Aarhus Multiplex Analysis

Benjamin Draves

Data Overview

The Aarhus multiplex dataset is a multiplex network over $n = 61$ vertices corresponding to professors, postdocs, PhD students, and adminstrative staff from Aarhus University's Computer Science department. There are $m = 5$ layers in the network each representing a different association; working relationships, repeated lesisurely activity, regularly eating lunch together, co-authorship, and Facebook friendship.

142 employees were sent a roster of all employees in the department and were asked to identify other members in the department with whom they worked regularly, ate lunch, and engaged in repeated leisurely activity. The edges were regarded as nondirectional meaning if employee i identified employee j as an associate in any of these activities, a non-directional edge was assigned between i and j in that layer. $n = 61$ employees participated in the study (43% participation). Every participant completed the full survey.

The Facebook friendship and co-authorship layers of these 61 participants were identified through a custom application and the DBLP bibligography database. 77% of the participants had a Facebook account and at least one paper in which two employees were co-authors resulted in an edge in the co-authorship layer.

Data Preperation

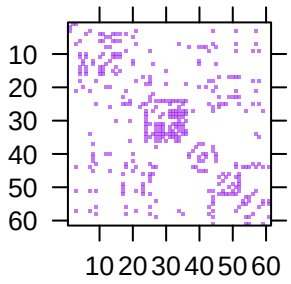
To correct for bias due to graph-specific sparsity, we normalize each adjacency matrix so that they share a common Frobenius norm. That is we analyze the normalized adjacencies $\mathbf{A}^{(g)}_{\text{norm}} = \mathbf{A}^{(g)}/\|\mathbf{A}^{(g)}\|_F$.

```
# fetch data
devtools::install_github("achab94/mplex")
library(mplex)
data(aarhus_mplex)

# format as Adjacency Matrices
n <- nrow(aarhus_mplex$nodes)
m <- length(unique(aarhus_mplex$layerNames))
A_list <- lapply(1:m, function(x) elist_to_adj(as.matrix(aarhus_mplex[[x +
    2]][1:2])))

# Normalize adjacency matrices
A_list_norm <- lapply(A_list, function(x) x/norm(x,
    type = "F"))

# Capitalize Layer Names
layer_names <- as.vector(sapply(aarhus_mplex$layerNames,
    function(x) paste0(toupper(substring(x, 1,
        1)), substring(x, 2))))
```

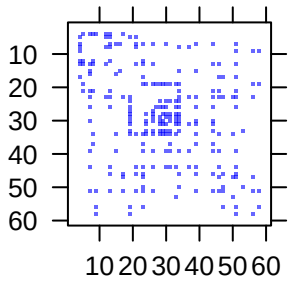
Exploratory Data Analysis

```
colors <- c("purple", "blue", "red", "green",
    "black")
```

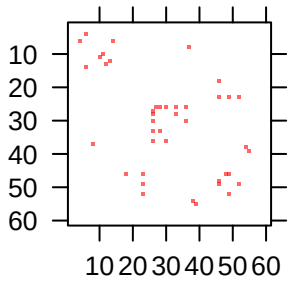
```
# plot adjacencies in a 3 x 2 Layout
p1 <- image(A_list[[1]], xlab = "", ylab = "",
    sub = layer_names[1], border.col = NA, col.regions = adjustcolor(colors[1],
        alpha.f = 0.6), axes = FALSE)
p2 <- image(A_list[[2]], xlab = "", ylab = "",
    sub = layer_names[2], border.col = NA, col.regions = adjustcolor(colors[2],
        alpha.f = 0.6))
p3 <- image(A_list[[3]], xlab = "", ylab = "",
    sub = layer_names[3], border.col = NA, col.regions = adjustcolor(colors[3],
        alpha.f = 0.6))
p4 <- image(A_list[[4]], xlab = "", ylab = "",
    sub = layer_names[4], border.col = NA, col.regions = adjustcolor(colors[4],
        alpha.f = 0.6))
p5 <- image(A_list[[5]], xlab = "", ylab = "",
    sub = layer_names[5], border.col = NA, col.regions = adjustcolor(colors[5],
        alpha.f = 0.6))

layout <- rbind(c(1, 1, 2, 2, 3, 3), c(NA, 4,
    4, 5, 5, NA))
grid.arrange(p1, p2, p3, p4, p5, layout_matrix = layout)
```



