Comparing CLARITY, Omnibus Embedding and UASE

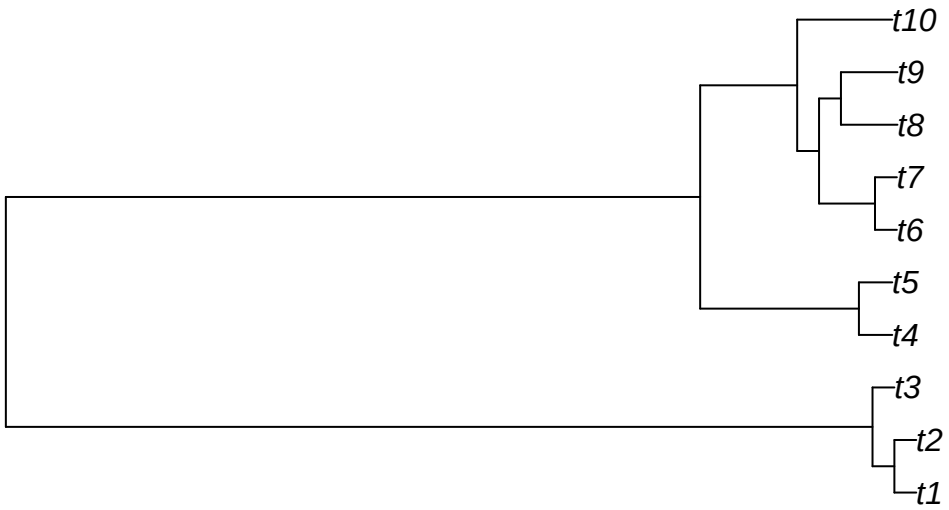
Rachel Wood

2023-03-31

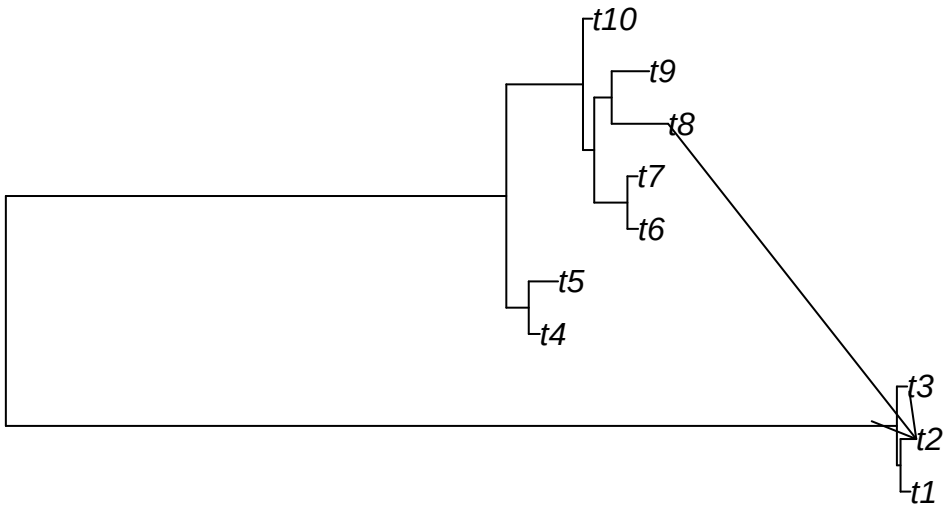
Data generation

We use the `ClaritySim` package to create the 'original' data generated according to some tree structure, which we will then scale and add a mixture to:

```
library(ClaritySim)
set.seed(124)
n <- 100
k <- 10
original <- simulateCoalescent(n,k,
                           sigma0=0.1,
                           Amodel="uniform",
                           alpha=0,
                           minedge=0.1) # Simlulate 100 objects in a 10 dimensional latent space
plot(original$tree)
```

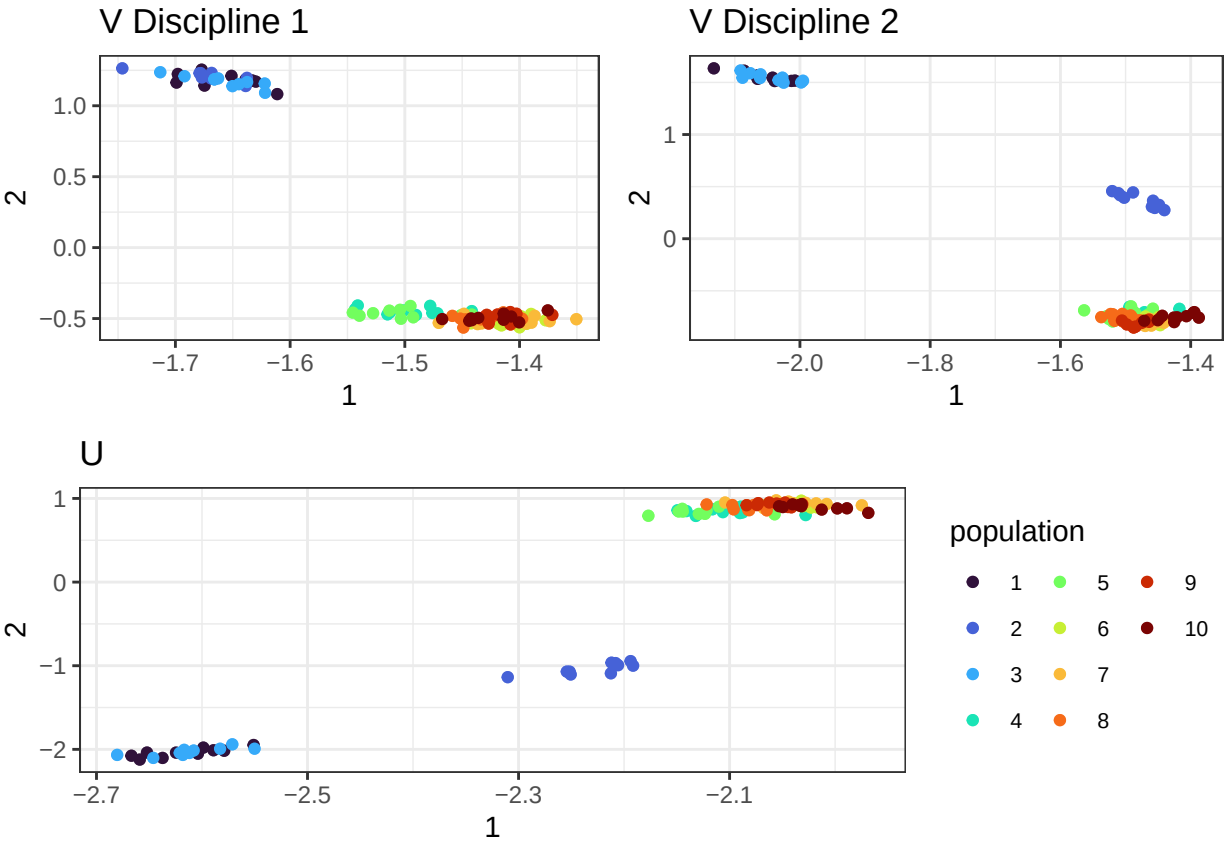



```
mixed <- mixCoalescent(original, fraction = 1, transform = TRUE)
mixture <- mixed$edges

library(ape)
plot(mixed$tree)
edges(mixture[1], mixture[2], arrows = 1)
```

