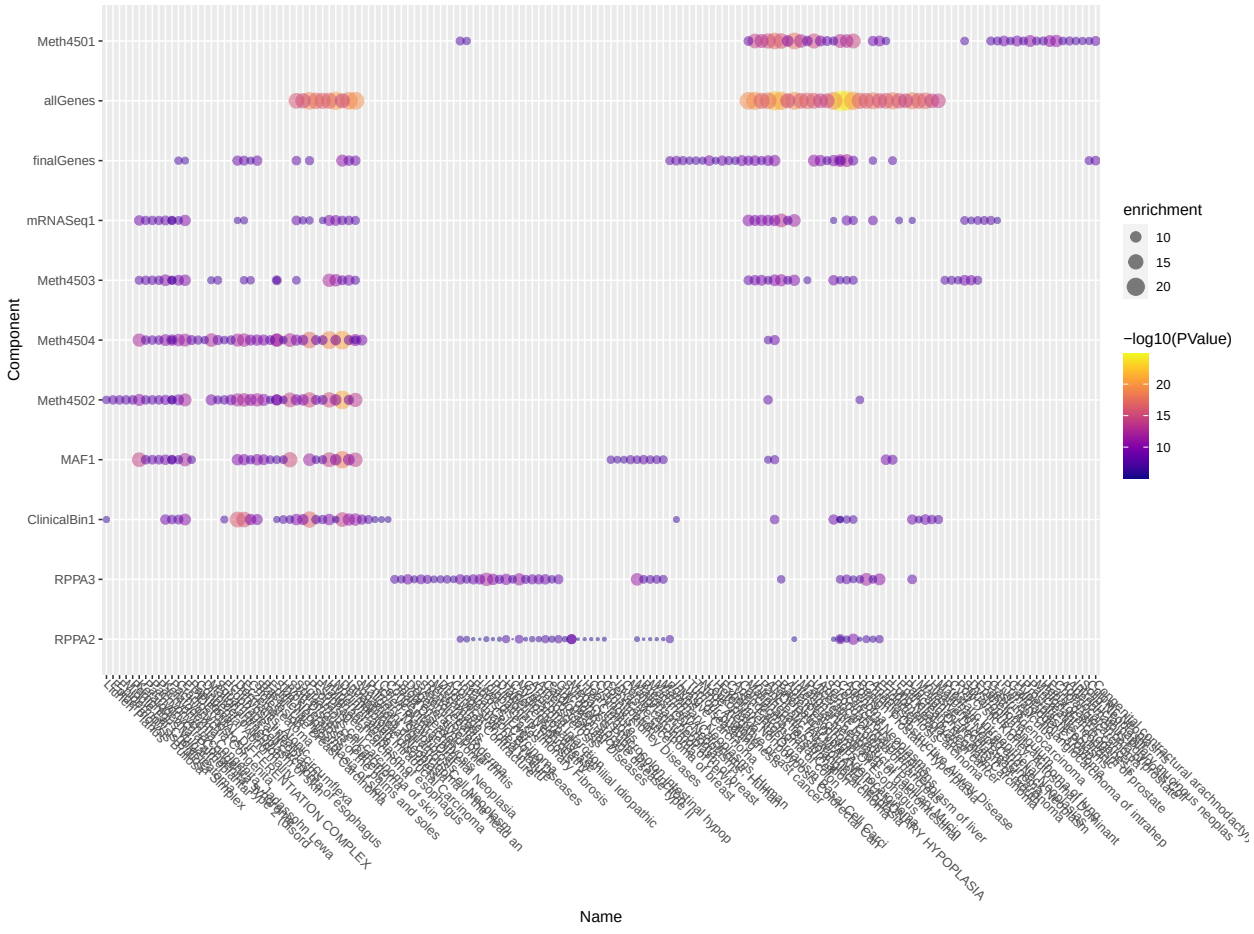

Figure 2: Clustering components based on disease annotation p-values.

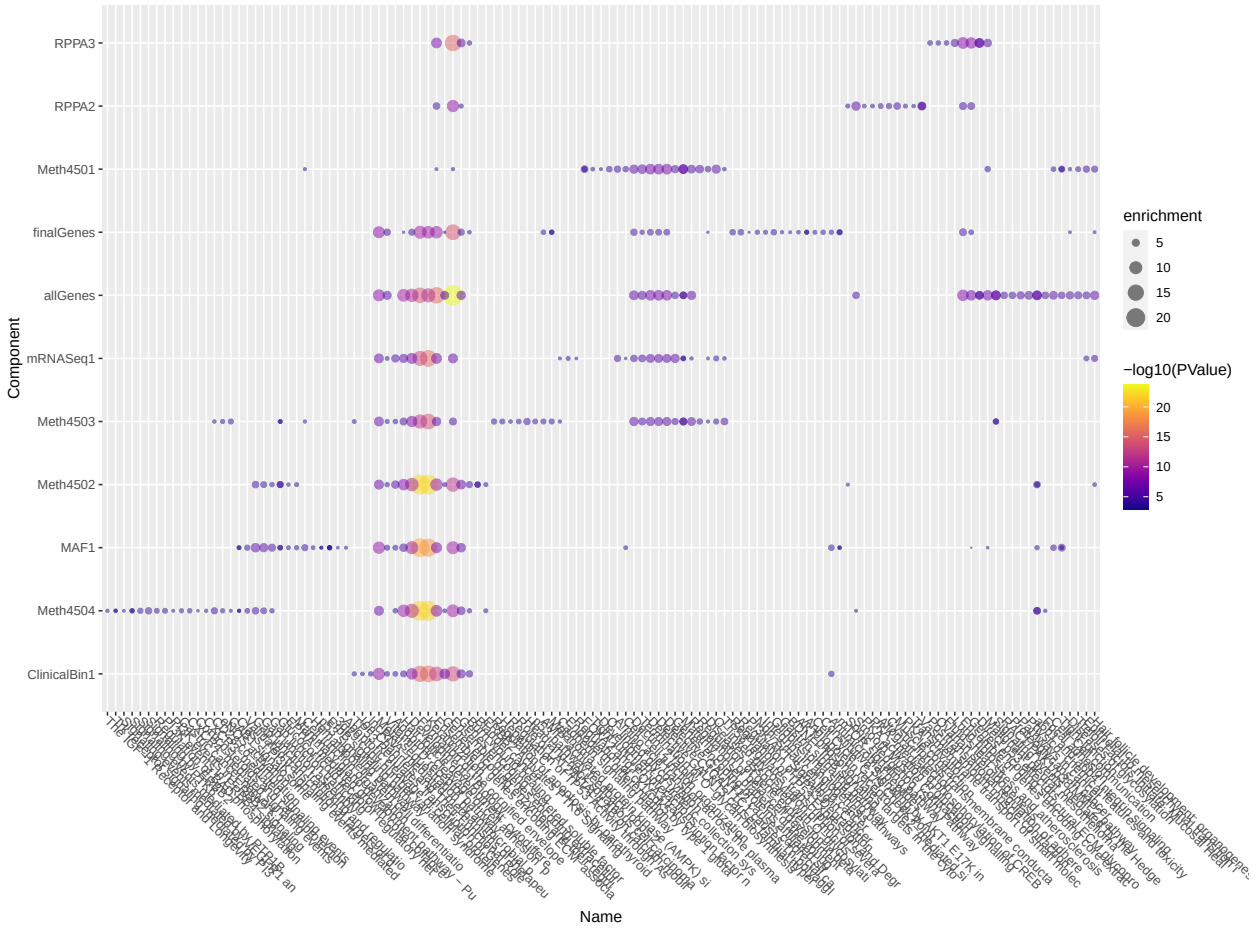

Figure 3: Clustering components based on pathway annotation p-values.