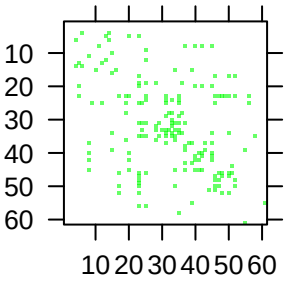
Lunch



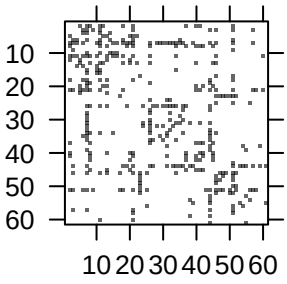
Facebook



Coauthor



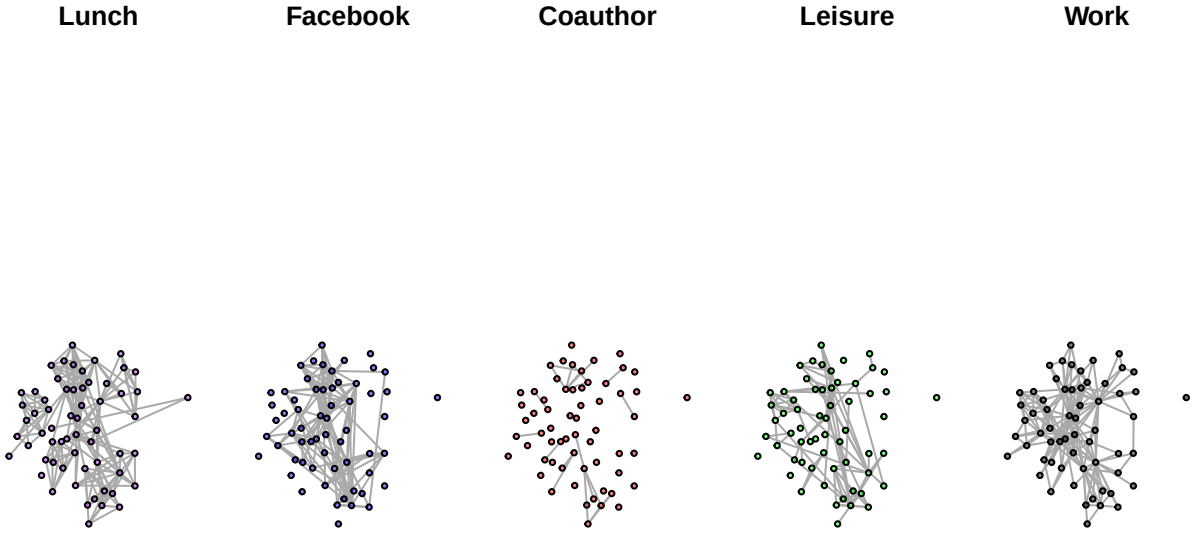
Leisure



Work

```
# Create igraph objects
g1 <- graph_from_adjacency_matrix(A_list[[1]],
    mode = "undirected")
g2 <- graph_from_adjacency_matrix(A_list[[2]],
    mode = "undirected")
g3 <- graph_from_adjacency_matrix(A_list[[3]],
    mode = "undirected")
g4 <- graph_from_adjacency_matrix(A_list[[4]],
```

```
    mode = "undirected")
g5 <- graph_from_adjacency_matrix(A_list[[5]],
    mode = "undirected")

# Plot graphs
layout(rbind(c(1, 1, 2, 2, 3, 3), c(0, 4, 4, 5,
    5, 0)))
layout(matrix(1:5, nrow = 1))
par(mar = rep(1, 4))

lay <- layout.fruchterman.reingold(g5)
plot(g1, vertex.label = NA, vertex.size = 6, layout = lay,
    vertex.color = adjustcolor(colors[1], alpha.f = 0.5),
    main = layer_names[1])
plot(g2, vertex.label = NA, vertex.size = 6, layout = lay,
    vertex.color = adjustcolor(colors[2], alpha.f = 0.5),
    main = layer_names[2])
plot(g3, vertex.label = NA, vertex.size = 6, layout = lay,
    vertex.color = adjustcolor(colors[3], alpha.f = 0.5),
    main = layer_names[3])
plot(g4, vertex.label = NA, vertex.size = 6, layout = lay,
    vertex.color = adjustcolor(colors[4], alpha.f = 0.5),
    main = layer_names[4])
plot(g5, vertex.label = NA, vertex.size = 6, layout = lay,
    vertex.color = adjustcolor(colors[5], alpha.f = 0.5),
    main = layer_names[5])
```

