plasma: Partial LeAst Squares for Multi-omics Analysis

Kevin R. Coombes

Contents

Introduction

Recent years have seen the development of numerous algorithms and computational packages for the analysis of multi-omics data sets. At this point, one can find multiple review articles summarizing progress in the field (Adossa, Khan, Rytkonen, and Elo 2021; Graw et al. 2021; Heo, Hwa, Lee, Park, and An 2021; Picard, Scott-Boyer, Bodein, Perin, and Droit 2021; Reel, Reel, Pearson, Trucco, and Jefferson 2021; Subramanian, Verma, Kumar, Jere, and Anamika 2020; Vlachavas, Bohn, Uckert, and Nurnberg 2021). As with other applications of machine learning, the kinds of problems addressed by these algorithms are divided into two categories: unsupervised (e.g., clustering or class discovery) or supervised (including class comparison and class prediction) (Simon and Dobbin 2003). Advances in the area of unsupervised learning have been broader and deeper than advances on supervised learning.

One of the most effective unsupervised methods is Multi-Omic Factor Analysis (MOFA) (Argelaguet et al. 2018, 2020). A key property of MOFA is that it does not require all omics assays to have been performed on all samples under study. In particular, it can effectively discover class structure across omics data sets even when data for many patients have only been acquired on a subset of the omics technologies. As of this

writing, we do not know of any supervised multi-omics method that can effectively learn to predict outcomes when samples have only been assayed on a subset of the omics data sets.

MOFA starts with a standard method – Latent Factor Analysis – that is known to work well on a single omics data set. It then fits a coherent model that identifies latent factors that are common to, and able to explain the data well in, all the omics data sets under study. Our investigation (unpublished) of the factors found by MOFA suggests that, at least in come cases, it is approximately equivalent to a two-step process:

1. Use principal components analysis to identify initial latent factors in each individual omics data set.
2. For each pair of omics data sets, use overlapping samples to train and extend models of each factor to the union of assayed samples.

That re-interpretation of MOFA suggests that an analogous procedure might work for supervised analyses as well. In this article, we describe a two-step algorithm, which we call "*plasma*", to find models that can predict time-to-event outcomes on samples from multi-omics data sets even in the presence of incomplete data. We use partial least squares (PLS) for both steps, using Cox regression to learn the single omics models and linear regression to learn how to extend models from one omics data set to another. To illustrate the method, we use a subset of the esophageal cancer (ESCA) data set from The Cancer Genome Atlas (TCGA).

Methods

Our computational method is implemented and the data are available in the `plasma` package.

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

```
## [1] '0.7.12'
```

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 Genomics Data Commons (GDC) (Jensen, Ferretti, Grossman, and Staudt 2017) on 6 August 2018. We filtered the data sets so that only the most variable, and presumably the most informative, features were retained. Here, we load this sample data set.

```
loadLUSCdata()
sapply(assemble, dim)
```

```
##      ClinicalBin ClinicalCont MAF Meth450 miRSeq mRNASeq RPPA
## [1,]          61            2 897    1940    872    2290  223
## [2,]         504          504 484     364    338     493  322
```

1. From TCGA, we obtained 162 columns of clinical, demographic, and laboratory data on 185 patients. We removed any columns that always contained the same value. We also removed any columns whose values were missing in more than 25% of the patients. We converted categorical variables into sets of binary variables using one-hot-encoding. We then separated the clinical data into three parts:
    1. Outcome (overall survival)
    2. Binary covariates (53 columns)
    3. Continuous covariates (6 columns)
2. Exome sequencing data for 184 patients with esophageal cancer was obtained as mutation allele format (MAF) files. We removed any gene that was mutated in fewer than 3% of the samples. The resulting data set contained 566 mutated genes.
3. Methylation data for 185 ESCA patients was obtained as beta values computed by the TCGA from Illumina Methylation 450K microarrays. We removed any CpG site for which the standard deviation of the beta values was less than 0.3. The resulting data set contained 1,454 highly variable CpG's.

4. Already normalized sequencing data on 2,566 microRNAs (miRs) was obtained for 185 patients. We removed any miR for which the standard deviation of normalized expression was less than 0.05, which left 926 miRs in the final data set.
5. Already normalized sequencing data on 20,531 mRNAs was obtained in 184 patients. We removed any mRNA whose mean normalized expression was less than 6 or whose standard deviation was less than 1.2. The final data set included 2,520 mRNAs.
6. Normalized expression data from reverse phase protein arrays (RPPA) was obtained from antibodies targeting 192 proteins in 126 patients. All data were retained for further analysis.

Finally, in order to be able to illustrate the ability of the plasma algorithm to work in the presence of missing data, we randomly selected 10% of the patients to remove from the miRSeq data set (leaving 166 patients) and 15% of the patients to remove from the mRNASeq data set (leaving 157 patients). We provid a summary of the outcome data below.

Imputation

We recommend imputing small amounts of missing data in the input data sets. The underlying issue is that the PLS models we use for individual omics data sets will not be able to make predictions on a sample if even one data point is missing. As a result, if a sample is missing at least one data point in every omics data set, then it will be impossible to use that sample at all.

For a range of available methods and R packages, consult the CRAN Task View on Missing Data. We also recommend the R-miss-tastic web site on missing data. Their simulations suggest that, for purposes of producing predictive models from omics data, the imputation method is not particularly important. Because of the latter finding, we have only implemented two simple imputation methods in the `plasma` package:

1. `meanModeImputer` will replace any missing data by the mean value of the observed data if there are more than five distinct values; otherwise, it will replace missing data by the mode. This approach works relatively well for both continuous data and for binary or small categorical data.
2. `samplingImputer` replaces missing values by sampling randomly from the observed data distribution.

```
set.seed(54321)
imputed <- lapply(assemble, samplingImputer)
```

Computational Approach

The `plasma` algorithm is based on Partial Least Squares (PLS), which has been shown to be an effective method for finding components that can predict clinically interesting outcomes (Bastien, Bertrand, Meyer, and Maumy-Bertrand 2015). The workflow of the plasma algorithm is illustrated in **Figure 1** in the case of three omics data sets. First, for each of the omics data sets, we apply the PLS Cox regression algorithm (`plsRcox` Version 1.7.6 (Bertrand and Maumy-Bertrand 2021)) to the time-to-event outcome data to learn three separate predictive models (indicated in red, green, and blue, respectively). Each of these models may be incomplete, since they are not defined for patients who have not been assayed (shown in white) using that particular omics technology. Second, for each pair of omics data sets, we apply the PLS linear regression algorithm (`pls` Version 2.8.0 (Mishra and Liland 2022)) to learn how to predict the coefficients of the Cox regression components from one data set using features from the other data set. This step extends (shown in pastel red, green, and blue, resp.) each of the original models, in different ways, from the intersection of samples assayed on both data sets to their union. Third, we average all of the different extended models (ignoring missing data) to get a single coherent model of component coefficients across all omics data sets. Assuming that this process has been applied to learn the model from a training data set, we can evaluate the final Cox regression model on both the training set and a test set of patient samples.

All computations were performed in R version 4.2.1 (2022-06-23 ucrt) of the R Statistical Software Environment (R Core Team 2022). Cox proportioanl hazards models for survival analysis were fit using version 3.3.1 of e `survival` R package. We used additional exploratory graphical tools from version 1.3.1 of the `beanplot` R package (Kampstra 2008) and version 1.5.1 of the `Polychrome` R package (Coombes, Brock, Abrams, and Abruzzo 2019).

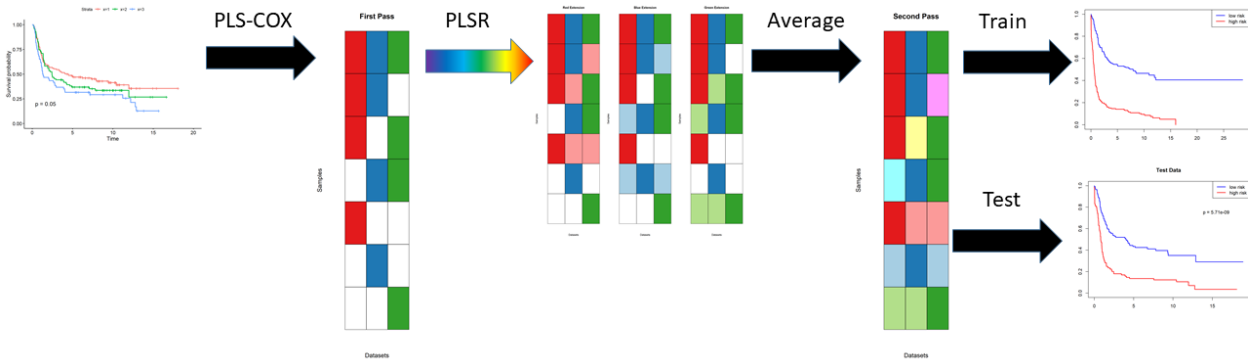

Figure 1: Workflow schematic for plasma algorithm with three omics data sets. See main text for an explanation.

Terminology

Because of the layered nature of the plasma algorithm, we intend to use the following terminology to help clarify the later discussions.