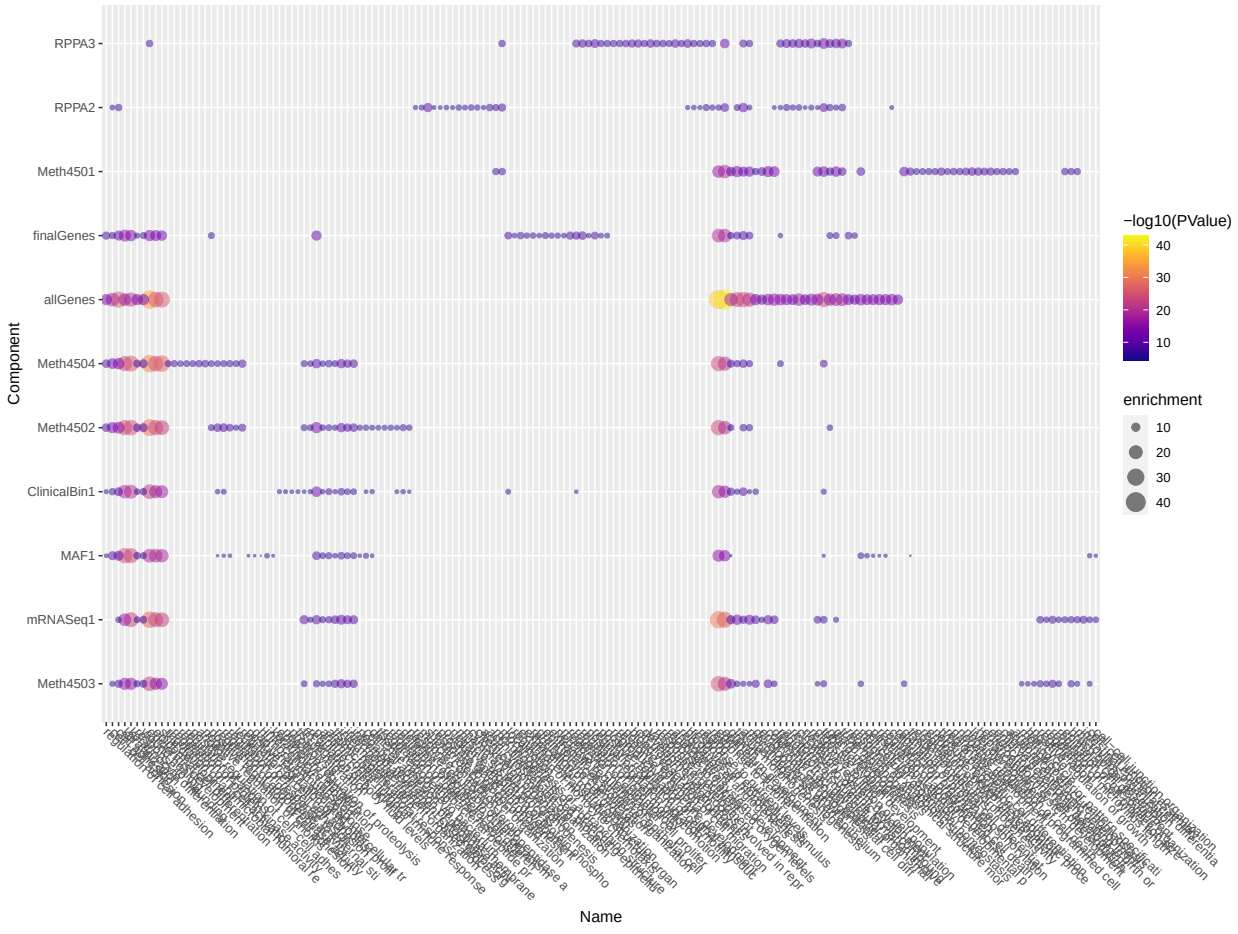

Figure 4: Clustering components based on bioogical process annotation p-values.

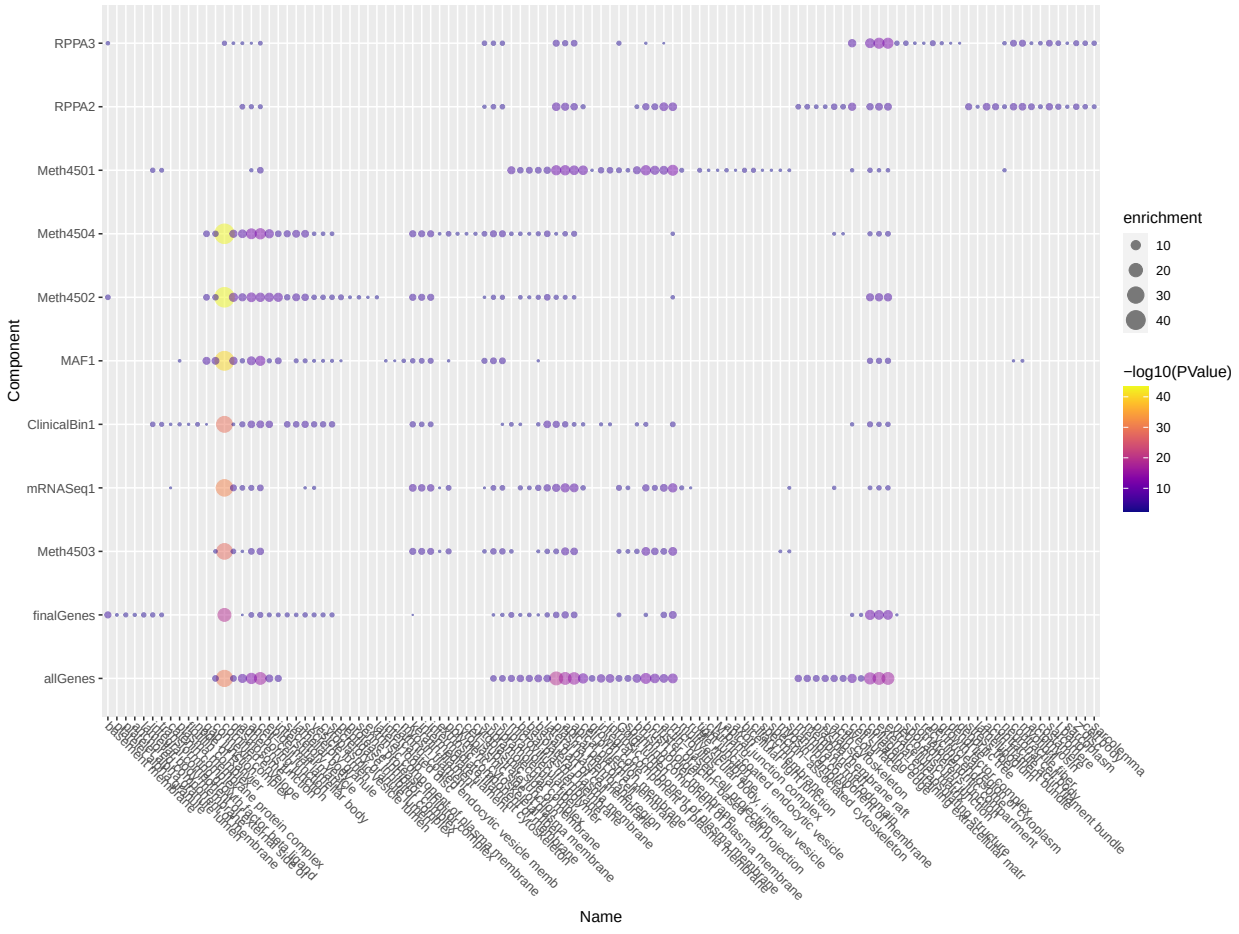

Figure 5: Clustering components based on cellular comnponent annotation p-values.

Interpretation by Data Set

Our interest now turns to understanding how the features from individual omics data sets contribute to the components that are used in the final model. As mentioned earlier, these contributions are mediated through two levels of linear regression models when extending a model from data set A to data set B. A linear combination of features from set B is used to define the secondary level of components; then a linear combination of these components is used to predict the components of the single Cox model built that had been from set A. These weights can be combined and extracted using the `getAllWeights` function, and can then be explored.