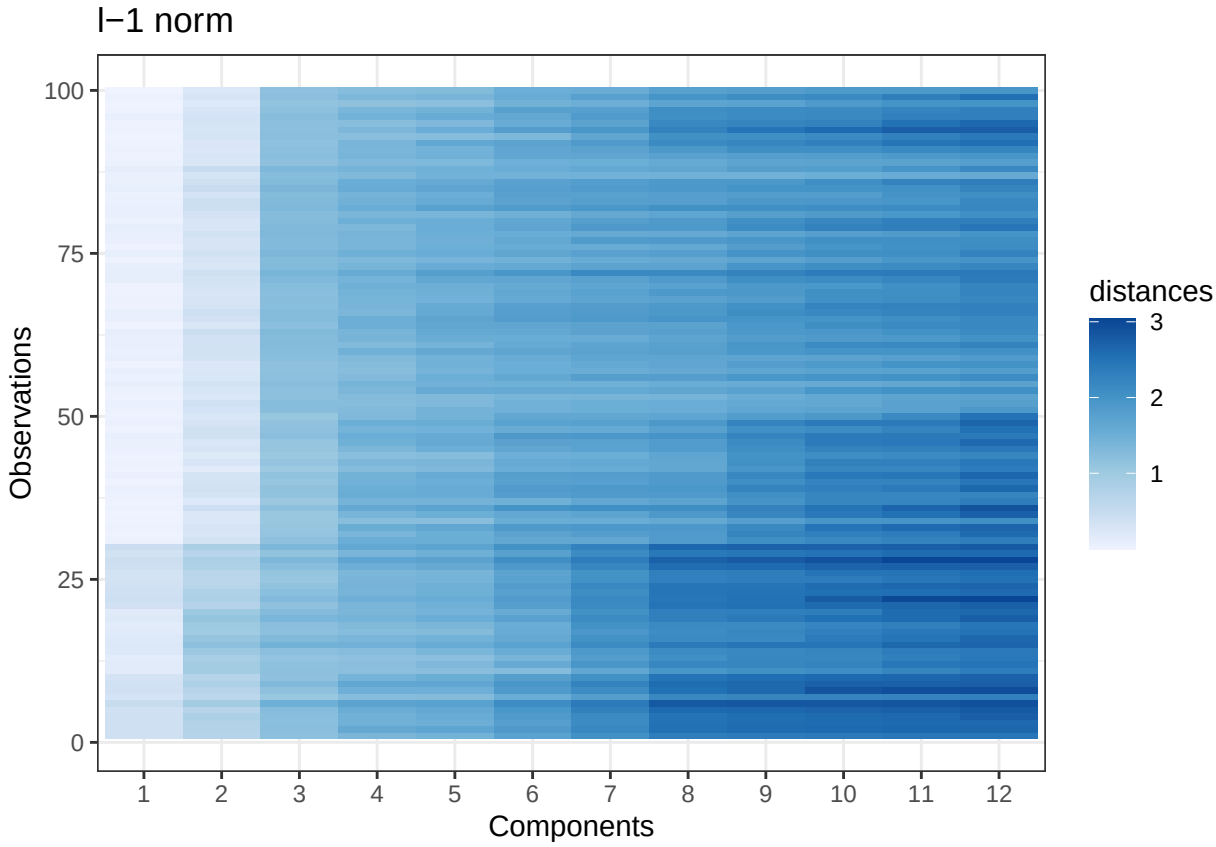


UASE

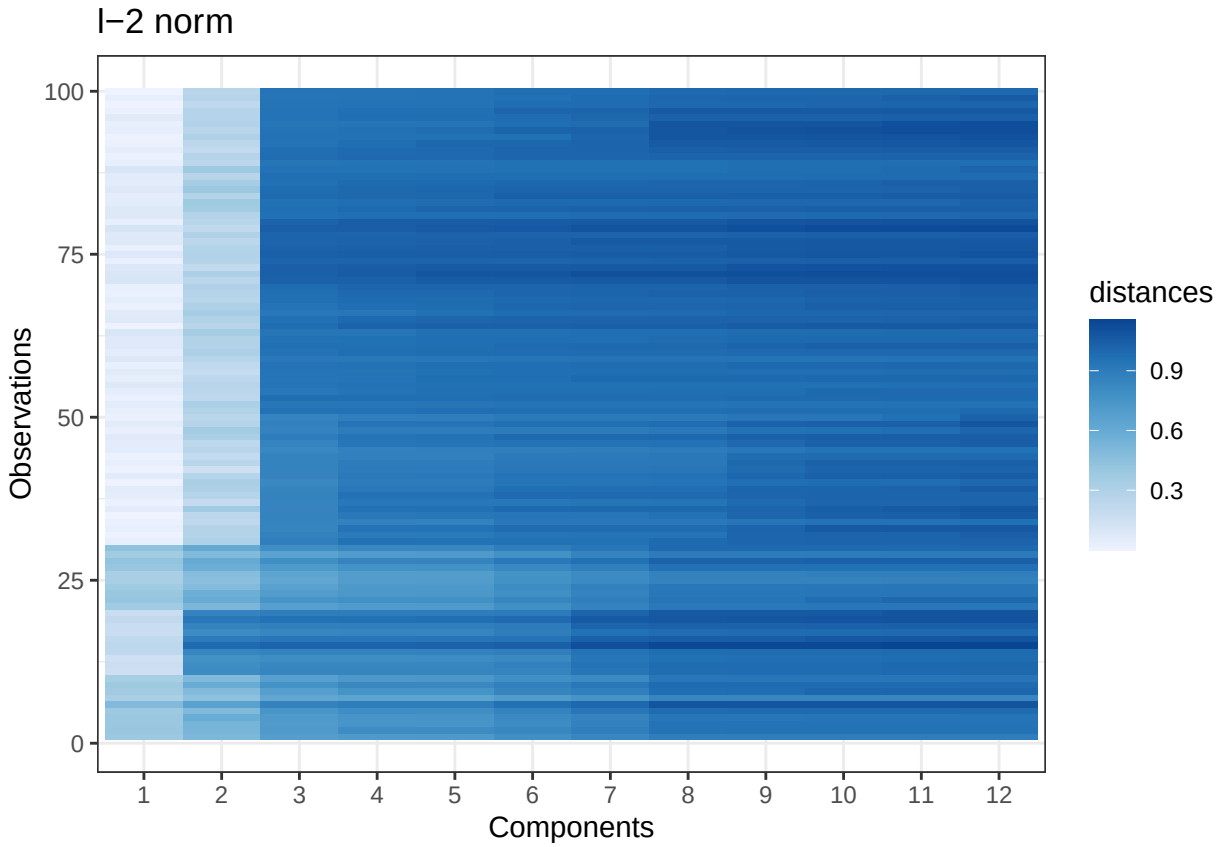
```
source("UASE.R")
groups <- original$A %*% 1:ncol(original$A)


similarity <- cbind(original$Y, mixed$Y)
mix_UASE <- UASE(similarity, d = 12, groups)
plot_UASE(mix_UASE)
```



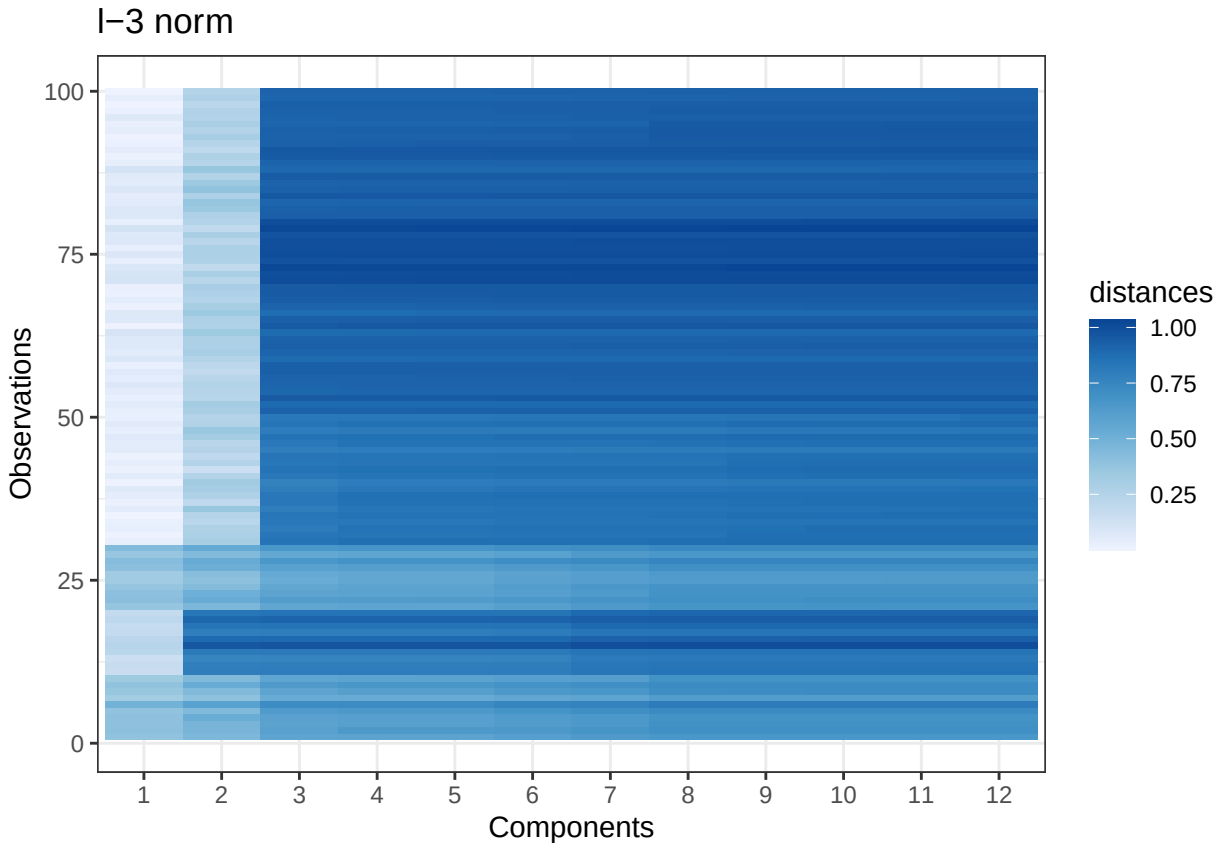
```
source("distances.R")
library(hrbrthemes)

l <- 5
plots <- list()
for (i in 1:l){
  distances <- apply(as.matrix(1:12, nrow = 1), MARGIN = 1, FUN = function(ii){
    distance_moved(mix_UASE$right, d = ii, l = i, scale = FALSE)
  })

  distance_plot <- expand.grid( Components = as.factor(1:12), Observations = 1:nrow(distances))
  distance_plot$distances <- c(t(distances))

  distance_plot <- as_tibble(distance_plot)
  plots[[i]] <- ggplot(distance_plot, aes(x = Components))+
    geom_tile(aes(y = Observations, fill = distances)) +
    scale_fill_distiller(direction = 1) +
    labs(title = paste("l-", i, " norm", sep = ""))
}

plots[[1]]
```