1. The input data contains a list of *omics data sets*.
2. Each omics data set contains measurements of multiple *features*.
3. The first step in the algorithm uses PLS Cox regression to find a set of *components*. Each component is a linear combination of features. The components are used as predictors in a Cox proportional hazards model, which predicts the log hazard ratio as a linear combination of components.
4. The second step in the algorithm creates a secondary layer of components. We do not give these components a separate name. They are not an item of particular focus; we view them as a way to extend the first level components to more samples by "re-interpreting" them in other omics data sets.

Preparing the Data

To be consistent with the `MOFA2` R package (Argelaguet et al. 2020), all of the data sets are arranged so that patient samples are columns and assay features are rows. Our first task is to pad each data set with appropriate `NA`'s to ensure that each set includes the same patient samples in the same order, where that order matches the outcome data frame.

```
MO <- prepareMultiOmics(imputed, Outcome)
summary(MO)
```

```
## Datasets:
##      ClinicalBin ClinicalCont MAF Meth450 miRSeq mRNASeq RPPA
## [1,]          61            2 897    1940    872    2290  223
## [2,]         498          498 498     498    498     498  498
## Outcomes:
```

```
##    patient_id  vital_status days_to_death    days_to_last_followup      Days
##  1000   :  1   alive:283    Min.   :   1.0   Min.   :   0          Min.   :   0.0
##  1002   :  1   dead :215    1st Qu.: 280.0   1st Qu.: 394          1st Qu.: 325.2
##  1005   :  1                Median : 550.0   Median : 757          Median : 666.5
##  1011   :  1                Mean   : 872.3   Mean   :1049          Mean   : 972.6
##  1012   :  1                3rd Qu.:1110.5   3rd Qu.:1374          3rd Qu.:1259.8
##  1016   :  1                Max.   :5287.0   Max.   :4765          Max.   :5287.0
##  (Other):492                NA's   :283      NA's   :215
```

We see that the number of patients in each data set is now equal to the number of patients with clinical outcome data.

Split Into Training and Test

As indicated above, we want to separate the data set into training and test samples. We will use 60% for training and 40% for testing.

```
set.seed(54321)
splitVec <- rbinom(nrow(Outcome), 1, 0.6)
```

Figure 2 presents a graphical overview of the number of samples (`N`) and the number of features (`D`) in each omics component of the training and test sets.

```
trainD <- MO[, splitVec == 1]
testD <- MO[, splitVec == 0]
opar <- par(mai = c(1.02, 1.32, 0.82, 0.22), mfrow = c(1,2))
plot(trainD, main = "Train")
plot(testD, main = "Test")

par(opar)
```

Results

Individual PLS Cox Regression Models

The first step of the `plasma` algorithm is to fit PLS Cox models on each omics data set using the function `fitCoxModels`. The returned object of class `MultiplePLSCoxModels` contains a list of `SingleModel` objects, one for each assay, and within each there are three regression models:

- The `plsRcoxmodel` contains the coefficients of the components learned by PLS Cox regression. The number of components is determined automatically as a function of the logarithm of the number of features in the omics data set. The output of this model is a continuous prediction of "risk" for the time-to-event outcome of interest.
- Two separate models are constructed using the prediction of risk on the training data.
  - The `riskModel` is a `coxph` model using continuous predicted risk as a single predictor.
  - The `splitModel` is a `coxph` model using a binary split of the risk (at the median) as the predictor.

```
cache <- "firstPass.Rda"
if (file.exists(cache)) {
  load(cache)
} else {
  suppressWarnings( firstPass <- fitCoxModels(trainD, timevar = "Days",
                            eventvar = "vital_status", eventvalue = "dead") )
  save(firstPass, file = cache)
}
rm(cache)

summary(firstPass)
```

```
## An object containing MultiplePLSCoxModels based on:
## [1] "ClinicalBin"  "ClinicalCont" "MAF"          "Meth450"      "miRSeq"       "mRNASeq"
## [7] "RPPA"
```

```
if (!interactive()) {
  plot(firstPass, legloc = "topright") # margins too small inside RStudio window
}
```

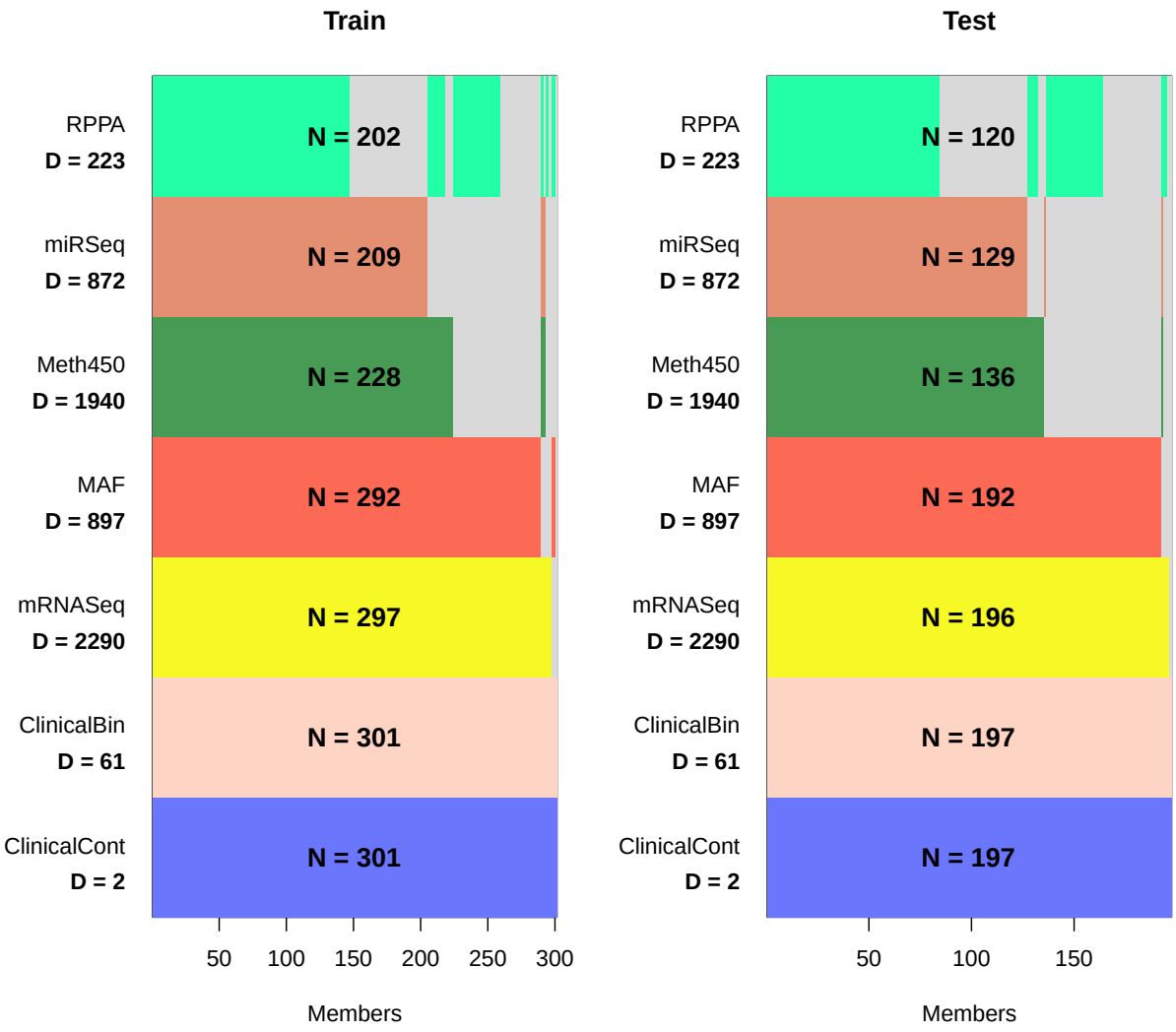

Figure 2: Overview of training and test data.