Clinical Binary Data

We use the binary clinical data set to begin illustrating one method for interpreting the components.

```
library(oompaBase)
HG <-  blueyellow(64)
cbin <- getAllWeights(pl, "ClinicalBin")
compcolors <- c("forestgreen", "purple")[1 + 1*(colnames(cbin@contrib) %in% mainterms)]
heat(cbin, cexCol = 0.9, cexRow = 0.5, col = HG, ColSideColors = compcolors)
```

Figure 6 shows the raw weights for each clinical binary feature in all of the original omics components. We would like to simplify this plot in several ways. First, we can remove any components that were not retained in the final model (indicated by the green color bar in the top dendrogram). Second, we hypothesize that some components intrinsically have a wider spread of weights, and that it might be more important to scale the components consistently to look at the relative contributions. Finally, we can remove any features that seem to make no contributions to any of the components; that is; those that do not have highly ranked weights (by absolute value) in any component.

```
shrink <- function(dset, N) {
  dset@contrib <- scale(dset@contrib)                # standardize
  feat <- unique(unlist(as.list(getTop(dset, N))))  # remove useless features
  dset@contrib <- dset@contrib[feat, mainterms]     # remove unused components
  dset
}

xbin <- shrink(cbin, 4)
heat(xbin, cexCol = 0.9, cexRow = 0.9, col = HG)
```

In **Figure 7**, we can identify strong contrasts between several pairs of variables. For example, one set of components is enriched with white, never smokers, who still have evidence of tumors, at stage T3 and grade 3 in the lower third of the esophagus (ICD-10 code C15.5), while another group is enriched for Asian, current smokers, who are tumor-free, with stage N0, T2 tumors from the lower third of the esophagus (ICD-10 code C15.4).

mRNA-Sequencing Data

We can apply the same method to visualize contributors from each of the omics data sets. As a second illustration, we look at the standardized weights from the mRNA data set in the components that are part of the final model, keeping only those features that are highly ranked by absolute weight in at least one component (**Figure 8**).

```
mrna <- getAllWeights(pl, "mRNASeq")
xmrna <- shrink(mrna, 7)
tmp <- rownames(xmrna@contrib)
```

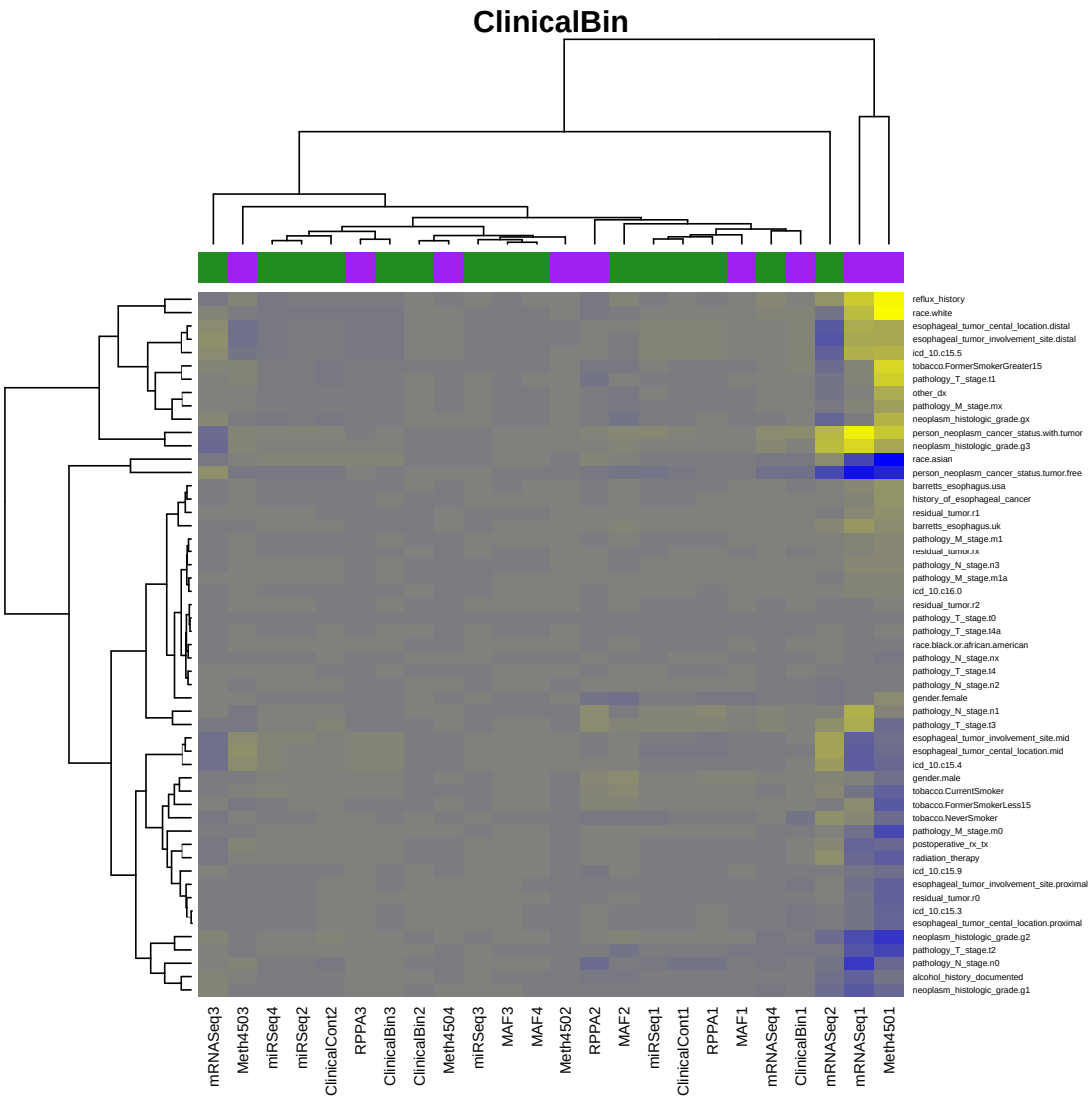

Figure 6: Unscaled heatmap of the contributions of binary clinical features to all components (purple = retained in final model, green = not retained).

