High-dimensional Numerical Examples

Luis Ángel Rodríguez García

2022-08-26

Numerical experiments are important for testing the effectiveness of the psc-sne algorithm since it can be identified whether the function is providing good results or not. Then, this document contains information about the simulation of high-dimensional data and defines different cases in each point.

First, we start with $p = 100$ and $r = 1$ where $(\mathbb{S}^p)^r$.

$\mathbb{S}^{100}$

Let us introduce some different scenarios where each of them has some particular distributions.

A mixture of a small circle distribution and uniform distribution

```
sc_unif_mix <- function(n, p, w_sc, w_unif, kappa = 50) {
  if (w_sc + w_unif != 1) {
    stop("w_sc and w_unif must sum 1")
  }
  n1 <- rbinom(1, n, w_sc)
  n2 <- n - n1
  r_1 <- sphunif::r_alt(n = n1, p = p, alt = "SC", kappa = kappa)
  r_2 <- sphunif::r_unif_sph(n = n2, p = p)
  data <- abind(r_1, r_2, along = 1)
  # Change the order of the data
  indexes <- sample(1:n)
  data <- data[indexes, , , drop = FALSE]
  cols <- c(rep(1, times = n1), rep(2, times = n2))
  cols <- cols[indexes]
  return(list("data" = data, "colors" = cols))
}
n <- 600
p <- 101
w_sc <- 0.5
w_unif <- 0.5
kappa <- 1000
sc_unif_mix_res <- sc_unif_mix(n = n, p = p, w_sc = w_sc, w_unif = w_unif,
                               kappa = kappa)
sc_unif_mix_data <- sc_unif_mix_res$data
sc_unif_mix_colors <- sc_unif_mix_res$colors
```

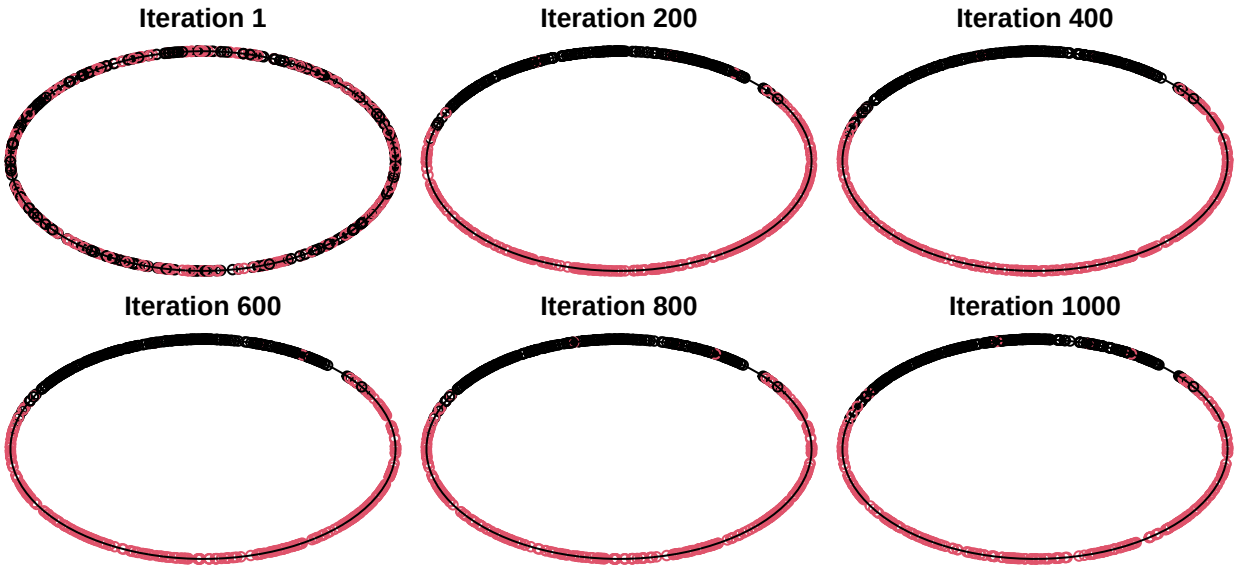
Let's now run psc-sne and see if it identifies each component of the mixture. The previous step is to calculate the rho values based on a perplexity of 30.

```
rho_30_1 <- rho_optim_bst(x = sc_unif_mix_data, perp_fixed = 30,
                          num_cores = num_cores_param)
```

```
#> Time difference of 59.20693 secs
```