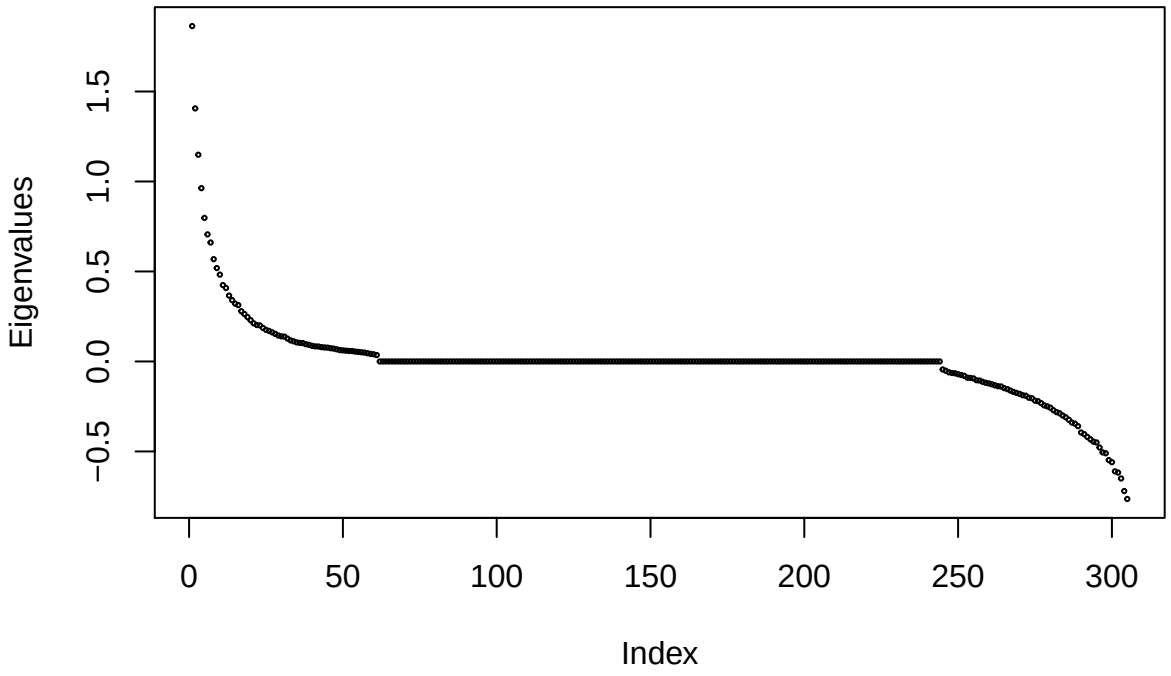


Omnibus Data Analysis

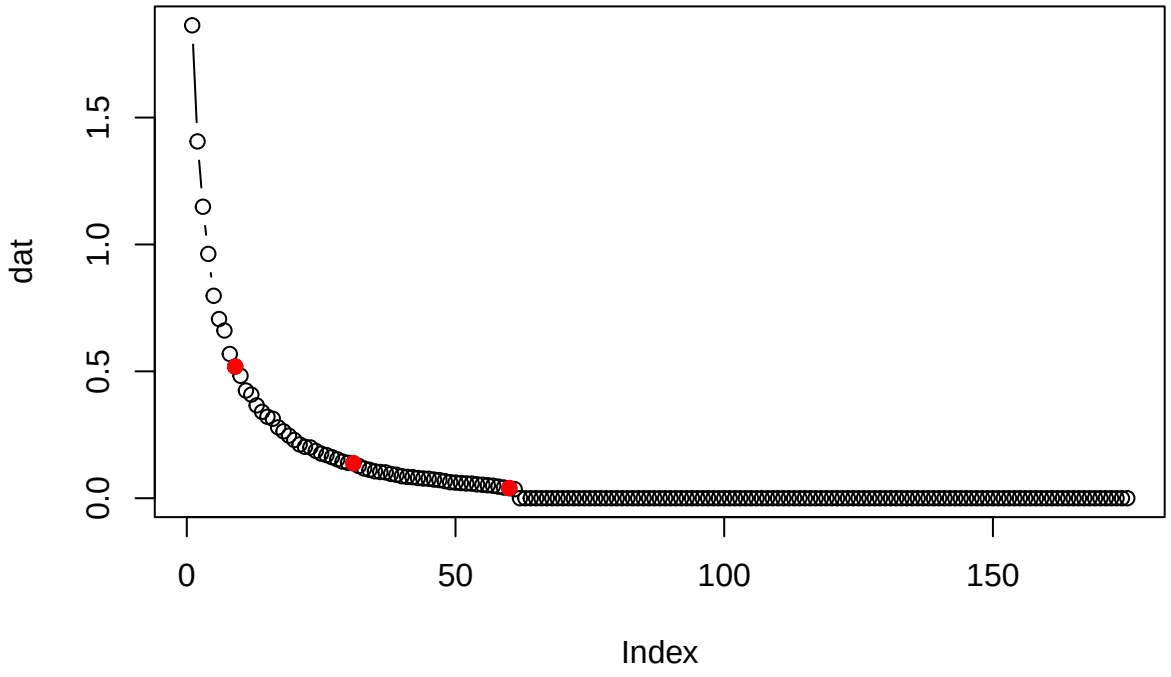
Determining Embedding Dimension

```
# make Omnibus matrix
Atil <- make_omni(A_list_norm)

# skree plot
```

```r
evals <- eigen(Atil)$values
plot(evals, ylab = "Eigenvalues", cex = 0.3)
```




```r
# determine d - get elbows
non_zero_evals <- evals[which(evals > 0)]
elbows <- getElbows(non_zero_evals, n = 3)
```




```r
# set d
d <- elbows[1]
```