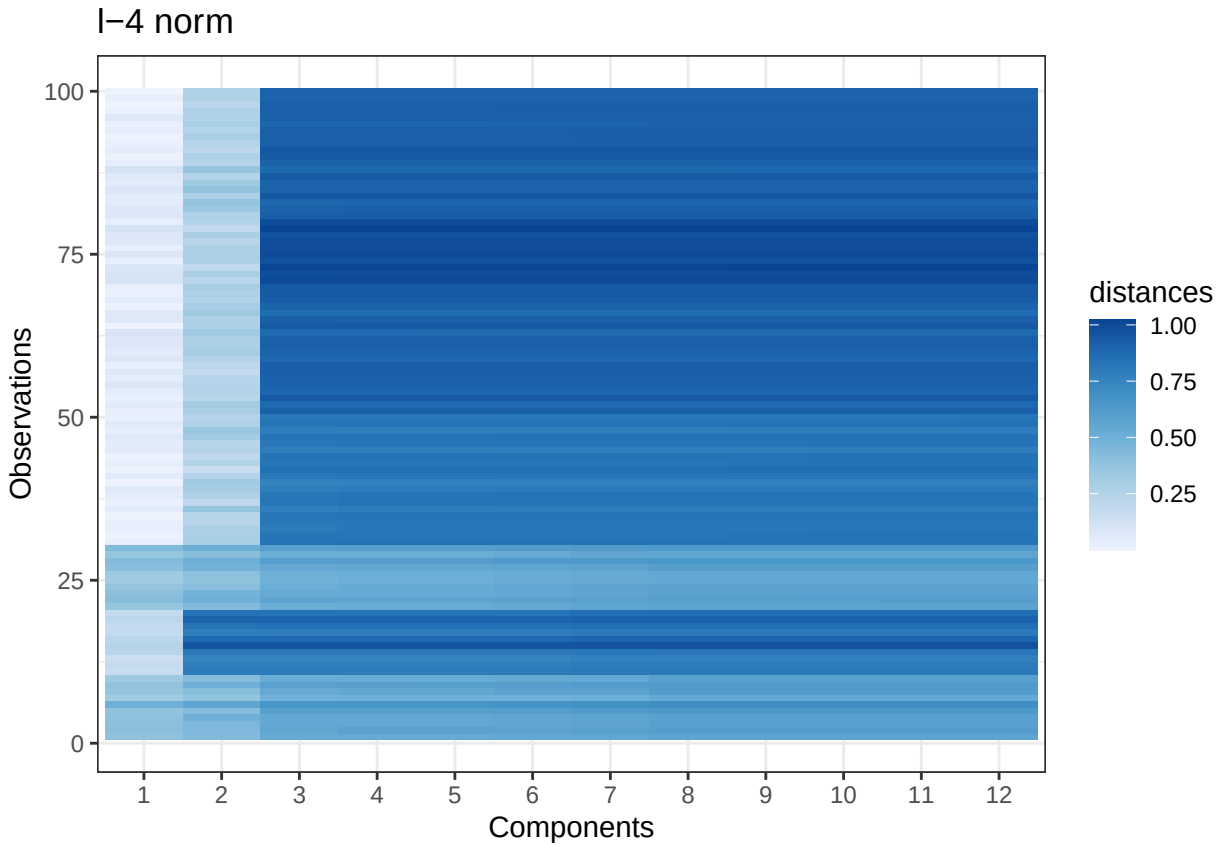

```
plots[[2]]
```


l−2 norm
Observations
100
75
50
25
0
1
2
3
4
5
6
7
8
9
10
11
12
Components
distances
0.9
0.6
0.3