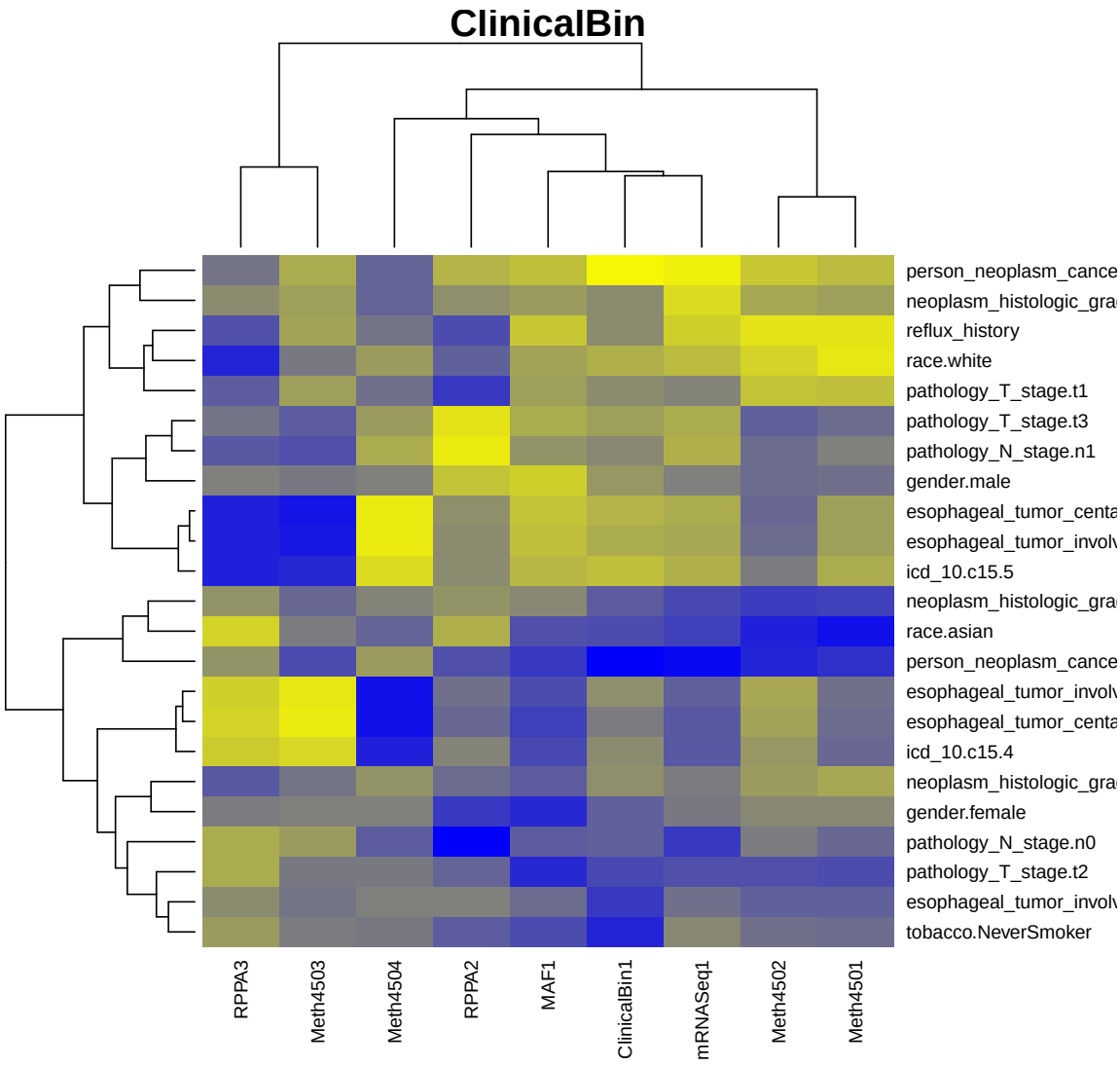

Figure 7: Scaled heatmap of the contributions of filtered binary clinical features to important components.

```
rownames(xmrna@contrib) <- sapply(strsplit(tmp, "\\."), function(x) x[1])
heat(xmrna, cexCol = 0.9, cexRow = 0.6, col = HG)
```

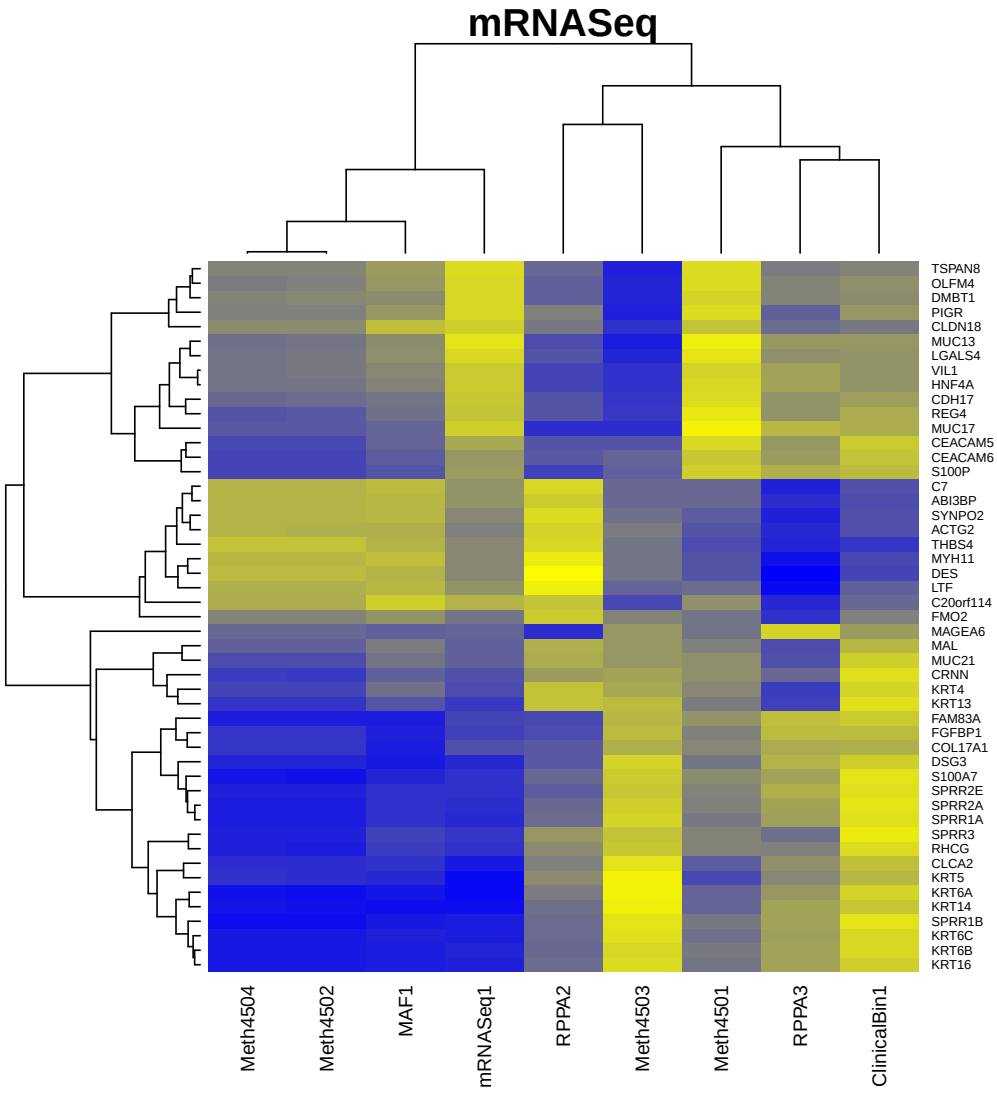

Figure 8: Scaled heatmap of the contributions of filtered mRNA features to important components

Uniting the Contributors

One difficulty with the heatmaps in the previous section is that they are focused on individual input data sets, and not on individual components. In order to fully understand which features contribute, for example, to the first mutation component (MAF1), one would have to scan all the heatmaps from all the datasets and then try to combine the influences. In order to help with that procedure, we can merge all the contributions into a single data frame, with an accompanying factor tracking the source data set.

Figure 9 displays the mean, standard deviation (SD), median, and median absolute deviation (MAD) of the weight-contributions for each data set in each component.

```
image(CW)
```