Goodness of Fit

To determine if the ESRDPG provides a good fit to the Aarhus multiplex network, we consider the matrix

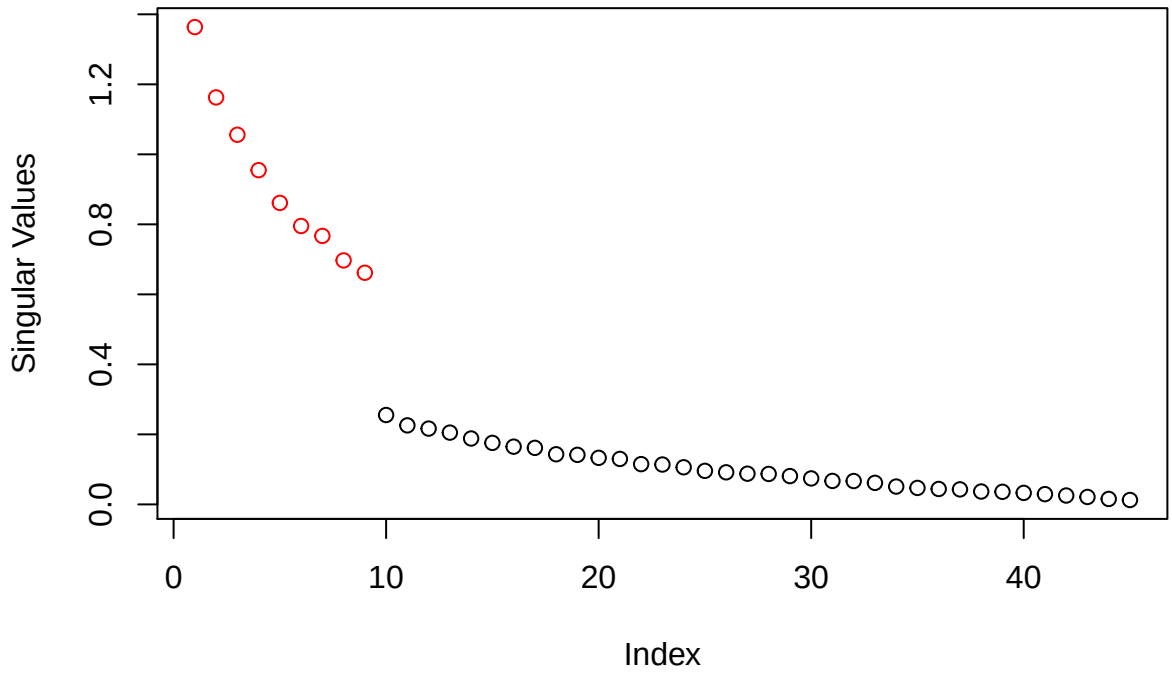
$$\hat{\mathbf{H}} = [\hat{\mathbf{X}}^{(1)}_{\text{Omni}} \hat{\mathbf{X}}^{(2)}_{\text{Omni}} \dots \hat{\mathbf{X}}^{(m)}_{\text{Omni}}] \in \mathbb{R}^{n \times md}.$$

Under the ESRDPG, $\mathbf{H} = \mathbb{E}[\hat{\mathbf{H}}] = \mathbf{X}[\mathbf{S}^{(1)}\mathbf{S}^{(2)} \dots \mathbf{S}^{(m)}]$ and

$$\text{rank}(\mathbf{H}) \leq \min \left\{ \text{rank}(\mathbf{X}), \text{rank}([\mathbf{S}^{(1)}\mathbf{S}^{(2)} \dots \mathbf{S}^{(m)}]) \right\} \leq d$$

Therefore, to determine if we have properly identified the model dimension, we can analyze the gap $\sigma_d(\hat{\mathbf{H}}) - \sigma_{d+1}(\hat{\mathbf{H}})$.

```
# Embedd Atil
Lhat <- ase(Atil, d)

# get goodness of fit matrix
H <- matrix(NA, nrow = n, ncol = m * d)
for (i in 1:m) {
    H[, (d * (i - 1) + 1):(d * i)] <- Lhat[(n *
        (i - 1) + 1):(n * i), ]
}

# Is M rank d?
plot(svd(H)$d, col = c(rep("red", d), rep("black",
    d * (m - 1))), ylab = "Singular Values")
```