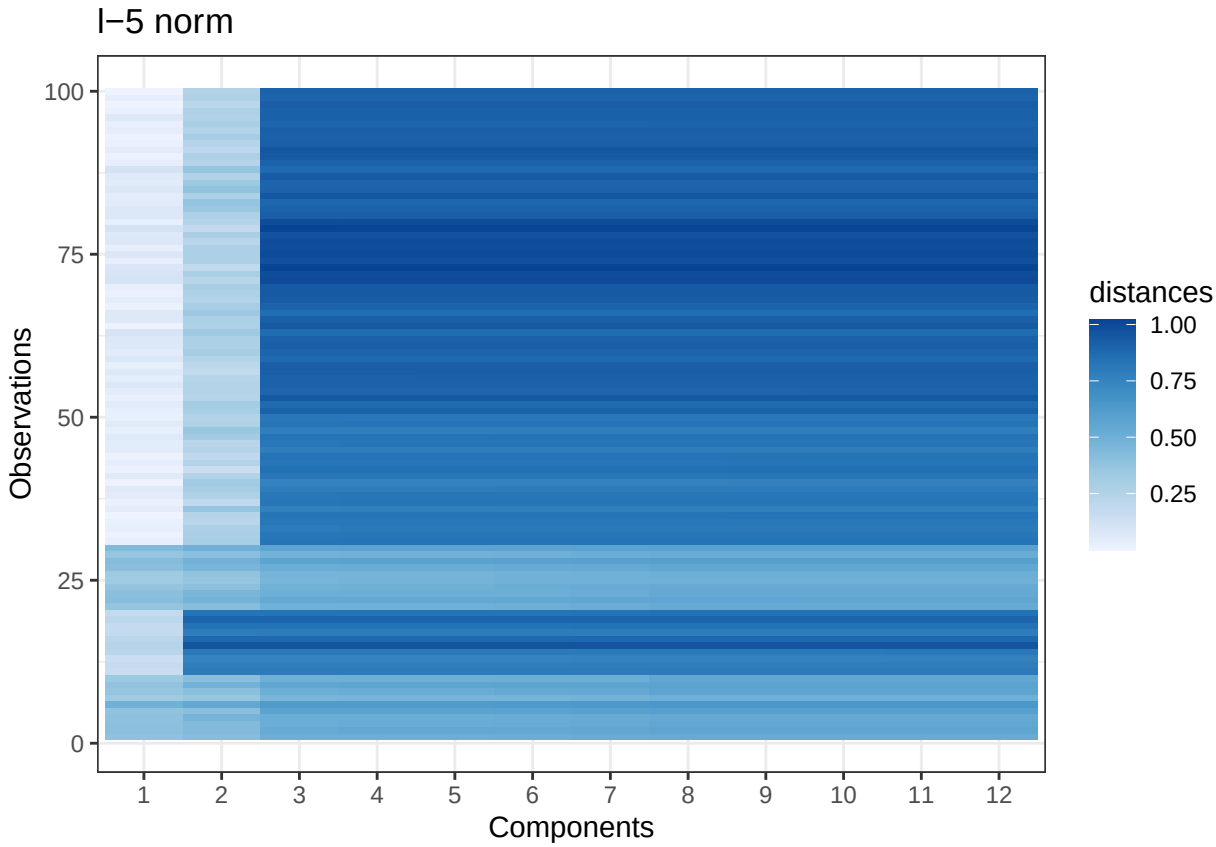
```
plots[[3]]
```


l−3 norm
Observations
100
75
50
25
0
1
2
3
4
5
6
7
8
9
10
11
12
Components
distances
1.00
0.75
0.50
0.25

```
plots[[4]]
```

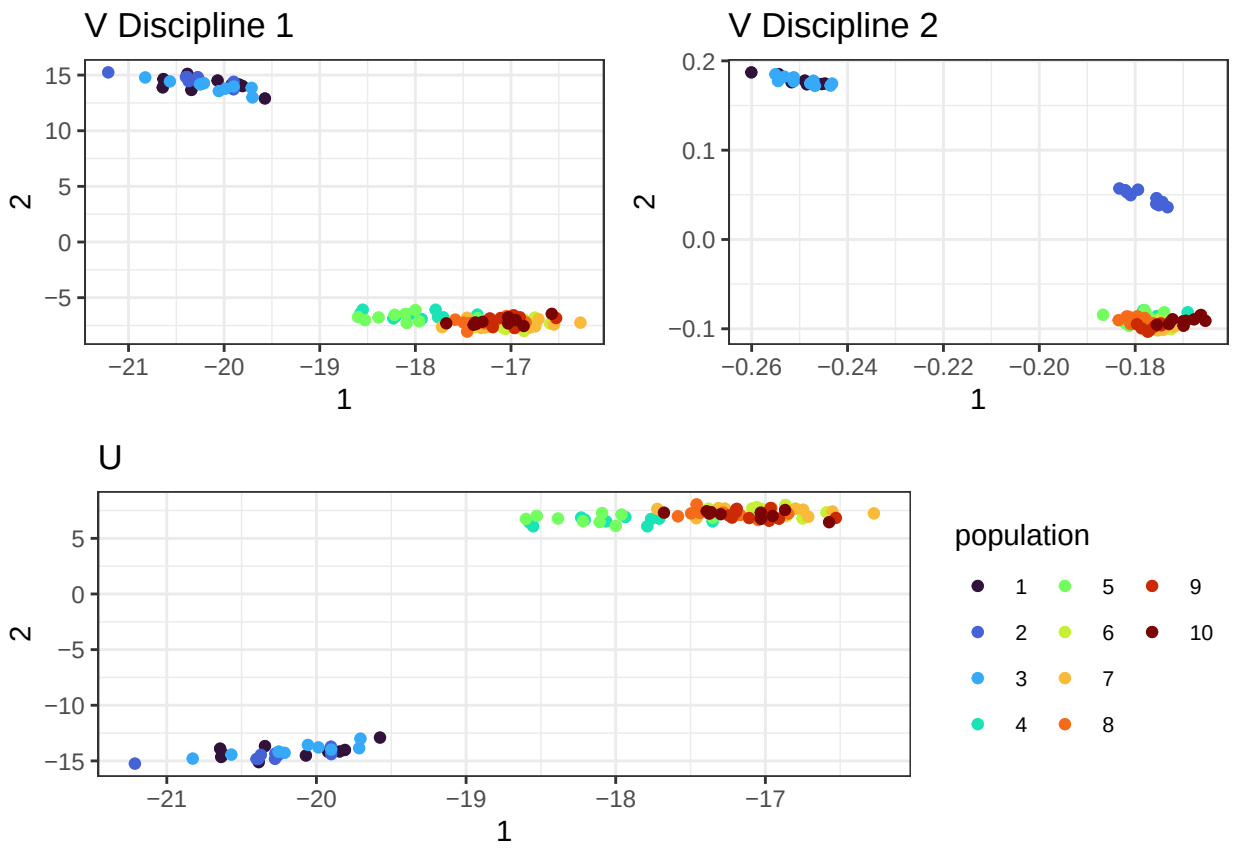

l−4 norm
Observations
100
75
50
25
0
1
2
3
4
5
6
7
8
9
10
11
12
Components
distances
1.00
0.75
0.50
0.25

```
plots[[5]]
```