The next thing is to run the psc-sne algorithm with the parameter $d = 1$:

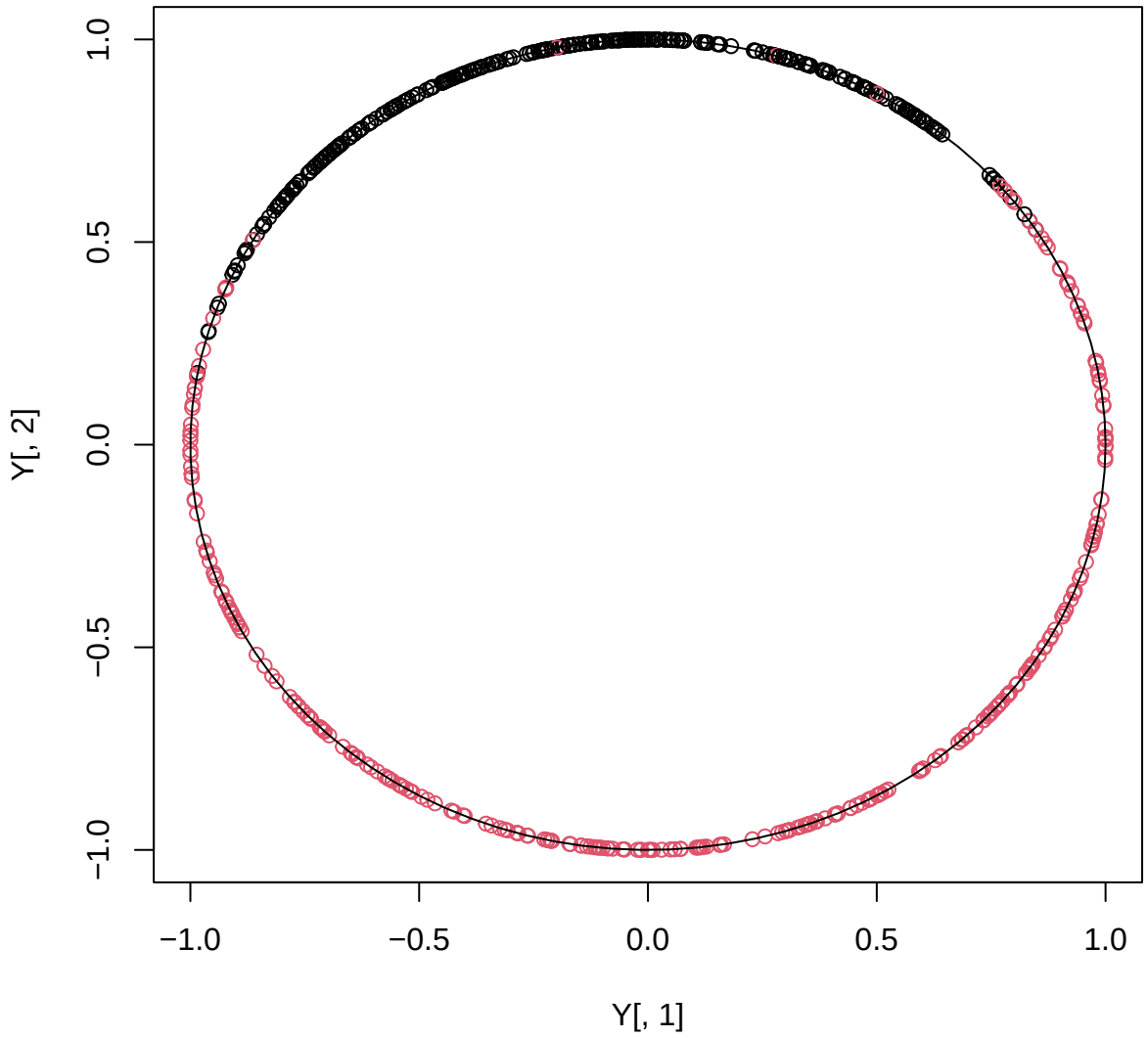
```
res_pscsne_11 <- psc_sne(X = sc_unif_mix_data, d = 1, rho_psc_list = rho_30_1,
                         show_prog = TRUE, colors = sc_unif_mix_colors,
                         parallel_cores = num_cores_param, eta = 100)
#> It: 1; obj: 1.766e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 7.530e-02; mom: 0.000e+00;
#> best it: 1; best obj: 1.766e+01
#> It: 100; obj: 1.730e+01; abs: 2.628e-01; rel: 1.496e-02; norm: 2.607e-01; mom: 6.949e+00;
#> best it: 10; best obj: 1.627e+01
#> It: 200; obj: 2.418e+00; abs: 1.667e-04; rel: 6.895e-05; norm: 9.322e-04; mom: 8.811e-02;
#> best it: 200; best obj: 2.418e+00
#> It: 300; obj: 2.413e+00; abs: 3.749e-06; rel: 1.554e-06; norm: 8.971e-05; mom: 3.582e-02;
#> best it: 300; best obj: 2.413e+00
#> It: 400; obj: 2.412e+00; abs: 9.304e-07; rel: 3.857e-07; norm: 5.910e-05; mom: 2.090e-02;
#> best it: 400; best obj: 2.412e+00
#> Warning in selectChildren(ac[!fin], -1): error 'No child processes' in select
#> It: 500; obj: 2.412e+00; abs: 6.194e-08; rel: 2.568e-08; norm: 1.111e-05; mom: 4.472e-03;
#> best it: 500; best obj: 2.412e+00
#> It: 600; obj: 2.412e+00; abs: 5.649e-09; rel: 2.342e-09; norm: 3.283e-06; mom: 1.404e-03;
#> best it: 600; best obj: 2.412e+00
#> It: 700; obj: 2.412e+00; abs: 1.279e-09; rel: 5.303e-10; norm: 1.592e-06; mom: 6.455e-04;
#> best it: 700; best obj: 2.412e+00
#> It: 800; obj: 2.412e+00; abs: 2.210e-09; rel: 9.161e-10; norm: 2.134e-06; mom: 8.173e-04;
#> best it: 800; best obj: 2.412e+00
#> It: 900; obj: 2.412e+00; abs: 2.177e-07; rel: 9.028e-08; norm: 2.198e-05; mom: 7.594e-03;
#> best it: 900; best obj: 2.412e+00
#> It: 1000; obj: 2.412e+00; abs: 3.394e-08; rel: 1.407e-08; norm: 8.081e-06; mom: 3.415e-03;
#> best it: 1000; best obj: 2.412e+00
```