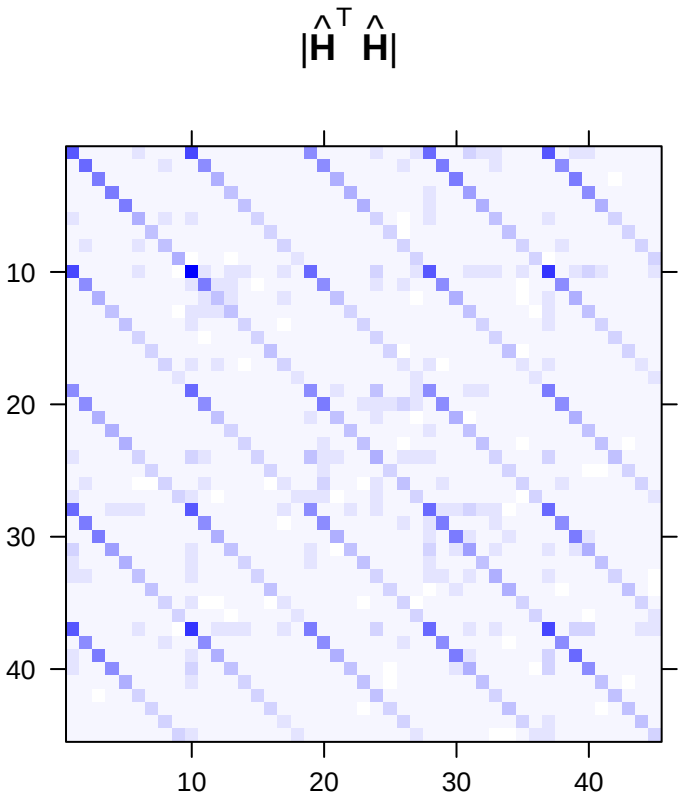


While this analysis provides evidence to suggest that the model is of the correct dimension, it does suggest that the scaling matrices are diagnol. Under the ESRDPG, in the limit $\mathbf{X}^T\mathbf{X}$ is diagnol and each scaling matrix $\{\mathbf{S}^{(g)}\}_{g=1}^m$ is diagonal. Therefore, the matrix

$$\mathbf{H}^T\mathbf{H} = [\mathbf{S}^{(i)}\mathbf{X}^T\mathbf{X}\mathbf{S}^{(j)}]_{i,j=1}^m$$

should be diagnol in each $d \times d$ block. To that end we visualize $|\mathbf{H}^T\mathbf{H}|$.

```
# Diagonal Scaling?
image(Matrix(crossprod(H)), sub = "", xlab = "",
    ylab = "", useAbs = TRUE, col.regions = colorRampPalette(c("white",
        "blue"))(30), main = expression(paste("|",
        hat(bold(H))^T ~ hat(bold(H)), "|")),
    border.col = NA)
```



Moreover, we look to analyze the element-wise mean matrix of the combination of these blocks to determine the behavior of these on and off diagnol block elements. Let $\hat{\mathbf{M}}^{(ij)}$ be the i,j-th $(d \times d)$ block of $\hat{\mathbf{H}}^T\hat{\mathbf{H}}$. From here define the estimated mean matrix

$$\hat{\mathbf{M}} = \frac{2}{m(m+1)} \sum_{i=1}^{m} \sum_{j \leq i} \hat{\mathbf{M}}^{(ij)}.$$

and the estimated standard deviation matrix $\hat{\mathbf{S}} \in \mathbb{R}^{d \times d}$ by