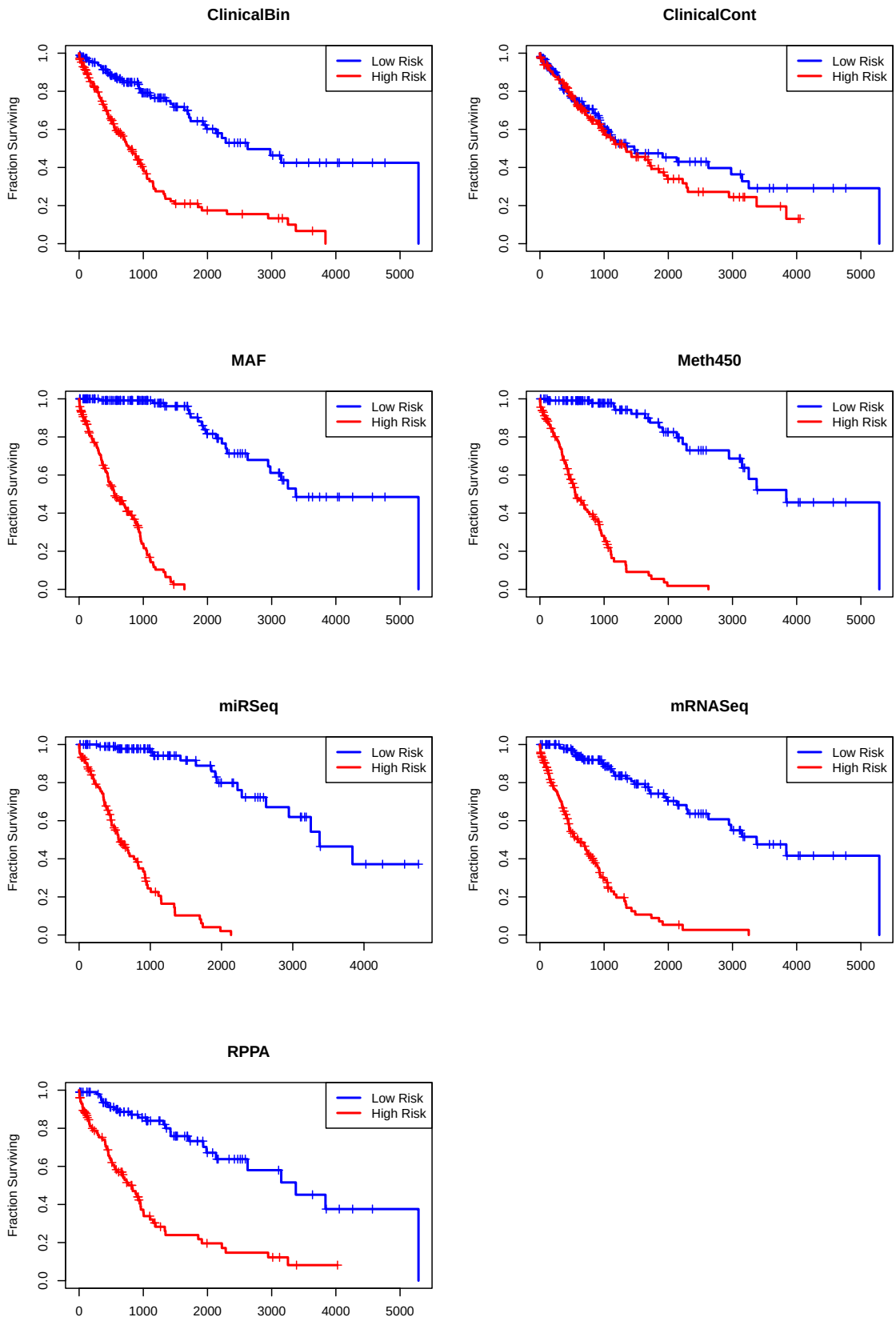

Figure 3: Kaplan-Meier plots of overall survival on the training set from separate PLS Cox omics models

On the training set, each of the seven contributing omics data sets is able to find a PLS model that can successfully separate high risk from low risk patients (**Figure 3**).

Extend Model Components Across Omics Data Sets

The second step of the algorithm is to extend the individual omics-based models across other omics data sets. This step is performed using the `plasma` function, which takes in the previously created objects of class `multiOmics` and `MultiplePLSCoxModels`. The function operates iteratively, so in our case there are seven different sets of predictions of the PLS components. These different predictions are averaged and saved internally as a data frame called `meanPredictions`. The structure of models created and stored in the `plasma` object is the same as for the separate, individual, omics models. **Figure 4** shows the Kaplan-Meier plot using the predicted risk, split at the median value, on the training data set.

```
cache = "pl.Rda"
if (file.exists(cache)) {
  load(cache)
} else {
  pl <- plasma(MO, firstPass)
  save(pl, file = cache)
}
rm(cache)
plot(pl, legloc = "topright", main = "Training Data", xlab = "Time (Days)")
```

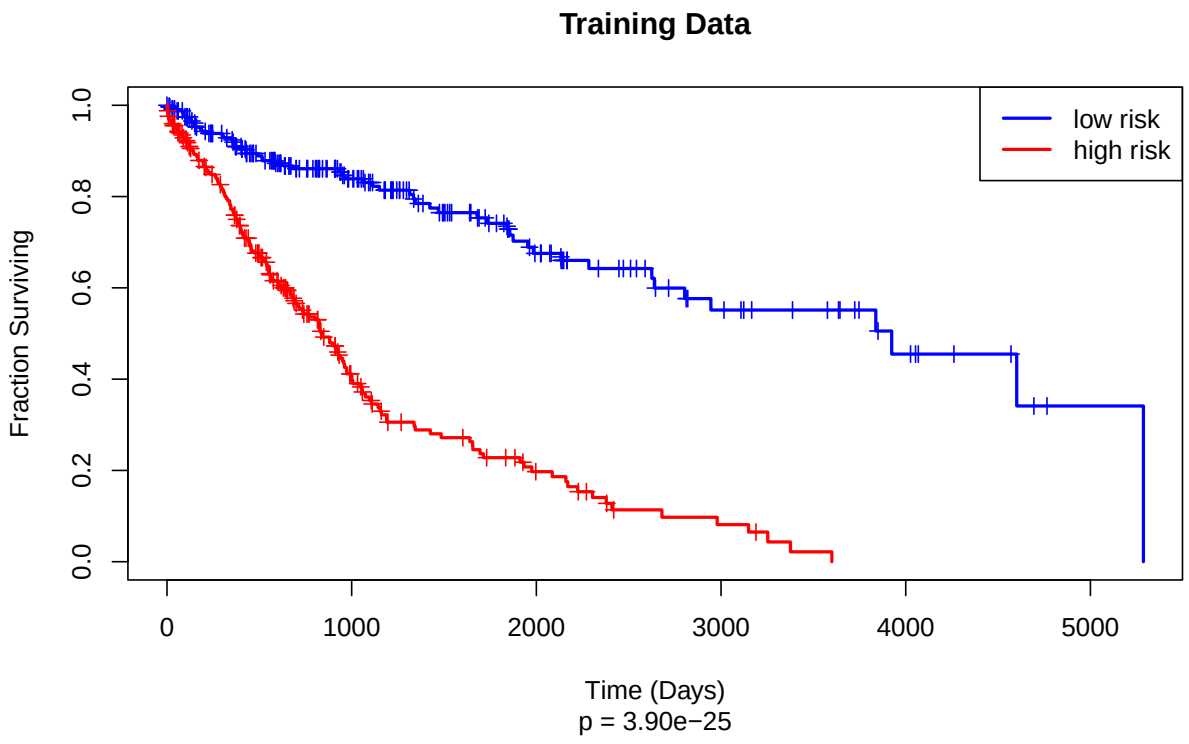

Figure 4: Kaplan-Meier plot of overall survival on the training set using the unified `plasma` Cox model.

Independent Test Set

Now we want to see how well the final composite model generalizes to our test set. **Figure 5** uses the predicted risk, split at the median of the training data, to construct a Kaplan-Meier plot on the test data.

The model yields a statistically significant (p = 0.0063) separation of outcomes between the high and low risk patients.

```
testpred <- predict(pl, testD)
plot(testpred, main="Testing Data", xlab = "Time (Days)")
```

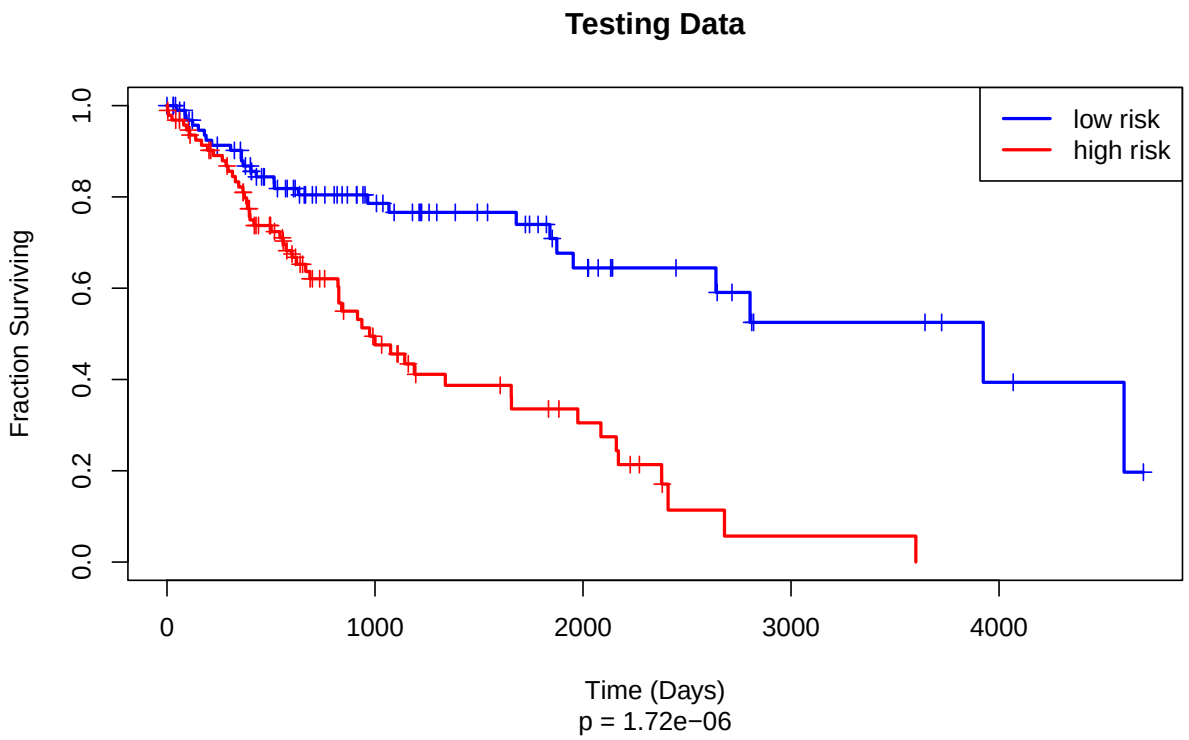

Figure 5: Kaplan-Meier plot of overall survival on the test set.