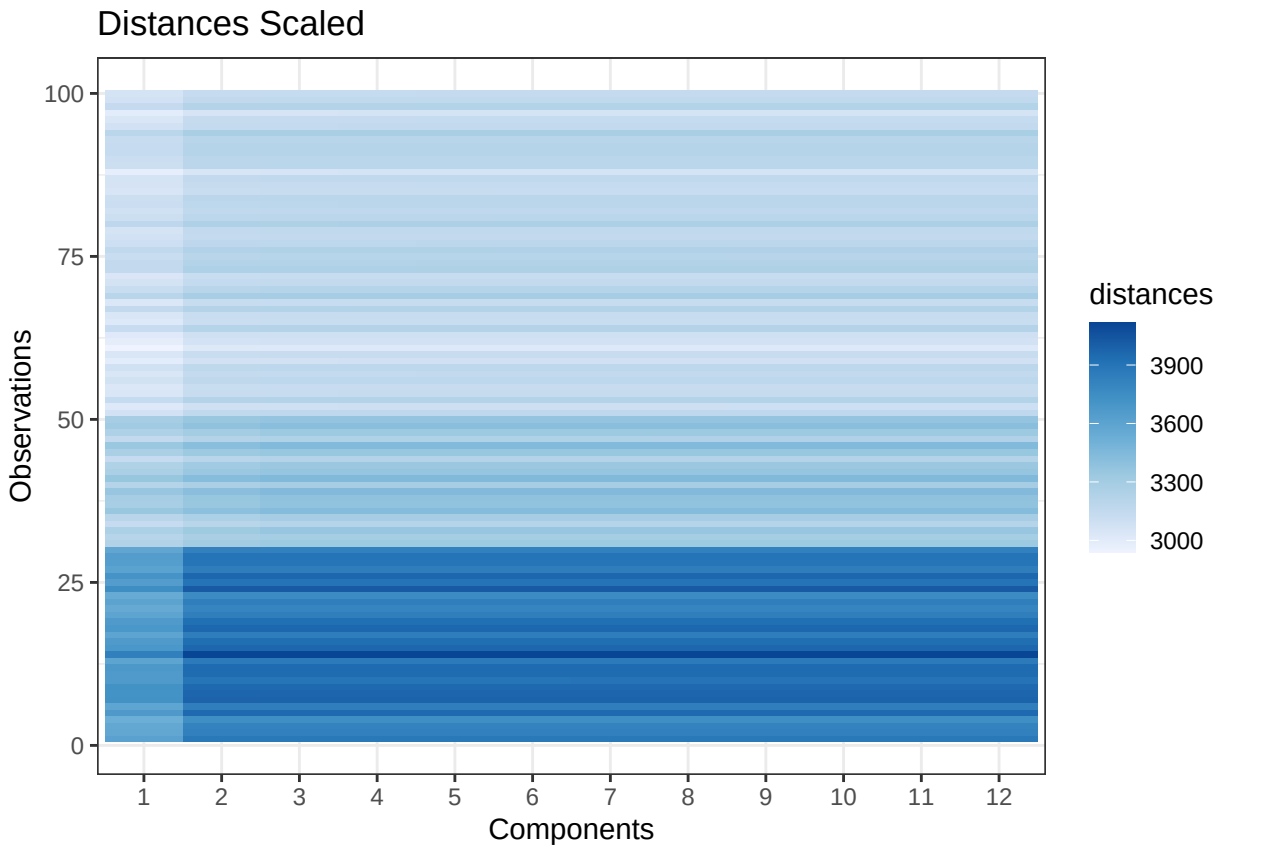
On different scales

```
test<- original
test$Y<- test$Y * 100
similarity <- cbind(test$Y, mixed$Y)
mixed_UASE<-UASE(similarity, d = 12, groups)
plot_UASE(mixed_UASE)
```