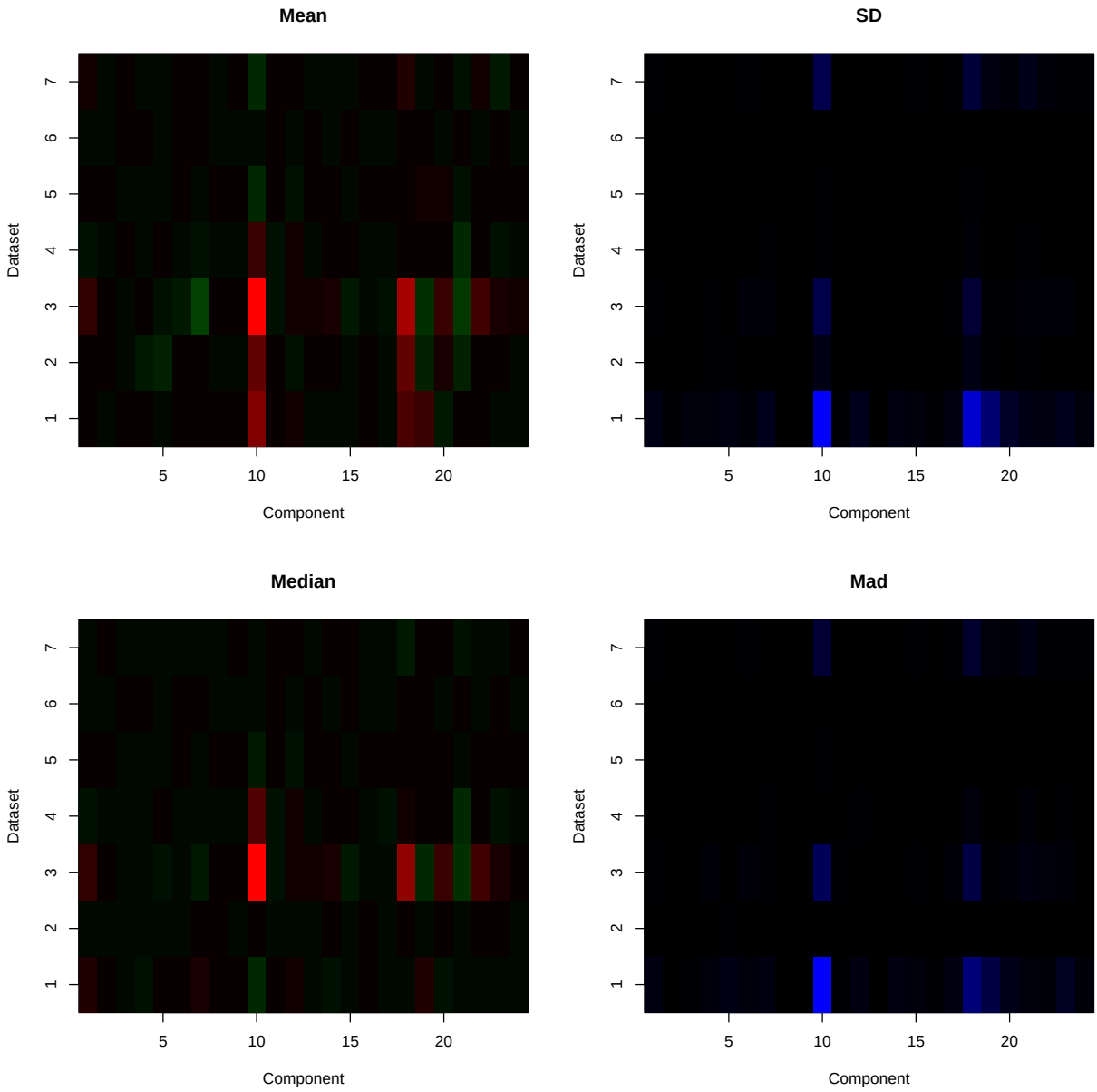

Figure 9: Summary statistics of weights by component and dataset.

Standardized Weights

To remain consistent with the previous heatmaps, we have standardized the weights in each data set and component. In **Figure 10**, we create beanplots showing the distributions of weights arising from each data set in each (retained) component. (Similar plots for the raw weights are available in **Supplementary Data**.) Some data sets have very different contribution patterns than others. For example, the miRSeq data set appears to have significant outliers making large contributions in almost every component, the MAF and RPPA data sets also frequently (but not always) include such outliers.

```
library(beanplot)
library(Polychrome)
data(palette36)
foo <- computeDistances(palette36[3:36])
colist <- as.list(palette36[names(foo)[1:7]])
brute <- stdize(CW, "standard")
opar <- par(mfrow = c(3,3), mai = c(0.2,0.2, 0.5, 0.2))
for (i in which(colnames(contra) %in% mainterms)) {
  beanplot(brute[, i] ~ datasrc, what = c(1,1,1,0), col = colist,
           main = paste("Std Wts, Component", i))
}

par(opar)
rm(opar)
```

We also include plots of the histograms of distributions by component (**Figure 11**). None of these really looks quite normal; almost all have some slightly odd shape.

```
opar <- par(mfrow = c(3, 3), mai = c(0.2,0.2, 0.5, 0.2))
for (i in which(colnames(contra) %in% mainterms)) {
  hist(brute[,i], breaks = 77, main = paste("Component", i))
}

par(opar)
rm(opar)
```

Data Set Sources of Top Twenty Lists

Next, we want to see how many items in the lists of "top twenty" contributers to each component come from each data set. The results are shown in **Figure 12**. Using the raw weights, the vast majority of contributions come from the clinical binary data, with secondary contributions from the MAF and RPPA data sets (as we expected from the above distribution plots). After standardization, most of the contributions arise from miRs, but the methylation, mRNA and, to a lesser extent, the MAF and RPPA data sets also are present.

```
top20 <- apply(contra, 2, function(X) {
  A <- abs(X)
  S <- rev(sort(A))
  which(A > S[21])
})
top20types <- apply(top20, 2, function(X) {
  table(datasrc[X])
})

chop20 <- apply(brute, 2, function(X) {
  A <- abs(X)
  S <- rev(sort(A))
```

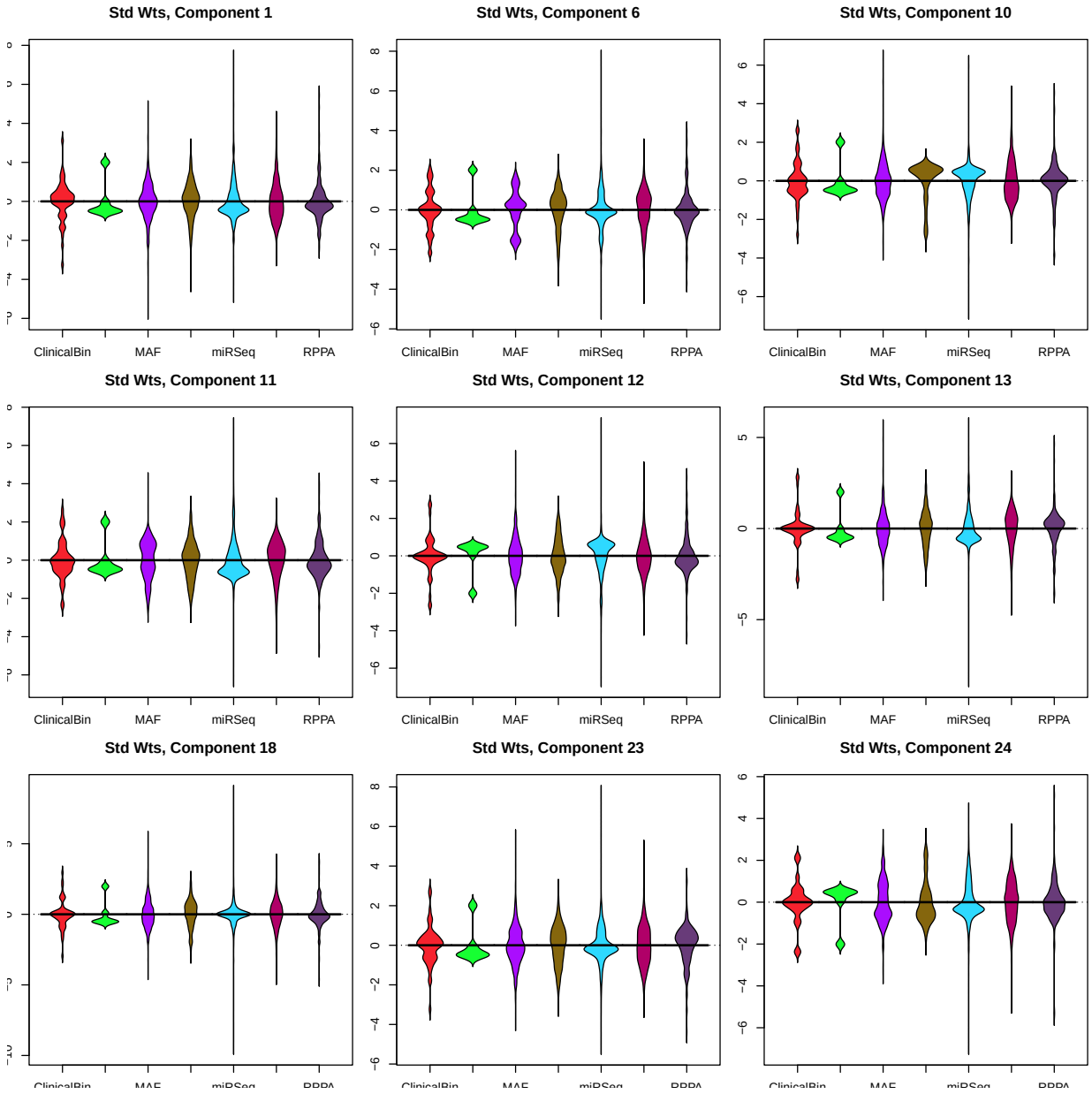

Figure 10: Distributions of standardized weights by data set and component.