Interpretation

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") ~ ClinicalBin2 +
##     ClinicalBin3 + MAF1 + MAF2 + MAF3 + MAF4 + Meth4501 + Meth4502 +
##     Meth4503 + Meth4504 + miRSeq1 + miRSeq4 + mRNASeq1 + RPPA2,
##     data = riskDF)
##
##                   coef exp(coef)  se(coef)      z        p
```

```
## ClinicalBin2  1.943430  6.982659  0.853921  2.276 0.022853
## ClinicalBin3  1.646241  5.187445  0.622666  2.644 0.008197
## MAF1          0.993317  2.700175  0.146533  6.779 1.21e-11
## MAF2          0.462372  1.587836  0.253874  1.821 0.068566
## MAF3          1.362046  3.904174  0.480869  2.832 0.004619
## MAF4         -5.082248  0.006206  1.536750 -3.307 0.000943
## Meth4501      0.126886  1.135287  0.053448  2.374 0.017597
## Meth4502      0.415770  1.515537  0.129797  3.203 0.001359
## Meth4503      1.381189  3.979632  0.314414  4.393 1.12e-05
## Meth4504      0.840972  2.318620  0.209150  4.021 5.80e-05
## miRSeq1       1.816057  6.147571  0.574857  3.159 0.001582
## miRSeq4       0.609111  1.838796  0.366624  1.661 0.096632
## mRNASeq1     -0.093122  0.911083  0.046610 -1.998 0.045729
## RPPA2         0.307385  1.359864  0.165605  1.856 0.063434
##
## Likelihood ratio test=174.4  on 14 df, p=< 2.2e-16
## n= 498, number of events= 215
```

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

```
##  [1] "ClinicalBin2" "ClinicalBin3" "MAF1"         "MAF2"         "MAF3"         "MAF4"
##  [7] "Meth4501"     "Meth4502"     "Meth4503"     "Meth4504"     "miRSeq1"      "miRSeq4"
## [13] "mRNASeq1"     "RPPA2"
```

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 ws retained in the final model.

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.

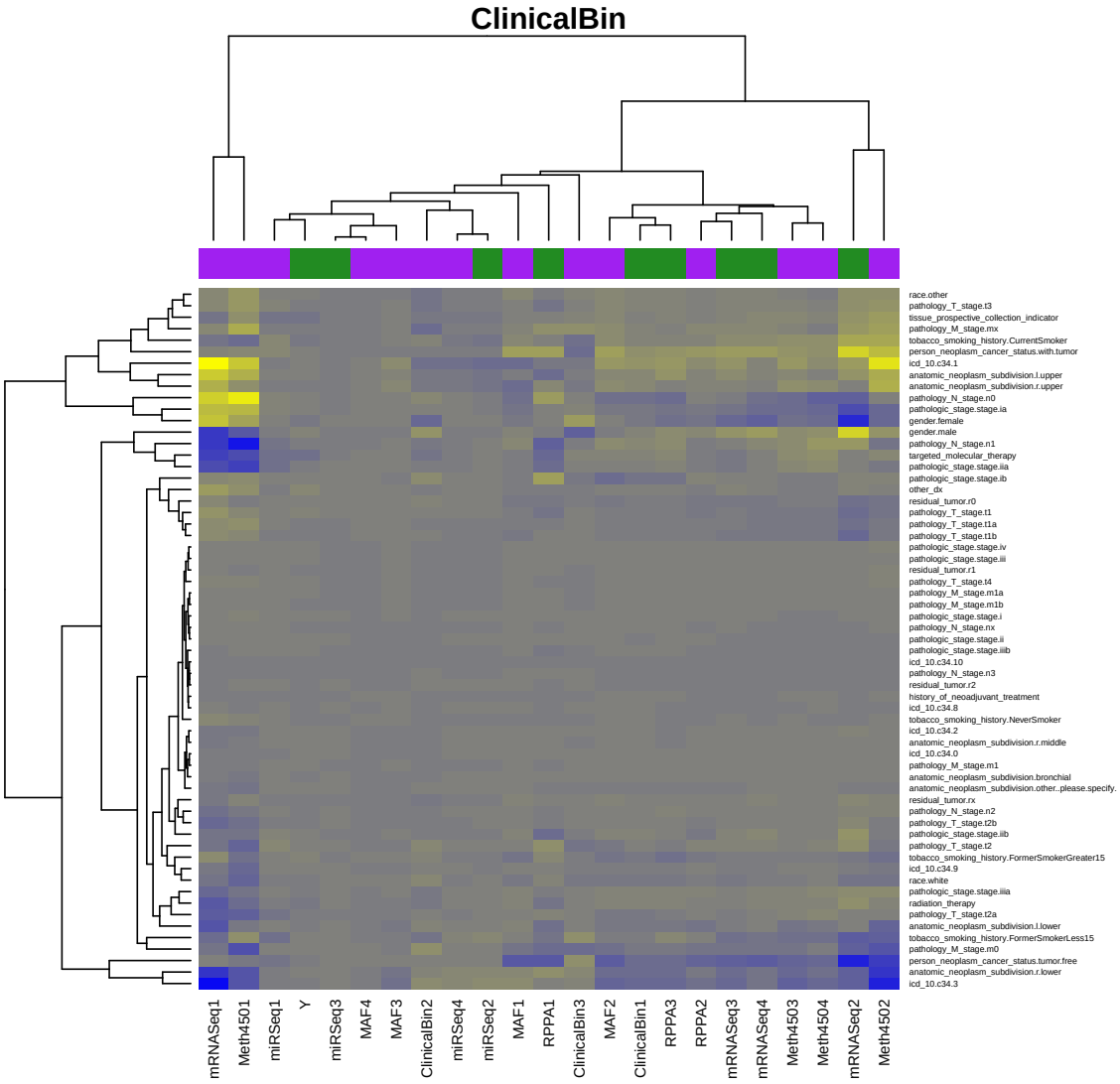

Figure 6: Unscaled heatmap of the contributions of binary clinical features to all components (purple = retained in final model, green = not retained).