The best result related with the smallest value of the objective function is:

```
Y <- res_pscsne_11$best_Y
plot(Y[, 1], Y[, 2], col = sc_unif_mix_colors, xlim = c(-1, 1),
     ylim = c(-1, 1))
```

```
polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```

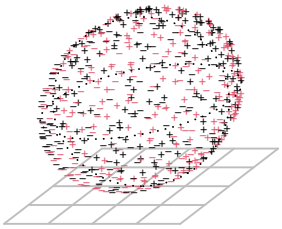


After that, let's execute psc-sne with the parameter $d = 2$:

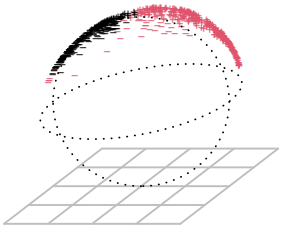
```
res_pscsne_12 <- psc_sne(X = sc_unif_mix_data, d = 2, rho_psc_list = rho_30_1,
                         show_prog = TRUE, colors = sc_unif_mix_colors,
                         parallel_cores = num_cores_param, eta = 10)
#> It: 1; obj: 1.953e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.606e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.953e+01
#> It: 100; obj: 1.685e+01; abs: 2.756e-05; rel: 1.636e-06; norm: 1.988e-04; mom: 2.202e-03;
#> best it: 17; best obj: 1.596e+01
#> It: 200; obj: 2.412e+00; abs: 9.375e-03; rel: 3.871e-03; norm: 2.203e-02; mom: 2.076e-01;
#> best it: 200; best obj: 2.412e+00
#> It: 300; obj: 2.216e+00; abs: 9.324e-04; rel: 4.206e-04; norm: 4.267e-03; mom: 1.784e-01;
#> best it: 300; best obj: 2.216e+00
#> It: 400; obj: 2.191e+00; abs: 7.402e-05; rel: 3.379e-05; norm: 1.258e-03; mom: 4.649e-02;
#> best it: 400; best obj: 2.191e+00
#> It: 500; obj: 2.184e+00; abs: 3.555e-05; rel: 1.628e-05; norm: 8.547e-04; mom: 3.334e-02;
#> best it: 500; best obj: 2.184e+00
```