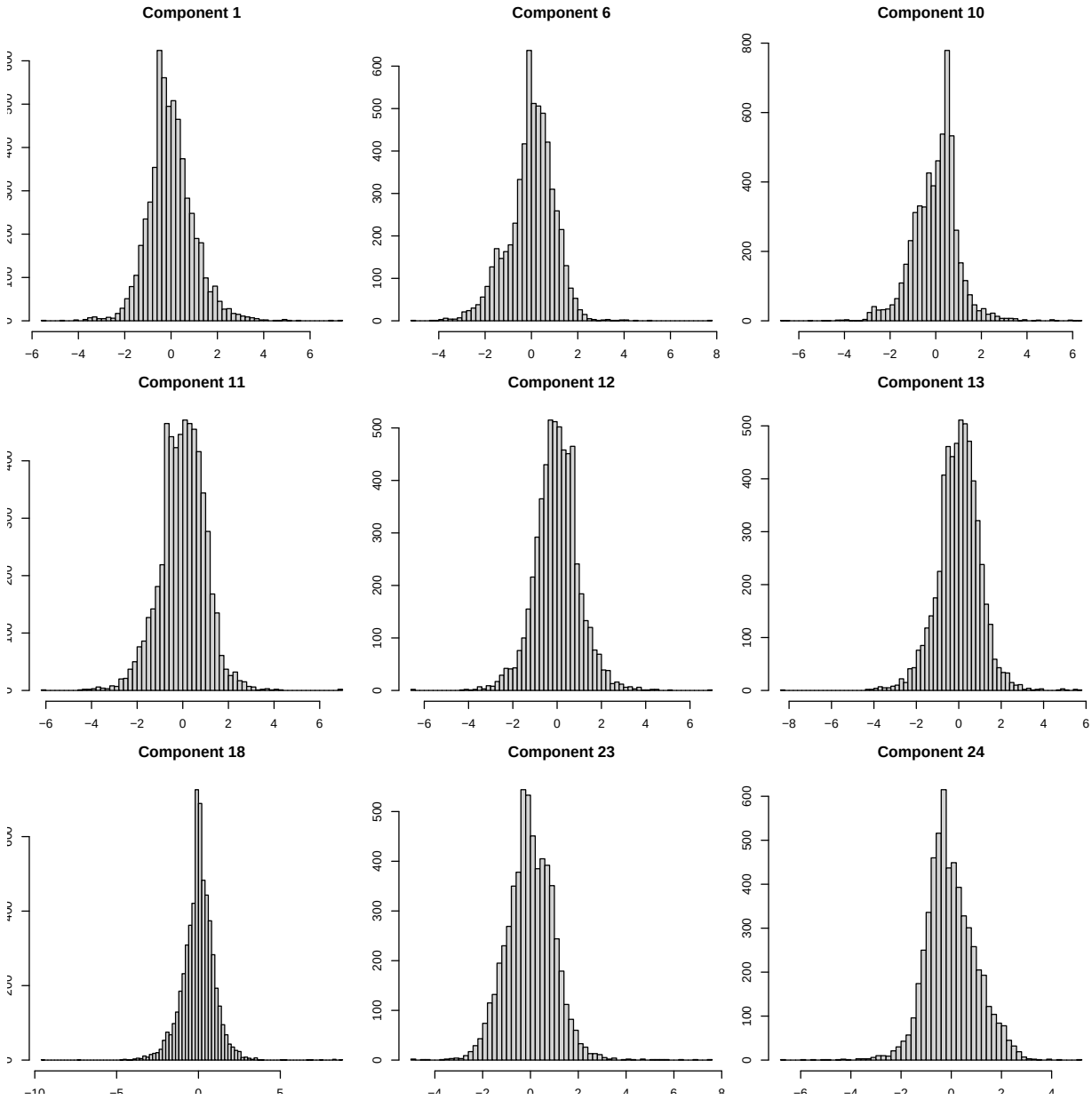

Figure 11: Histogram of standardized weights by component.

```
  which(A > S[21])
})
chop20types <- apply(chop20, 2, function(X) {
  table(datasrc[X])
})
```

```
opar <- par(mfrow = c(1, 2), mai = c(1.02, 0.82, 1.22, 0.32))
image(1:7, 1:24, top20types, ylab = "Components", xlab = "Data Sets")
mtext(levels(datasrc), side = 3, at = 1:7, line = 1/2, las=2)
mtext("Raw", side = 3, at = 0, line = 2, font = 2, cex = 1.2)
image(1:7, 1:24, chop20types, ylab = "Components", xlab = "Data Sets")
mtext(levels(datasrc), side = 3, at = 1:7, line = 1/2, las=2)
mtext("Std", side = 3, at = 0, line = 2, font = 2, cex = 1.2)
```

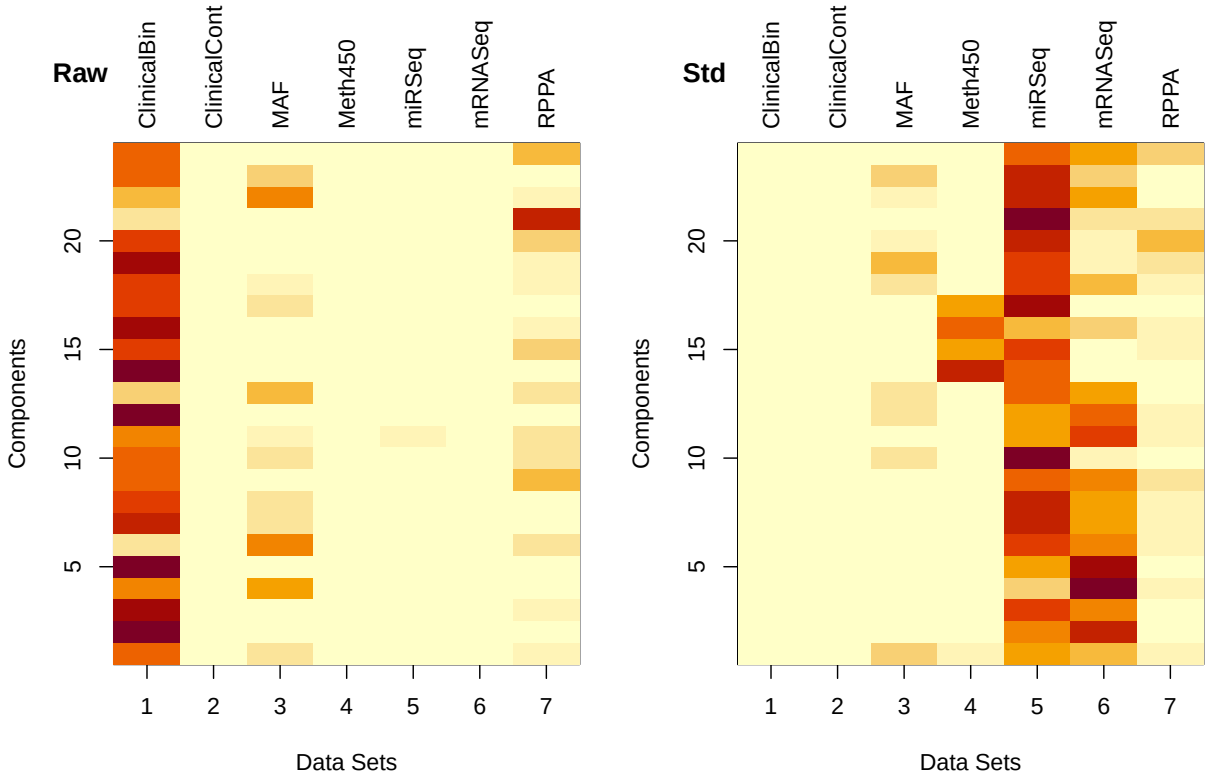

Figure 12: Number of contributers to top twenty lists.

```
par(opar)
rm(opar)
```

Overall Distribution of Weights

In **Figure 13**, we pool all the weights (across all data sets and all components) to look at histograms of the distributions. We also overlay the theoretical normal distribution that one would expect to see. Using the

usual mean and distribution (right panel), the actual weights are slightly more conservative (i.e., concentrated near zero) than expected. This figure suggests that we might want to use standardization, and decide on significance purely from the theoretical normal distribution rather than from deviations away from that distribution.

```
opar <- par(mfrow = c(1, 2))
hist(contra, breaks = 123, main = "Raw Weights", prob = TRUE)
xx <- seq(-10, 10, length=1001)
yy <- dnorm(xx, mean(contra), sd(contra))
lines(xx, yy, col = "red", lwd=2)
hist(brute, breaks = 123, main = "Standardized Weights", prob = TRUE)
yy <- dnorm(xx)
lines(xx, yy, col = "red", lwd=2)
```