```
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)
```

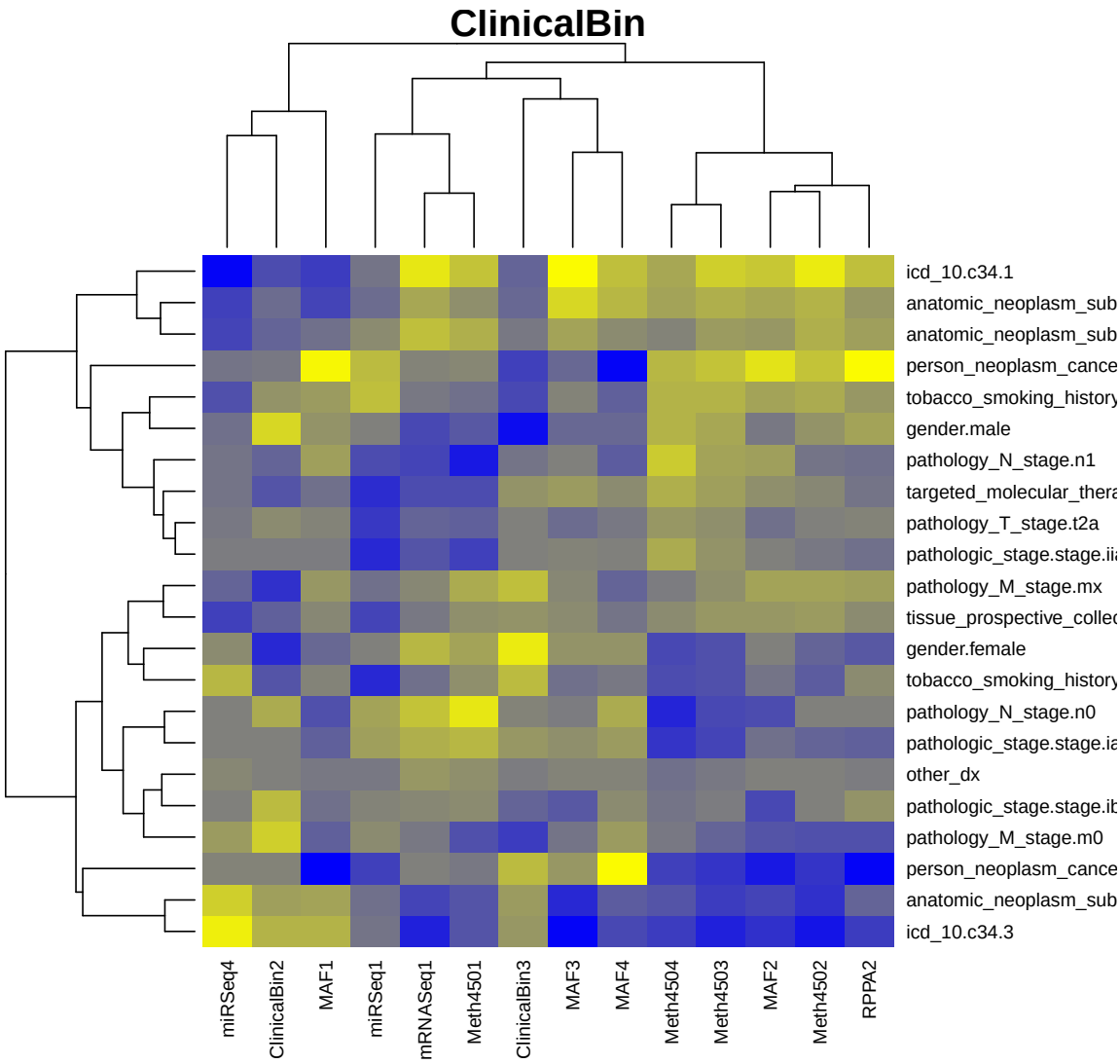

Figure 7: Scaled heatmap of the contributions of filtered binary clinical features to important components.

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

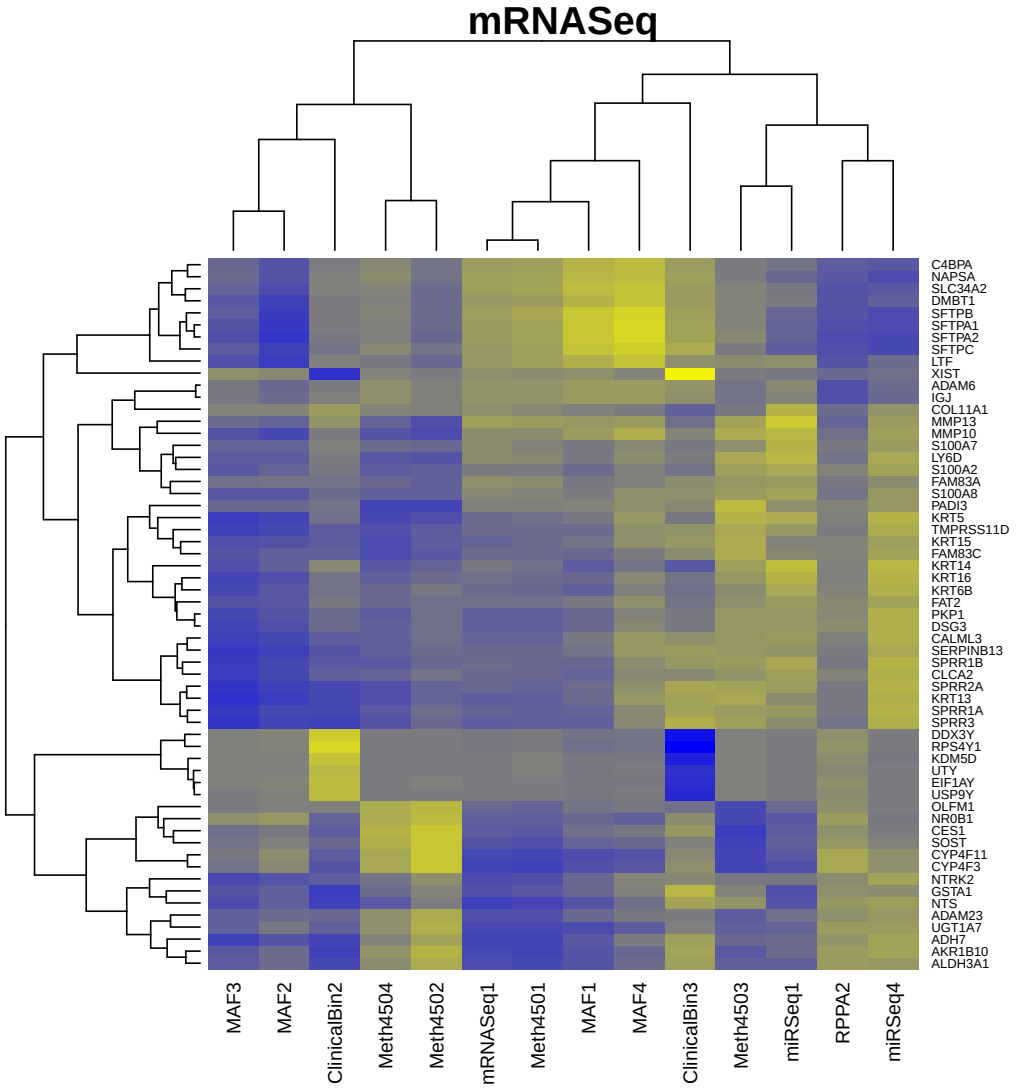

Figure 8: Scaled heatmap of the contributions of filtered mRNA features to important components

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.

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

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)
```

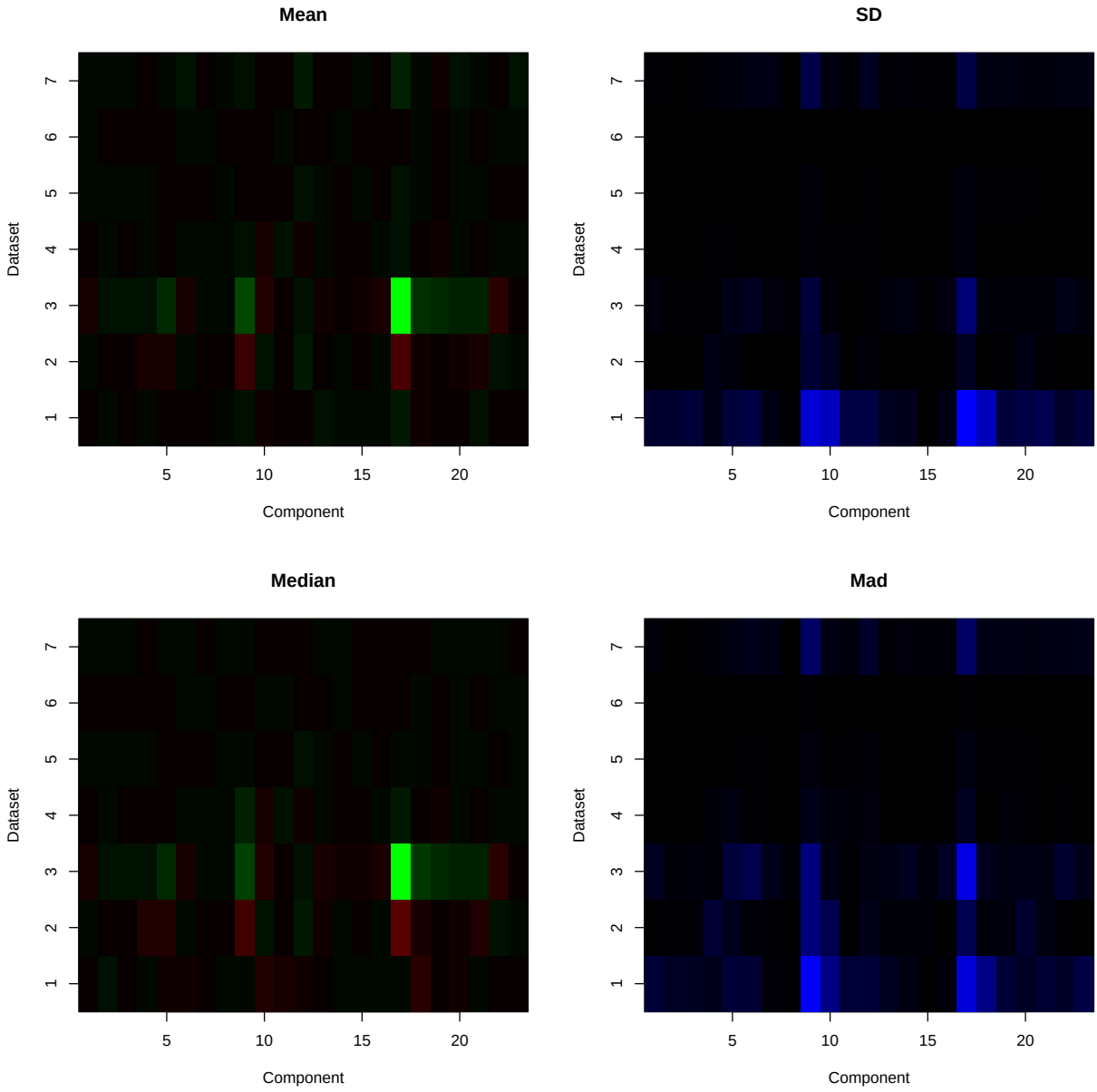

Figure 9: Summary statistics of weights by component and dataset.