```
#> It: 600; obj: 2.181e+00; abs: 1.206e-05; rel: 5.531e-06; norm: 4.909e-04; mom: 1.974e-02;
#> best it: 600; best obj: 2.181e+00
#> It: 700; obj: 2.180e+00; abs: 1.043e-05; rel: 4.785e-06; norm: 4.586e-04; mom: 1.818e-02;
#> best it: 700; best obj: 2.180e+00
#> It: 800; obj: 2.179e+00; abs: 1.187e-05; rel: 5.449e-06; norm: 4.811e-04; mom: 2.006e-02;
#> best it: 800; best obj: 2.179e+00
#> It: 900; obj: 2.178e+00; abs: 3.355e-06; rel: 1.540e-06; norm: 2.584e-04; mom: 1.043e-02;
#> best it: 900; best obj: 2.178e+00
#> It: 1000; obj: 2.178e+00; abs: 5.611e-06; rel: 2.576e-06; norm: 3.354e-04; mom: 1.344e-02;
#> best it: 1000; best obj: 2.178e+00
```

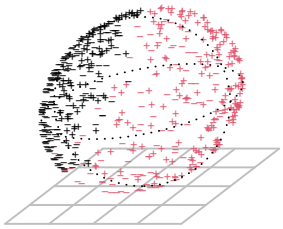
Iteration 1



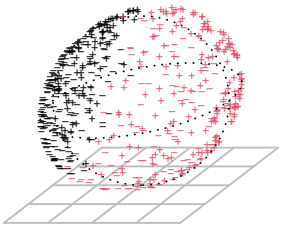
Iteration 200



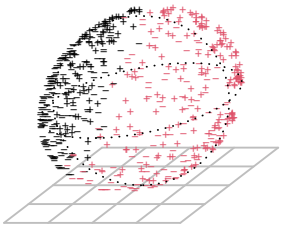
Iteration 400



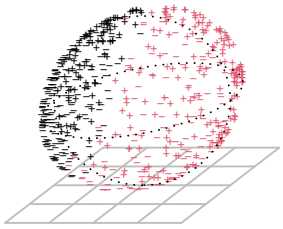
Iteration 600



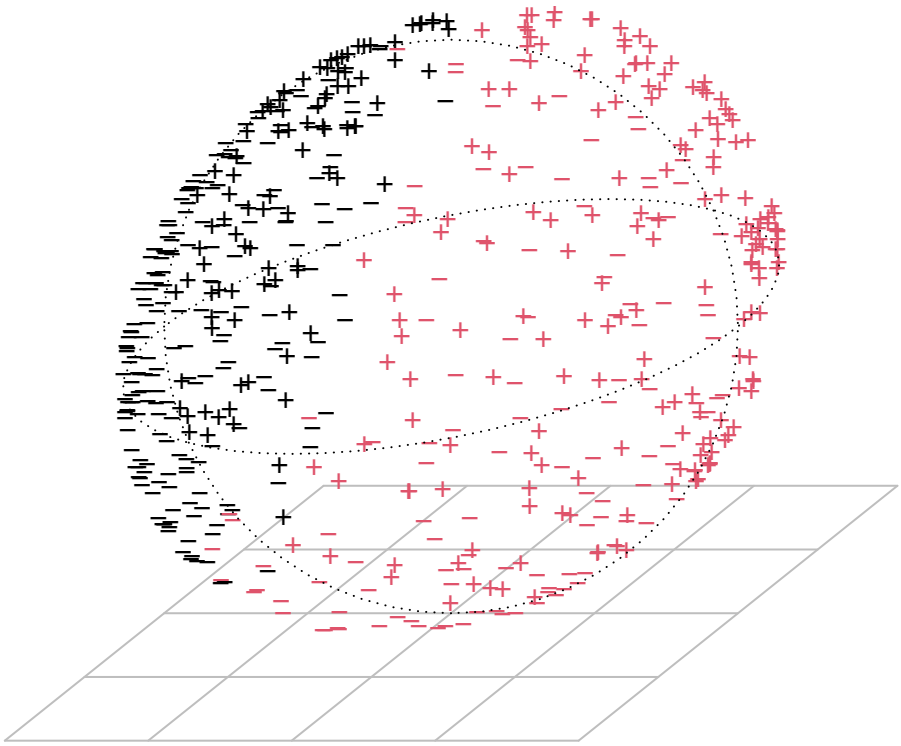
Iteration 800



Iteration 1000