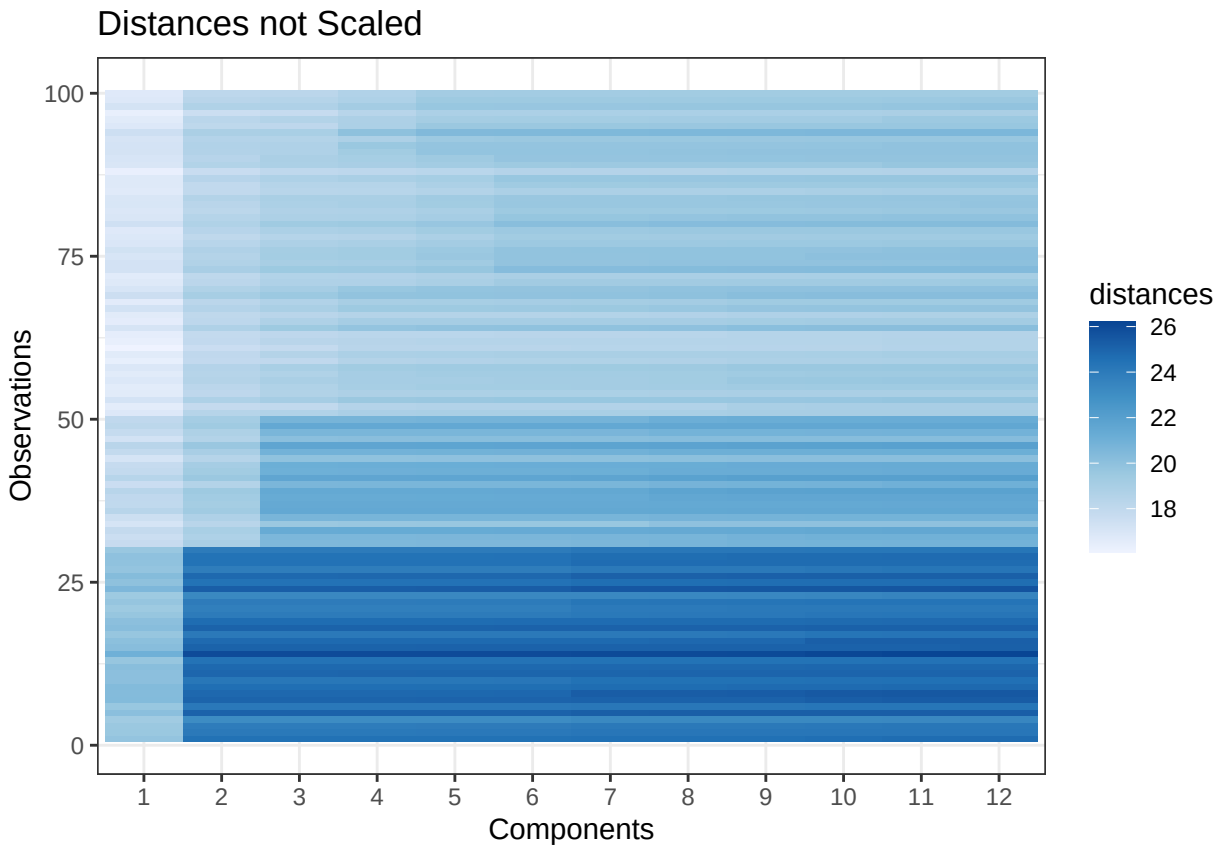



```
distances_scaled <- apply(as.matrix(1:12, nrow = 1), MARGIN = 1, FUN = function(ii){
  distance_moved(mixed_UASE$right, d = ii, scale = TRUE, eigenvals = mixed_UASE$eigenvals)
})

scaled_distance_plot <- expand.grid( Components = as.factor(1:12), Observations = 1:nrow(distances_scale
scaled_distance_plot$distances <- c(t(distances_scaled))
scaled_distance_plot$groups <- trunc( (scaled_distance_plot$Observations-1)/10) +1


scaled_distance_plot <- as_tibble(scaled_distance_plot)
p_scaled <- ggplot(scaled_distance_plot, aes(x = Components))+
  geom_tile(aes(y = Observations, fill = distances)) +
  scale_fill_distiller(direction = 1) +
  ggtitle("Distances Scaled")

distances <- apply(as.matrix(1:12, nrow = 1), MARGIN = 1, FUN = function(ii){
  distance_moved(mixed_UASE$right, d = ii, scale = FALSE)
})

distance_plot <- expand.grid( Components = as.factor(1:12), Observations = 1:nrow(distances))
distance_plot$distances <- c(t(distances))
distance_plot$groups <- trunc( (distance_plot$Observations-1)/10) +1


distance_plot <- as_tibble(distance_plot)
p <- ggplot(distance_plot, aes(x = Components))+
  geom_tile(aes(y = Observations, fill = distances)) +
```

```
  scale_fill_distiller(direction = 1) +
  ggtitle("Distances not Scaled")

p_scaled
```




```
p
```



Omnibus

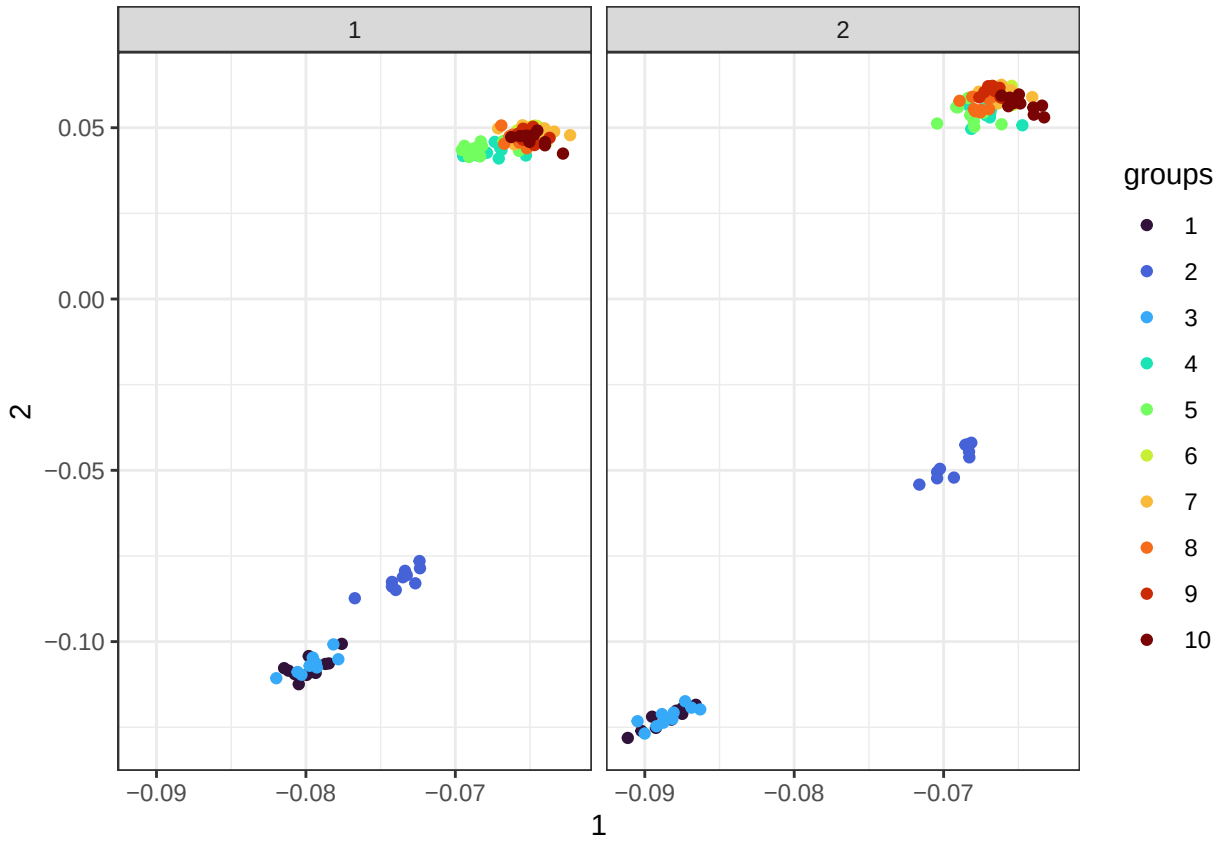
Use the `graphstats` package to compute omnibus embedding:

```
library(graphstats)

omni <- svd(gs.omni(original$Y, mixed$Y))
```

```
pcs <- 2
omni_plot <- omni$u[,1:pcs] %>% as_tibble()
```