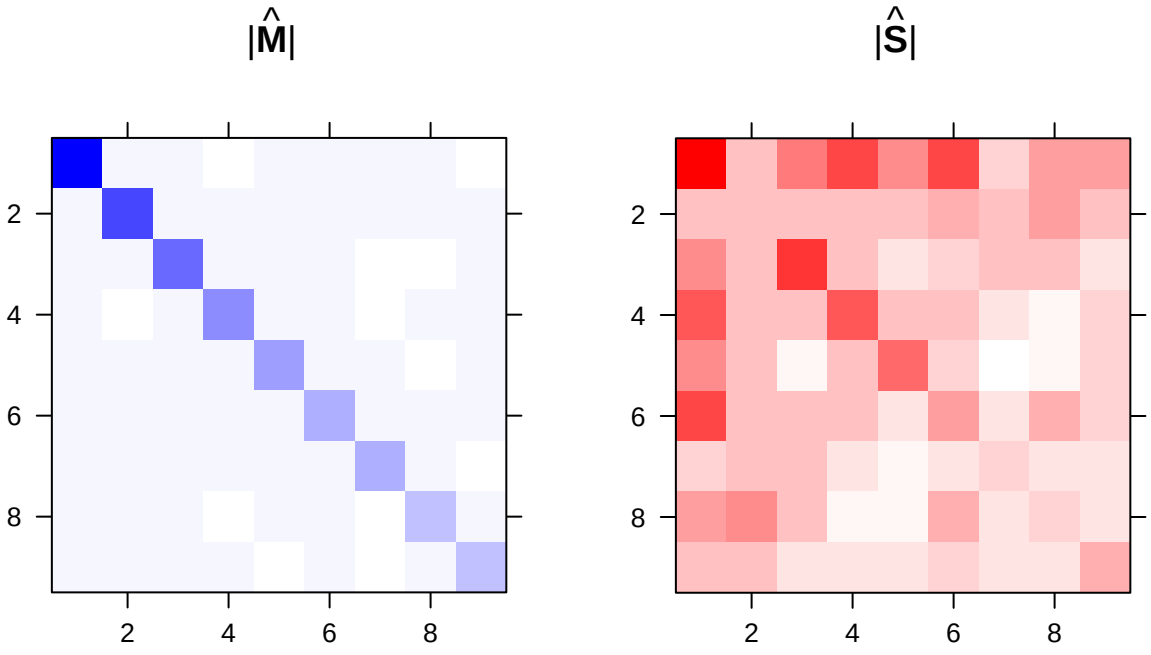
$$\hat{\mathbf{S}}_{ij} = \sqrt{\frac{2}{m(m+1)} \sum_{k=1}^{m} \sum_{\ell \leq k} (\hat{\mathbf{M}}_{ij}^{(k\ell)} - \hat{\mathbf{M}}_{ij})^2}.$$

Under the ESRDPG,

$$\mathbf{M} = \mathbb{E}[\hat{\mathbf{M}}] = \frac{2}{m(m+1)} \sum_{i=1}^{m} \sum_{j \leq i} \mathbf{S}^{(i)} \mathbf{X}^T \mathbf{X} \mathbf{S}^{(j)}$$

and $\mathbf{M}$ should be diagonal. Moreover, in theory the elementwise standard deviation matrix $\mathbf{S} = \mathbb{E}[\hat{\mathbf{S}}]$ should be diagonal. Provided that $\mathbf{M}$ is diagonal, the off-diagonal elements of $\hat{\mathbf{S}}$ indicate the level of variation from the diagonal scaling setting.

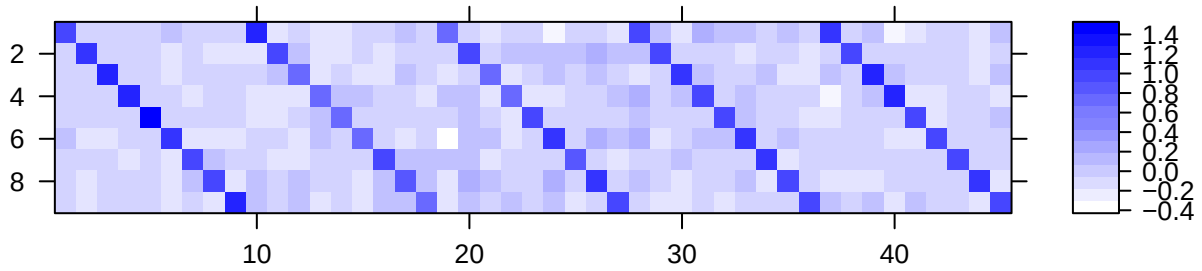
```r
# Elementwise mean and sd H^TH matrix
M <- matrix(NA, nrow = d, ncol = d)
S <- matrix(NA, nrow = d, ncol = d)

for (i in 1:d) {
    for (j in 1:d) {
        # looping over entries fetch matrix
        tmp <- crossprod(H)[seq(i, m * d, by = d),
            seq(j, m * d, by = d)]

        # all entries
        M[i, j] <- mean(tmp[upper.tri(tmp, diag = TRUE)])
        S[i, j] <- sd(tmp[upper.tri(tmp, diag = TRUE)])
    }
}

p1 <- image(Matrix(M), sub = "", xlab = "", ylab = "",
    useAbs = TRUE, col.regions = colorRampPalette(c("white",
        "blue"))(30), main = expression(paste("|",
        hat(bold(M)), "|")), border.col = NA)
p2 <- image(Matrix(S), sub = "", xlab = "", ylab = "",
    col.regions = colorRampPalette(c("white",
        "red"))(30), main = expression(paste("|",
        hat(bold(S)), "|")), border.col = NA)

grid.arrange(p1, p2, layout_matrix = matrix(1:2,
    nrow = 1))
```