```
seq_rad <- seq(-pi, pi, by = pi / 30)
meridian <- do.call(rbind, lapply(seq_rad, function(i) c(0, i)))
equator <- do.call(rbind, lapply(seq_rad, function(i) c(i, pi/2)))
Y <- res_pscsne_12$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = sc_unif_mix_colors, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



An example in variable selection

We define 5 groups of 20 variables that are strongly positive correlated each of them. For example, consider

$$(\mathbf{X}_1, \ldots, \mathbf{X}_5)' \sim \mathcal{N}_{100}(\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

where

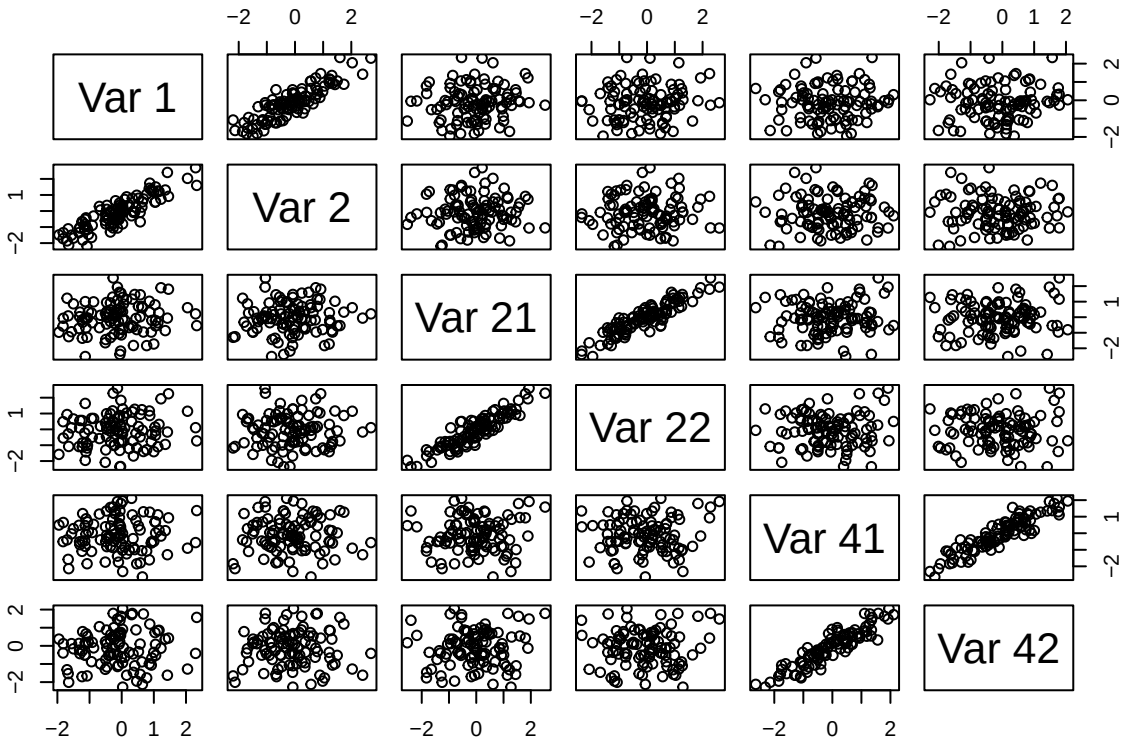
$$\boldsymbol{\Sigma} = \text{diag}(\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_5)$$