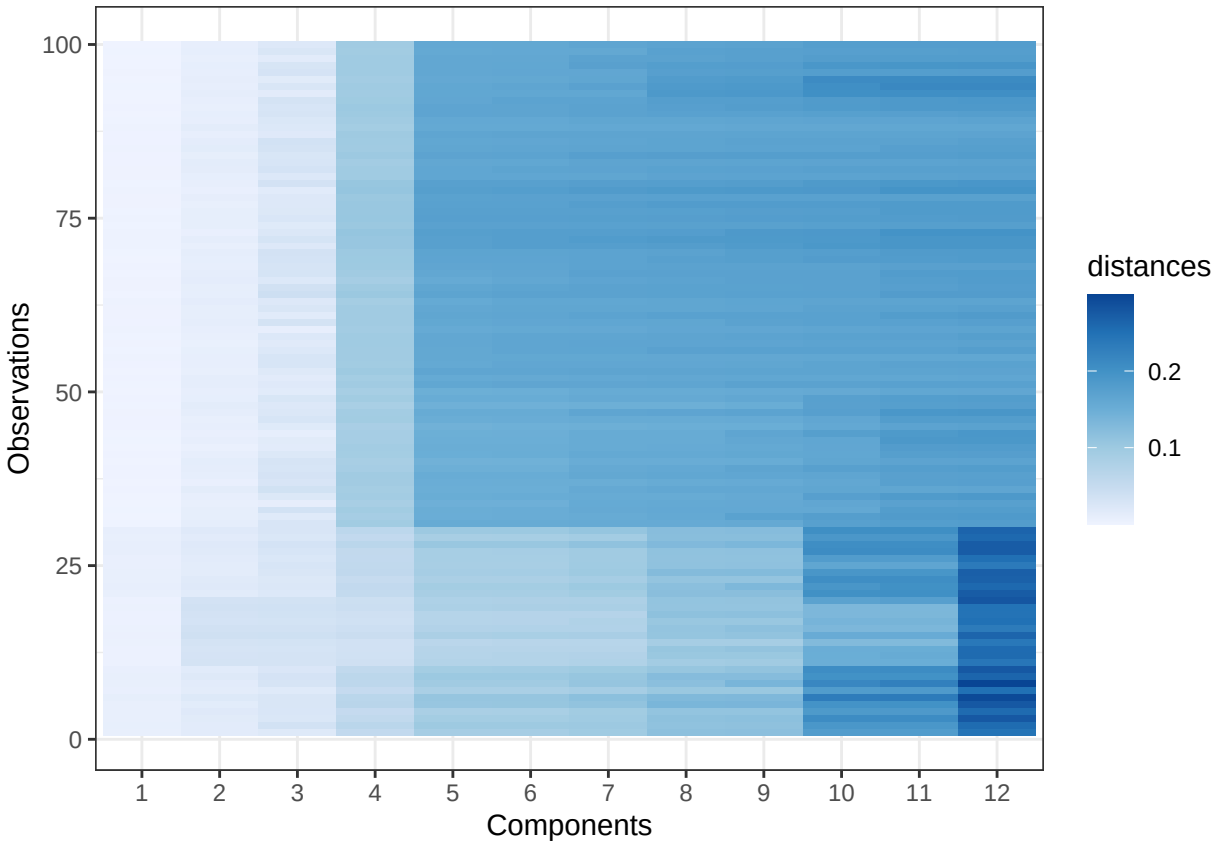
```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if
## `.name_repair` is omitted as of tibble 2.0.0.
## i Using compatibility `.name_repair`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
colnames(omni_plot) <- as.character(1:pcs)
omni_plot <- cbind( omni_plot,groups = as.factor(groups), discipline = rep(c(1,2), each = nrow(omni_plo

ggplot(omni_plot, aes(x = `1`, y = `2`)) +
  geom_point(aes(colour = groups)) +
  facet_wrap(~discipline) +
  scale_color_viridis_d(option = "turbo")
```




```
distances_omni <- apply(as.matrix(1:12, nrow = 1), MARGIN = 1, FUN = function(ii){
  distance_moved(omni$u, d = ii, scale = FALSE, eigenvals = omni$d)
})
```

```
omni_distance_plot <- expand.grid( Components = as.factor(1:12), Observations = 1:nrow(distances_omni))
omni_distance_plot$distances <- c(t(distances_omni))
omni_distance_plot$groups <- trunc( (omni_distance_plot$Observations-1)/10) +1


omni_distance_plot <- as_tibble(omni_distance_plot)
p <- ggplot(omni_distance_plot, aes(x = Components))+
  geom_tile(aes(y = Observations, fill = distances)) +
  scale_fill_distiller(direction = 1)

p
```


Observations
100
75
50
25
0
1
2
3
4
5
6
7
8
9
10
11
12
Components
distances
0.2
0.1