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(5,3), 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(5, 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))
  which(A > S[21])
```

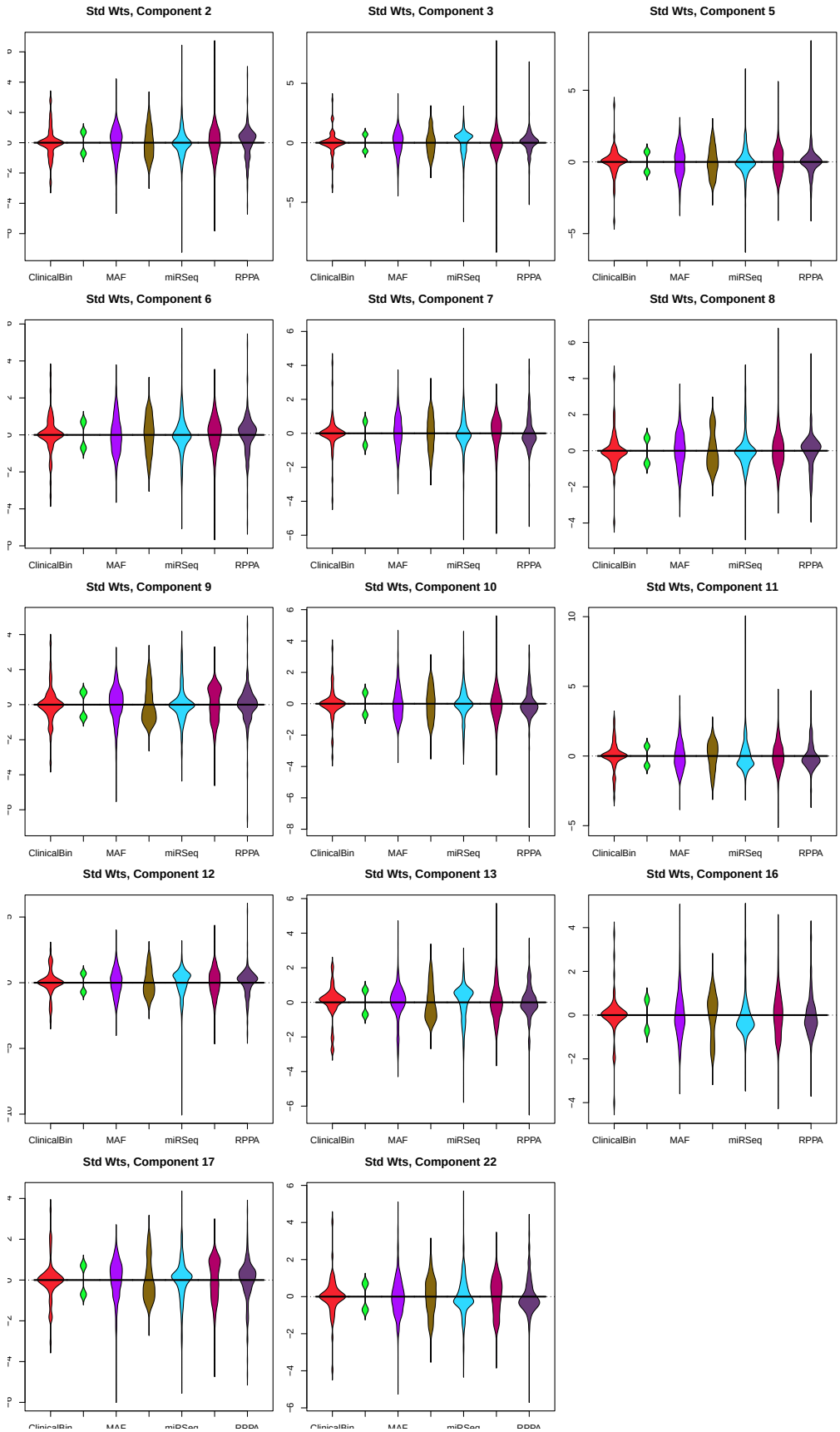

Figure 10: Distributions of standardized weights by data set and component.

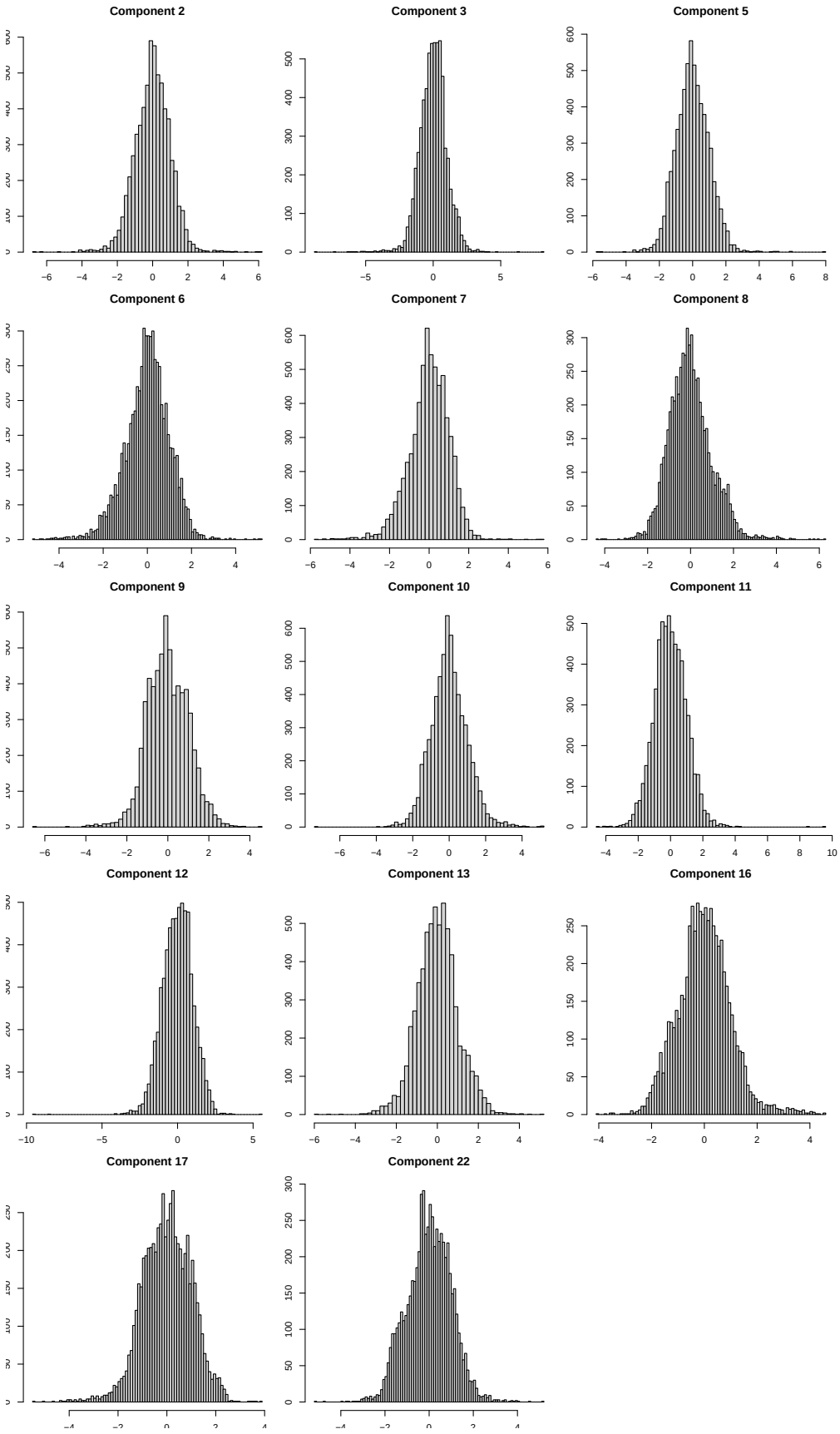

Figure 11: Histogram of standardized weights by component.