Multivariable ANOVA

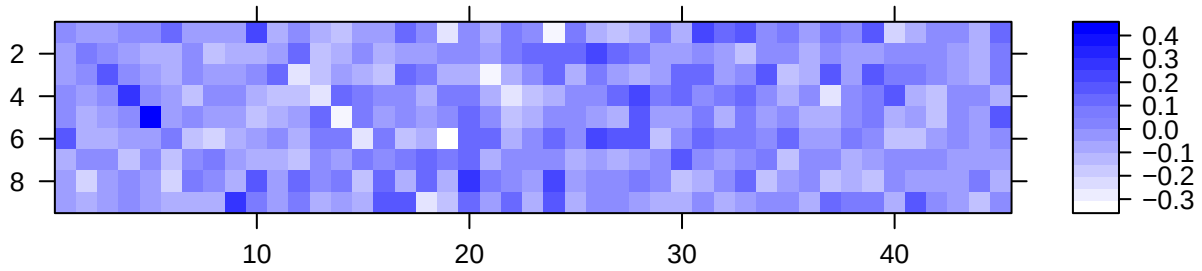
```r
# get omnibar matrix
X_sum <- matrix(0, nrow = n, ncol = d)
for (i in 1:m) {
    X_sum <- X_sum + Lhat[(n * (i - 1) + 1):(n *
        i), ]
}
X_omnibar <- X_sum/m

# regularize
anova_mat <- ginv(X_omnibar) %*% H

# Estimate Regualrized S values
image(Matrix(anova_mat), sub = "", xlab = "",
    ylab = "", col.regions = colorRampPalette(c("white",
        "blue"))(30), main = expression(paste(hat(bold(S)),
        " = ", bar(bold(X))^{
            -1
        }, hat(bold(H)))), border.col = NA)
```

$$\hat{\mathbf{S}} = \bar{\mathbf{X}}^{-1}\hat{\mathbf{H}}$$

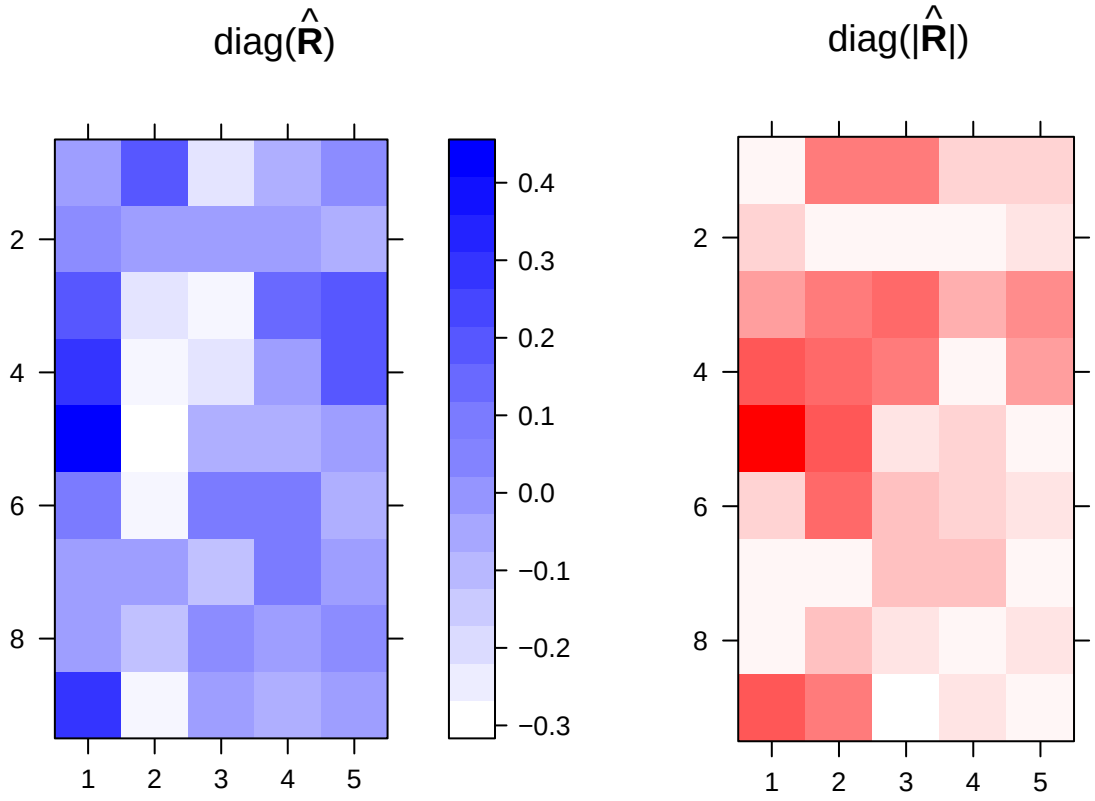



```r
#Residual Matrix
anova_center <- anova_mat - kronecker(matrix(1, ncol = m), diag(d))
image(Matrix(anova_center), sub = '', xlab = '', ylab = '',
          col.regions = colorRampPalette(c('white', 'blue'))(30),
          #main = expression(paste(bar(bold(X))^{-1}, hat(bold(H))), '-', bold(I),..., bold(I)),
          main = expression(paste(hat(bold(R)), ' = ', bar(bold(X))^{-1}, hat(bold(H)) - bold(I),..., bo
          #main = 'Noise Matrix',
          border.col = NA
          )
```

$$\hat{\mathbf{R}} = \bar{\mathbf{X}}^{-1}\hat{\mathbf{H}} - \mathbf{I}\cdots\mathbf{I}$$