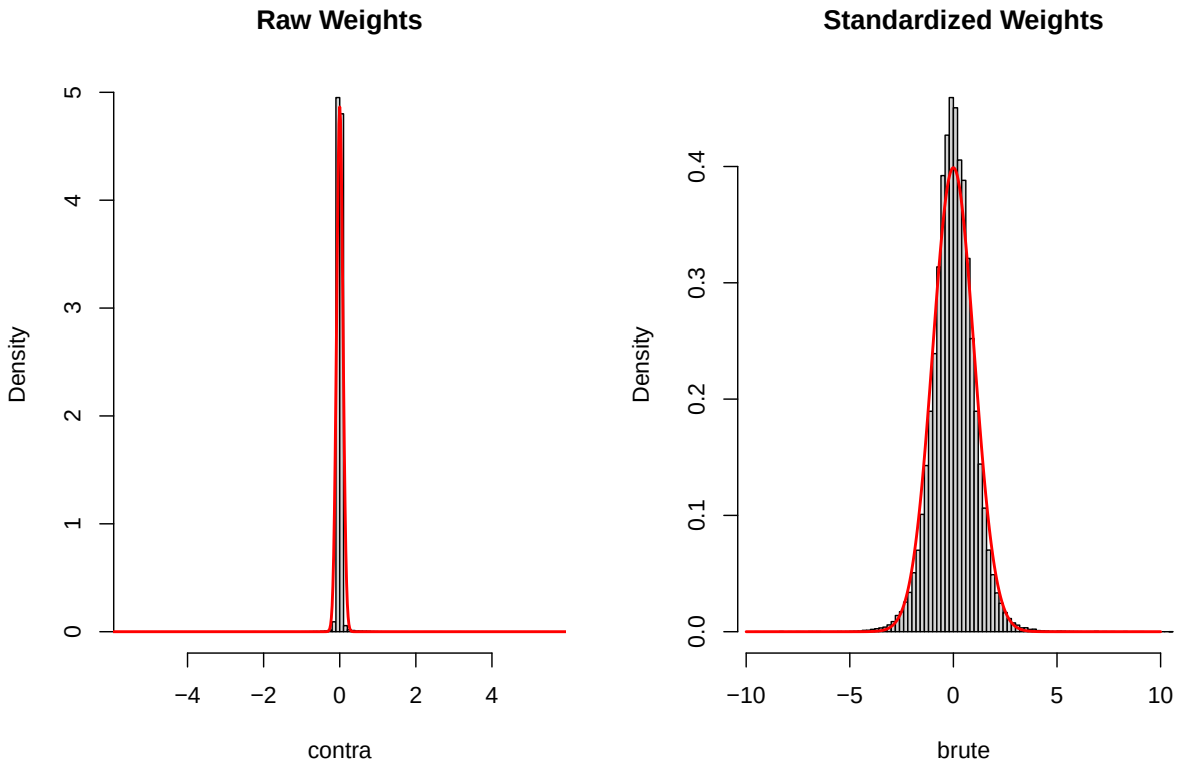

Figure 13: Histograms of all weights (combined).

```
par(opar)
rm(opar)
```

Number of Contributers Selected by Normal Significance

Finally, we create yet another image (**Figure 14**), counting the number of significant features from each data set for each component, when using a significance cutoff of 5% derived from the standard normal distribution. We feel that this result is more reasonable than any thing we got just by looking at the top 20 lists. Most contributions come from the biggest omics data sets (mRNA, methylation, and miR) with fewer from MAF, RPPA, and clinical binary.

```
Q <- qnorm(0.975) # two-sided 5% cutoff
top1p <- aggregate(brute, list(datasrc), function(X) sum(abs(X) > Q)) # top 5 percent
rownames(top1p) <- top1p[, 1]
top1p <- as.matrix(top1p[, -1])
top1p <- top1p[, mainterms]
opar <- par(mai = c(1.02, 0.82,1.22, 0.22))
image(1:7, 1:9, top1p, ylab = "Components", xlab = "Data Sets")
mtext(levels(datasrc), side = 3, at = 1:7, line = 1, las=2)
```

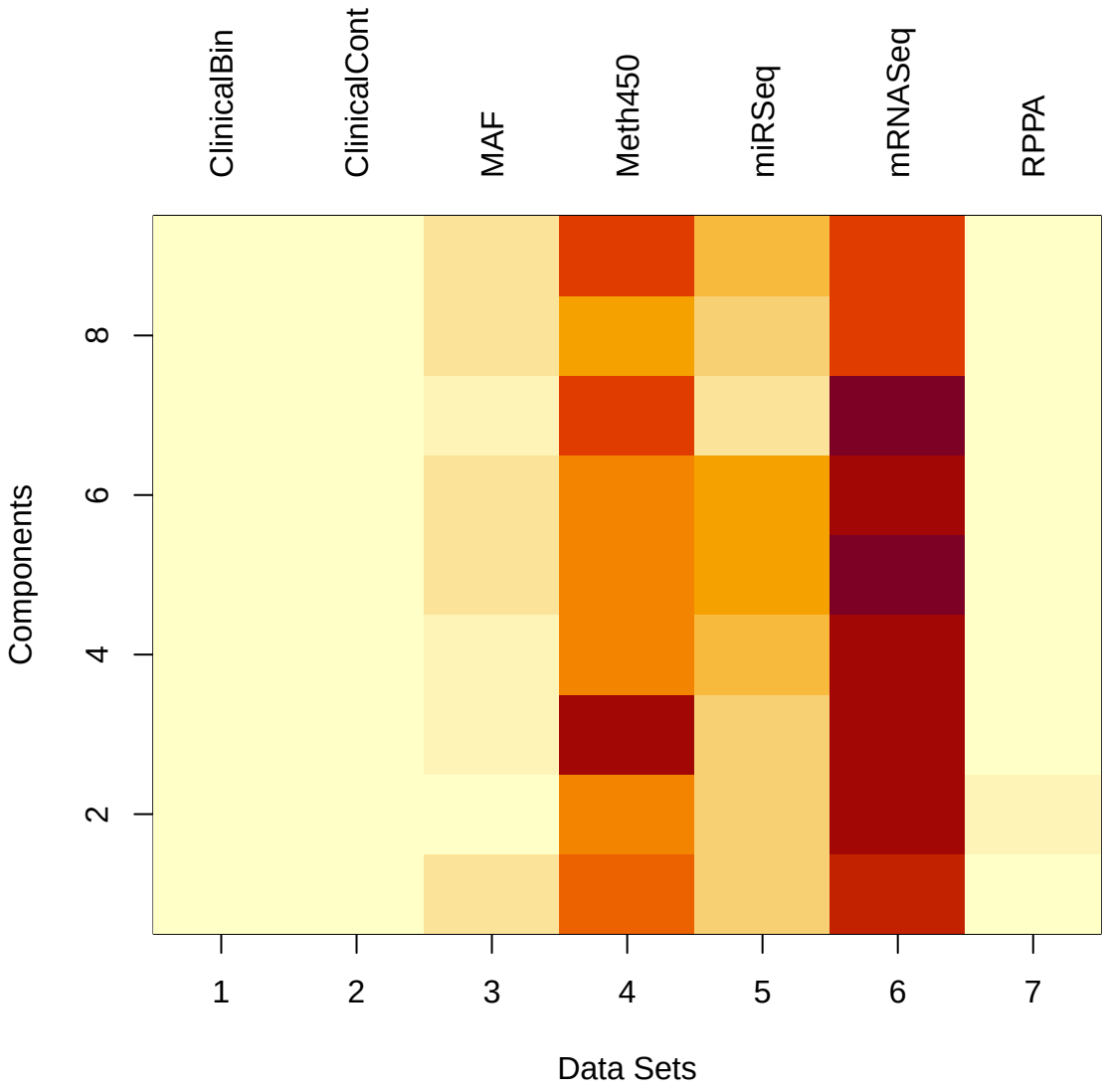

Figure 14: Number of significant contributions by data set and component.

```
par(opar)
```

Interpeting the MAF1 Component

In the final Cox proportional hazards model of overall survival, the component with the largest hazard ratio was "MAF1" (the first feature discovered from the MAF mutation data set), for which each one unit increase in the component corresponds to a 9-fold increase in the hazard. Our next goal is to find a biological interpretation of this component. Since many of the feature names obtained from TCGA include extra annotations that we won't use later, we are going to simplify them.