```
})
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)
```

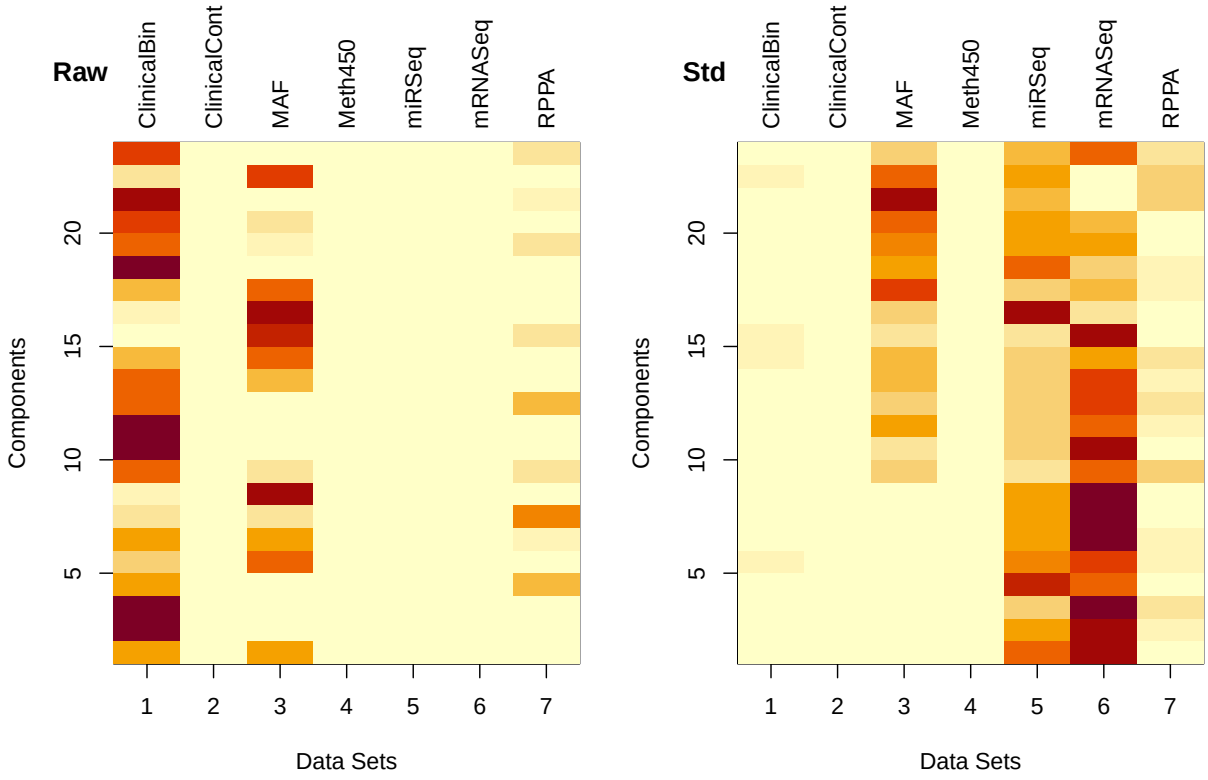

Figure 12: Number of contributers to top twenty lists.

```
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)
```

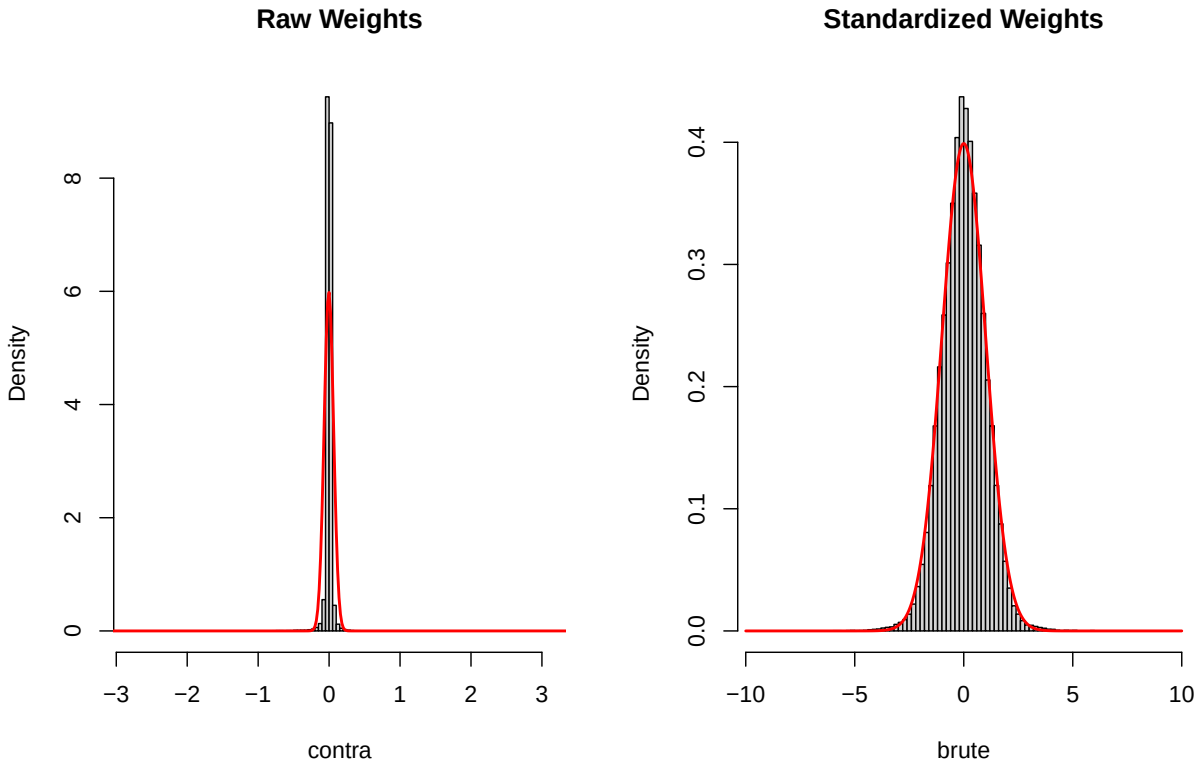

Figure 13: Histograms of all weights (combined).

```
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]
```

```
L <- length(mainterms)
opar <- par(mai = c(1.02, 0.82,1.22, 0.22))
image(1:7, 1:L, top1p, ylab = "Components", xlab = "Data Sets")
mtext(levels(datasrc), side = 3, at = 1:7, line = 1, las=2)
```

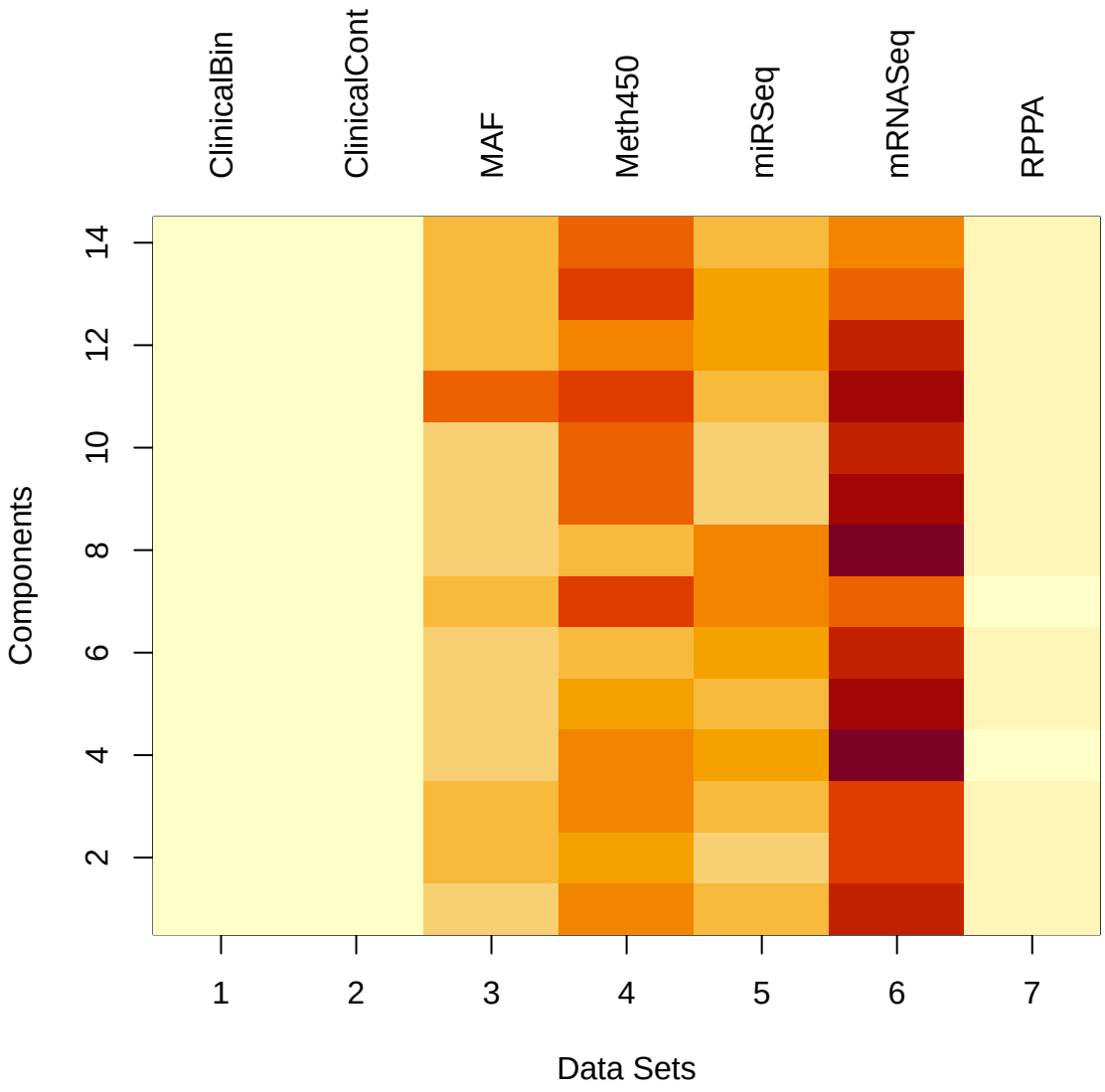

Figure 14: Number of significant contributions by data set and component.