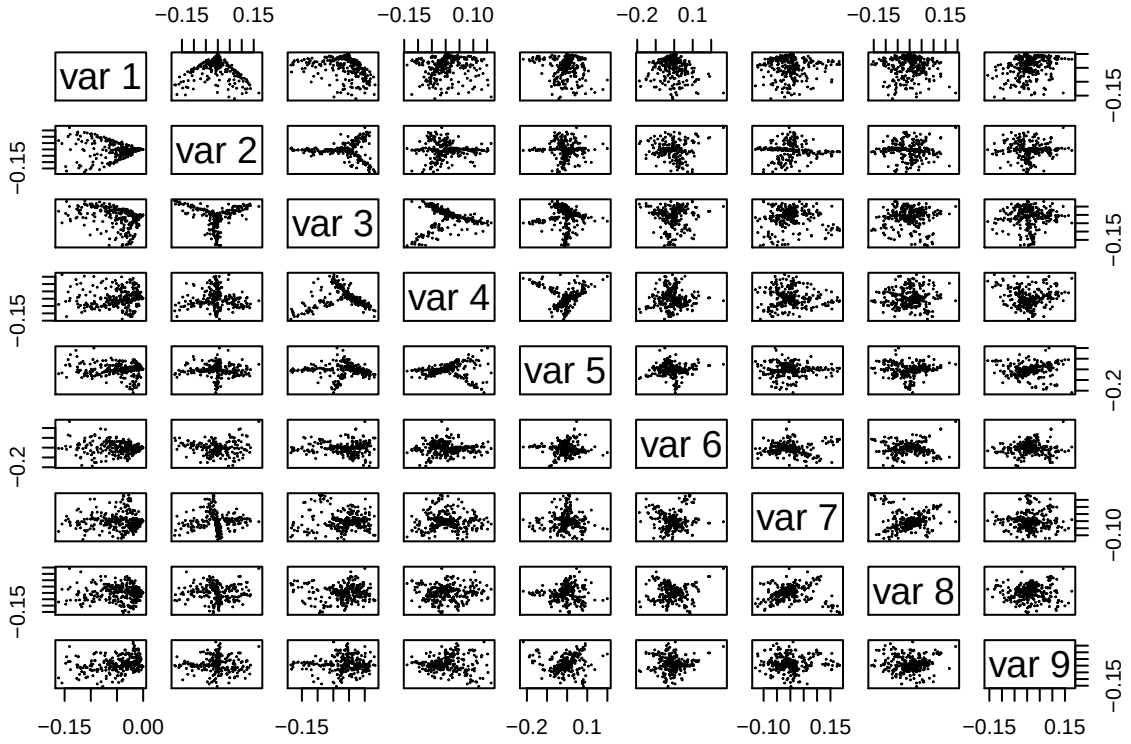


```
#Residual on diagonals
mask <- ifelse(kronecker(matrix(TRUE, ncol = m), diag(d)) == 1, TRUE, FALSE)

p1 <-image(Matrix(matrix(anova_center[mask], ncol = m, nrow = d)),
           sub = '', xlab = '', ylab = '',
           col.regions = colorRampPalette(c('white', 'blue'))(30),
           main = expression(paste('diag(',hat(bold(R)), ')')),
           border.col = NA)

p2 <-image(Matrix(matrix(anova_center[mask], ncol = m, nrow = d)),
           sub = '', xlab = '', ylab = '',
           useAbs = TRUE,
           col.regions = colorRampPalette(c('white', 'red'))(30),
           main = expression(paste('diag(|',hat(bold(R)), '|)')),
           border.col = NA)

grid.arrange(p1, p2, layout_matrix = matrix(1:2, nrow = 1))
```



Eigenspokes

```
# look at pairs plot of eigenvectors
pairs(eigen(Atil)$vectors[, 1:d], cex = 0.1)
```


var 1
var 2
var 3
var 4
var 5
var 6
var 7
var 8
var 9
−0.15 0.15
−0.15 0.10
−0.2 0.1
−0.15 0.15
−0.15
−0.15
−0.2
−0.10
−0.15
−0.15 0.00
−0.15
−0.2 0.1
−0.10 0.15
−0.15 0.15