Here is an overview of all the contributing features.

```
pickers <- function(dsname, Q = 0.05) {
  pickA <- interpret(CW, dsname, Q)
  rap <- pickA$Feature[pickA$Source == "RPPA"]
  rap <- sapply(strsplit(rap, "\\."), function(x) x[1])
  map <- pickA$Feature[pickA$Source == "MAF"]
  nap <- pickA$Feature[pickA$Source == "mRNASeq"]
  nap <- sapply(strsplit(nap, "\\."), function(x) x[1])
  nap <- nap[-1]
  nick <- rep("", nrow(pickA))
  nick[pickA$Source == "RPPA"] <- rap
  nick[pickA$Source == "MAF"] <- map
  nick[pickA$Source == "mRNASeq"] <- c("", nap)
  pickA$Nickname <- nick
  pickA
}
pickA <- pickers("MAF1")
summary(pickA)
```

```
##    Feature                   Source        Weight         Nickname
##  Length:286         ClinicalBin :  3   Min.   :-5.058   Length:286
##  Class :character   ClinicalCont:  1   1st Qu.:-2.599   Class :character
##  Mode  :character   MAF         :  6   Median :-2.178   Mode  :character
##                     Meth450     : 83   Mean   :-1.216
##                     miRSeq      : 46   3rd Qu.: 1.961
##                     mRNASeq     :131   Max.   : 7.607
##                     RPPA        : 16
```

We start by looking more closely at the clinical features.

```
pickA[1:4, ]
```

```
##                                  Feature       Source    Weight Nickname
## gender.female               gender.female  ClinicalBin -2.153483
## gender.male                   gender.male  ClinicalBin  2.113973
## pathology_T_stage.t2 pathology_T_stage.t2  ClinicalBin -2.170436
## weight                             weight ClinicalCont  2.010542
```

Because of our decision to use one-hot encoding of categorical variables, our data set includes separate features for "male" and"female". Both terms are strongly related to the MAF1 component, but (as one would hope) with approximately equal standardized weights but opposite signs. Being female decreases the hazard; being male increases it. Since the coefficient of MAF1 in the final model of overall survival is itself positive, we can infer the direction that the hazard changes. It is harder to "eyeball" the magnitude of the change in the hazard, since these coefficients only measure the relative contribution of these factors to the MAF1 component.

Here are the mutated genes (from the MAF data set) associated with the component "MAF1".

```
pickA[pickA$Source == "MAF", ]
```

```
##         Feature Source    Weight Nickname
## CFAP54   CFAP54    MAF -2.045903   CFAP54
## DNAH9     DNAH9    MAF -2.061517    DNAH9
## DYNC2H1 DYNC2H1    MAF -1.994866  DYNC2H1
## FBN3       FBN3    MAF -2.029872     FBN3
## MYH13     MYH13    MAF -2.008811    MYH13
## PCDHA12 PCDHA12    MAF -1.995470  PCDHA12
```

Note that they all have negative coefficients, meaning that having these mutations decreases the effect of this component. Since the coefficient of `MAF1` in the final model of overall survival is itself positive, that means that having any of these mutations decreases the hazard for that patient. Here are the Entrez Gene descriptions of the genes:

CFAP54 (Cilia And Flagella Associated Protein 54) Predicted to be involved in cilium assembly; cilium movement involved in cell motility; and spermatogenesis. Predicted to act upstream of or within cerebrospinal fluid circulation; motile cilium assembly; and mucociliary clearance. Predicted to be located in axoneme.

DNAH9 (Dynein Axonemal Heavy Chain 9) This gene encodes the heavy chain subunit of axonemal dynein, a large multi-subunit molecular motor. Axonemal dynein attaches to microtubules and hydrolyzes ATP to mediate the movement of cilia and flagella.

DYNC2H! (Dynein Cytoplasmic 2 Heavy Chain 1) This gene encodes a large cytoplasmic dynein protein that is involved in retrograde transport in the cilium and has a role in intraflagellar transport, a process required for ciliary/flagellar assembly. Mutations in this gene cause a heterogeneous spectrum of conditions related to altered primary cilium function and often involve polydactyly, abnormal skeletogenesis, and polycystic kidneys.

FBN3 (Fibrillin 3) This gene encodes a member of the fibrillin protein family. Fibrillins are extracellular matrix molecules that assemble into microfibrils in many connective tissues. This gene is most highly expressed in fetal tissues and its protein product is localized to extracellular microfibrils of developing skeletal elements, skin, lung, kidney, and skeletal muscle. This gene is potentially involved in Weill-Marchesani syndrome.

MYH13 (Myosin Heavy Chain 13) Predicted to enable microfilament motor activity. Predicted to be involved in muscle contraction. Predicted to act upstream of or within cellular response to starvation. Located in extracellular exosome.

PCDHA12 (Protocadherin Alpha 12) This gene is a member of the protocadherin alpha gene cluster, one of three related gene clusters tandemly linked on chromosome five that demonstrate an unusual genomic organization similar to that of B-cell and T-cell receptor gene clusters. The alpha gene cluster is composed of 15 cadherin superfamily genes related to the mouse CNR genes and consists of 13 highly similar and 2 more distantly related coding sequences. The tandem array of 15 N-terminal exons, or variable exons, are followed by downstream C-terminal exons, or constant exons, which are shared by all genes in the cluster. The large, uninterrupted N-terminal exons each encode six cadherin ectodomains while the C-terminal exons encode the cytoplasmic domain. These neural cadherin-like cell adhesion proteins are integral plasma membrane proteins that most likely play a critical role in the establishment and function of specific cell-cell connections in the brain.

These descriptions clearly indicate some common functional relationships between the mutated genes, including a role in cilia, flagella, and microfibrils.

We then extracted all gene names from the MAF, mRNASeq, and RPPA data sets and used thm to perform a gene enrichment (pathway) analysis using ToppGene (Chen *et al.*, 2009). Associated GeneOntology Biological Process categories included keratinization, epidermal and epithelial cell development, cell adhesion, intermediate filament organization, and wound healing. GeneOntology Cellular Components included cell-cell junctions, extracellular matrix, and intermediate filaments. Associated human phenotypes included hyperkeratosis (particularly follicular hyperkeratosis), epidermal thickening, and oral leukoplakia. Associated pathways included keratinization, gap junction assembly and trafficking, and both ErbB and mTOR signaling.

Associated cytogenetic regions included 1q21-1q22, 18q12.1, and 12q12.13. Associated gene families included cadherins, kallikreins, keratins, and gap-junction proteins. Associated diseases included hyperkertosis, squamous cell carcinoma of the head and neck, intraepithelial neoplasia, endometrial carcinoma, basal-like breast carcinoma, and esophageal carcinoma.

Conclusions

We have identified a method analogous to that of `MOFA` that allows us to combine different omics data without the need for prior imputation of missing values. A major difference is that while `MOFA` model learns "factors" that are composites of the variables in an unsupervised fashion, the `plasma` model learns "components" that are composites of the variables in an supervised fashion, using the outcomes "event" and "time-to-event" as response variables.

Although the factors from `MOFA` are defined such that the first factor, Factor 1, accounts for the greatest variance in the model, the factors may or may not be significantly associated with the outcome, and a post-hoc survival analysis would need to be done to assess this. It may be the case that some factors, although they are significantly associated with outcome, account for very small variance in the final `MOFA` model, which hinders interpretability. This was the case with the TCGA-ESCA dataset, in which, when 10 factors were learned from the `MOFA` model, only Factor 10 was significantly associated with survival, while accounting for [number] variance in the model [CITE SUPPLEMENTARY RESULTS?]. On the other hand, the components for `plasma` are created in a way that maximizes the covariance in the predictors and the response, and therefore these components will be automatically associated to some degree with the outcome. This could be advantageous in that dissecting the weights associated with the components would yield a list of variables from different omics datasets that contribute the most to defining the outcome, and any additional analyses could be refined by looking at these high-weighted variables most closely.

References

Cancer Genome Atlas Research Network (2017) Integrated genomic characterization of oesophageal carcinoma. *Nature*, **541**, 169–175.

Chen,J. *et al.* (2009) ToppGene suite for gene list enrichment analysis and candidate gene prioritization. *Nucleic Acids Res*, **37**, W305–11.

Deng,M. *et al.* (2017) FirebrowseR: An r client to the broad institute's firehose pipeline. *Database (Oxford)*, **2017**.

Jensen,M.A. *et al.* (2017) The NCI genomic data commons as an engine for precision medicine. *Blood*, **130**, 453–459.