```
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("ClinicalBin2")
summary(pickA)
```

```
##    Feature                   Source       Weight          Nickname
##  Length:267         ClinicalBin : 5   Min.   :-6.6680   Length:267
##  Class :character   ClinicalCont: 0   1st Qu.:-2.4424   Class :character
##  Mode  :character   MAF         :41   Median :-2.0293   Mode  :character
##                     Meth450     :67   Mean   :-0.3708
##                     miRSeq      :45   3rd Qu.: 2.2343
##                     mRNASeq     :96   Max.   : 6.1734
##                     RPPA        :13
```

We start by looking more closely at the clinical features.

```
pickA[1:5, ]
```

```
##                                                Feature      Source    Weight Nickname
## gender.female                            gender.female ClinicalBin -2.757341
## gender.male                                gender.male ClinicalBin  2.872117
## pathologic_stage.stage.ib pathologic_stage.stage.ib ClinicalBin  1.965747
## pathology_M_stage.m0           pathology_M_stage.m0 ClinicalBin  2.696807
## pathology_M_stage.mx           pathology_M_stage.mx ClinicalBin -2.510628
```

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
## ADAMTS16 ADAMTS16    MAF -2.241170 ADAMTS16
## AKAP13     AKAP13    MAF  2.155648   AKAP13
## CDH18       CDH18    MAF -2.472199    CDH18
## CNTNAP5   CNTNAP5    MAF -2.378060  CNTNAP5
## COL11A1   COL11A1    MAF -2.171134  COL11A1
```

```
## COL6A3     COL6A3    MAF -1.978799   COL6A3
## CSMD2       CSMD2    MAF -2.750257    CSMD2
## CSMD3       CSMD3    MAF -3.300840    CSMD3
## CUBN         CUBN    MAF -2.622299     CUBN
## DCHS2       DCHS2    MAF -3.417828    DCHS2
## DGKI         DGKI    MAF -2.533558     DGKI
## DNAH10     DNAH10    MAF -2.791308   DNAH10
## FAM135B   FAM135B    MAF -2.231716  FAM135B
## FAT2         FAT2    MAF  2.022856     FAT2
## FAT3         FAT3    MAF -2.353307     FAT3
## FBN2         FBN2    MAF -2.747776     FBN2
## FLG           FLG    MAF -3.338727      FLG
## FSCB         FSCB    MAF -2.093879     FSCB
## GRIN2B     GRIN2B    MAF -2.004967   GRIN2B
## HCN1         HCN1    MAF -3.638311     HCN1
## KEAP1       KEAP1    MAF -2.012947    KEAP1
## LRP1         LRP1    MAF  2.250414     LRP1
## LRP1B       LRP1B    MAF -3.841406    LRP1B
## MUC16       MUC16    MAF -3.348977    MUC16
## NAV3         NAV3    MAF -2.532153     NAV3
## NFE2L2     NFE2L2    MAF  3.652172   NFE2L2
## NLRP12     NLRP12    MAF -1.963323   NLRP12
## NLRP14     NLRP14    MAF -2.499965   NLRP14
## OR8G5       OR8G5    MAF -2.013133    OR8G5
## PCDH15     PCDH15    MAF -4.109602   PCDH15
## PIK3CA     PIK3CA    MAF  2.481328   PIK3CA
## PRDM9       PRDM9    MAF -2.314944    PRDM9
## RYR1         RYR1    MAF -1.996072     RYR1
## SCN7A       SCN7A    MAF -2.116237    SCN7A
## SI             SI    MAF -3.423093       SI
## SPHKAP     SPHKAP    MAF -2.823433   SPHKAP
## TENM3       TENM3    MAF -2.408292    TENM3
## TNR           TNR    MAF -2.181196      TNR
## USH2A       USH2A    MAF -2.811211    USH2A
## UTRN         UTRN    MAF -2.721005     UTRN
## ZFHX4       ZFHX4    MAF -3.458028    ZFHX4
```

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, Bardes, Aronow, and Jegga 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

Adossa N, Khan S, Rytkonen KT, Elo LL (2021). "Computational Strategies for Single-Cell Multi-Omics Integration." *Comput Struct Biotechnol J*, **19**, 2588–2596. Journal Article. https://doi.org/10.1016/j.csbj.2021.04.060.

Argelaguet R, Arnol D, Bredikhin D, Deloro Y, Velten B, Marioni JC, Stegle O (2020). "MOFA+: A Statistical Framework for Comprehensive Integration of Multi-Modal Single-Cell Data." *Genome Biol*, **21**(1), 111. Journal Article. https://doi.org/10.1186/s13059-020-02015-1.

Argelaguet R, Velten B, Arnol D, Dietrich S, Zenz T, Marioni JC, Buettner F, Huber W, Stegle O (2018). "Multi-Omics Factor Analysis-a Framework for Unsupervised Integration of Multi-Omics Data Sets." *Mol Syst Biol*, **14**(6), e8124. Journal Article. https://doi.org/10.15252/msb.20178124.

Bastien P, Bertrand F, Meyer N, Maumy-Bertrand M (2015). "Deviance Residuals-Based Sparse PLS and Sparse Kernel PLS Regression for Censored Data." *Bioinformatics*, **31**(3), 397–404. Journal Article. https://doi.org/10.1093/bioinformatics/btu660.

Bertrand F, Maumy-Bertrand M (2021). "Fitting and Cross-Validating Cox Models to Censored Big Data with Missing Values Using Extensions of Partial Least Squares Regression Models." *Front Big Data*, **4**, 684794. Journal Article. https://doi.org/10.3389/fdata.2021.684794.

Cancer Genome Atlas Research Network (2017). "Integrated Genomic Characterization of Oesophageal Carcinoma." *Nature*, **541**(7636), 169–175. Journal Article. https://doi.org/10.1038/nature20805.

Chen J, Bardes EE, Aronow BJ, Jegga AG (2009). "ToppGene Suite for Gene List Enrichment Analysis and Candidate Gene Prioritization." *Nucleic Acids Res*, **37**(Web Server issue), W305–11. Journal Article. https://doi.org/10.1093/nar/gkp427.

Coombes KR, Brock G, Abrams ZB, Abruzzo LV (2019). "Polychrome: Creating and Assessing Qualitative Palettes with Many Colors." *Journal of Statistical Software*, **90**(1), 1–23. Journal Article. https://doi.org/10.18637/jss.v090.c01.

Graw S, Chappell K, Washam CL, Gies A, Bird J, Robeson 2nd M. S., Byrum SD (2021). "Multi-Omics Data Integration Considerations and Study Design for Biological Systems and Disease." *Mol Omics*, **17**(2), 170–185. Journal Article. https://doi.org/10.1039/d0mo00041h.

Heo YJ, Hwa C, Lee GH, Park JM, An JY (2021). "Integrative Multi-Omics Approaches in Cancer Research: From Biological Networks to Clinical Subtypes." *Mol Cells*, **44**(7), 433–443. Journal Article. https://doi.org/10.14348/molcells.2021.0042.

Jensen MA, Ferretti V, Grossman RL, Staudt LM (2017). "The NCI Genomic Data Commons as an Engine for Precision Medicine." *Blood*, **130**(4), 453–459. Journal Article. https://doi.org/10.1182/blood-2017-03-735654.

Kampstra P (2008). "Beanplot: A Boxplot Alternative for Visual Comparison of Distributions." *Journal of Statistical Software*, **28**(1), 1–9. Journal Article. Retrieved from https://doi.org/10.18637/jss.v028.c01

Mishra P, Liland KH (2022). "Swiss Knife Partial Least Squares (SKPLS): One Tool for Modelling Single Block, Multiblock, Multiway, Multiway Multiblock Including Multi-Responses and Meta Information Under the ROSA Framework." *Anal Chim Acta*, **1206**, 339786. Journal Article. https://doi.org/10.1016/j.aca.2022.339786.

Picard M, Scott-Boyer MP, Bodein A, Perin O, Droit A (2021). "Integration Strategies of Multi-Omics Data for Machine Learning Analysis." *Comput Struct Biotechnol J*, **19**, 3735–3746. Journal Article. https://doi.org/10.1016/j.csbj.2021.06.030.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. Book, R Foundation for Statistical Computing, Vienna, Austria. Retrieved from https://www.R-project.org/

Reel PS, Reel S, Pearson E, Trucco E, Jefferson E (2021). "Using Machine Learning Approaches for Multi-Omics Data Analysis: A Review." *Biotechnol Adv*, **49**, 107739. Journal Article. https://doi.org/10.1016/j.biotechadv.2021.107739.

Simon RM, Dobbin K (2003). "Experimental Design of DNA Microarray Experiments." *Biotechniques*, **Suppl**, 16–21. Journal Article. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/12664680

Subramanian I, Verma S, Kumar S, Jere A, Anamika K (2020). "Multi-Omics Data Integration, Interpretation, and Its Application." *Bioinform Biol Insights*, **14**, 1177932219899051. Journal Article. https://doi.org/10.1177/1177932219899051.

Vlachavas EI, Bohn J, Uckert F, Nurnberg S (2021). "A Detailed Catalogue of Multi-Omics Methodologies for Identification of Putative Biomarkers and Causal Molecular Networks in Translational Cancer Research." *Int J Mol Sci*, **22**(6). Journal Article. https://doi.org/10.3390/ijms22062822.