is block-wise diagonal. Take now $n = 200$ observations and center and standardize the features. Then the features now live in $\mathbb{S}^{n-1}$.

Then, we consider 10 groups of 20 variables each of them with positive correlations. A covariance matrix of size `c(200, 200)` is created and filled in only those entries that belongs to the i-th group with the associated entry of the vector `rho` and powers of itself. Along with the mean vector defined as zeros, the parameters of the multivariate normal are already set up and some data are generated. These points have the peculiarity to lie onto the sphere, therefore, spherical data.

In order to avoid observations that belongs to the same group together, we have shuffled these points so the data-set is disorganized.

```
# Visualize some features
n <- 100
g <- 5
p <- 20
x <- r_block(n = n, g = g, p = p)
pairs(x[, c(1:2, p + 1:2, (2 * p) + 1:2)],
      labels = c("Var 1", "Var 2", paste("Var", p + 1), paste("Var", p + 2),
                 paste("Var", (2 * p) + 1), paste("Var", (2 * p) + 2)))
```




```
# Standardize variables -- now the vectors of observations for each variable
# (the columns) live on \sqrt{n - 1} * S^{n - 1}!
x_sca <- scale(x)
# Make the features live on S^{n - 1}
x_sca <- x_sca / sqrt(n - 1)
# Transpose matrix (features become observations)
feat_data <- t(x_sca)
# Colors of the groups
cols <- rep(1:g, each = p)
# Set an array of dimension c(g * p, n, 1)
dim(feat_data) <- c(dim(feat_data), 1)
# Change the order of the data
indexes <- sample(1:(g * p))
feat_data <- feat_data[indexes, , , drop = FALSE]
cols <- cols[indexes]
```

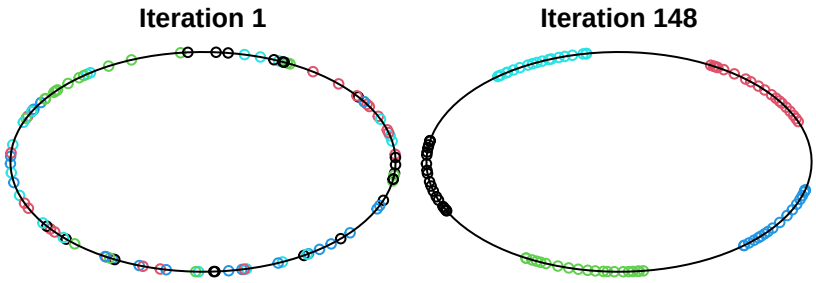
Let's calculate the rho based on a perplexity of 30

```
rho_list <- rho_optim_bst(x = feat_data, perp_fixed = 30,
                          num_cores = num_cores_param)
#> Time difference of 2.500508 secs
```

Run `psc-sne` on the standardized features on $\mathbb{S}^{199}$ for 100 observations being the colors the groups of variables, first set $d = 1$:

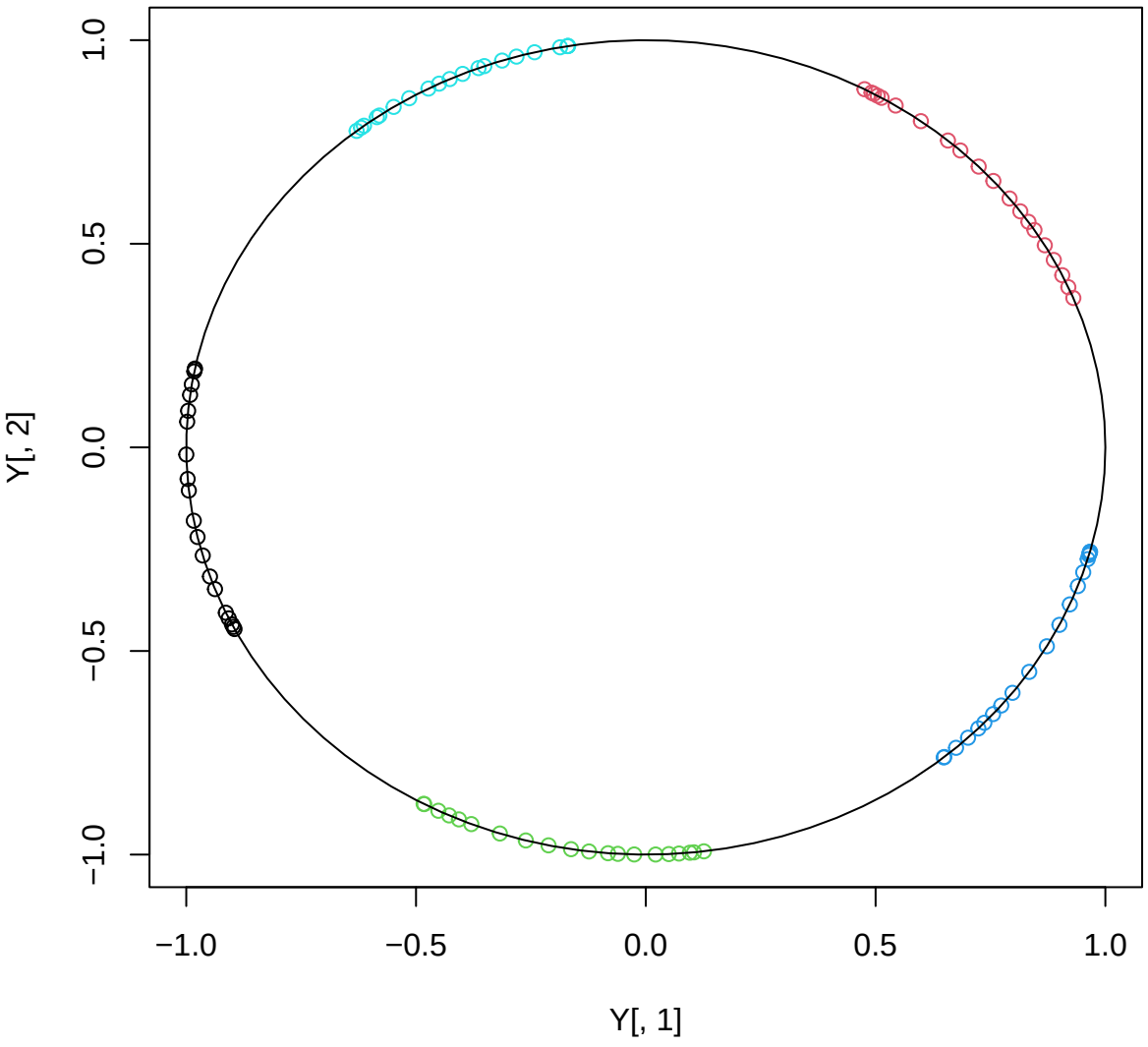
```
res_pscsne_21 <- psc_sne(X = feat_data, d = 1, rho_psc_list = rho_list,
                         colors = cols, show_prog = TRUE,
                         eta = 35, parallel_cores = num_cores_param)
#> It: 1; obj: 1.076e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.646e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.076e+01
#> It: 100; obj: 1.058e+01; abs: 1.049e-01; rel: 9.810e-03; norm: 4.493e-01; mom: 3.822e+00;
#> best it: 26; best obj: 8.692e+00
```

```
#> It: 148; obj: 4.827e-01; abs: 1.152e-11; rel: 2.386e-11; norm: 8.698e-07; mom: 1.365e-05;
#> best it: 148; best obj: 4.827e-01
```



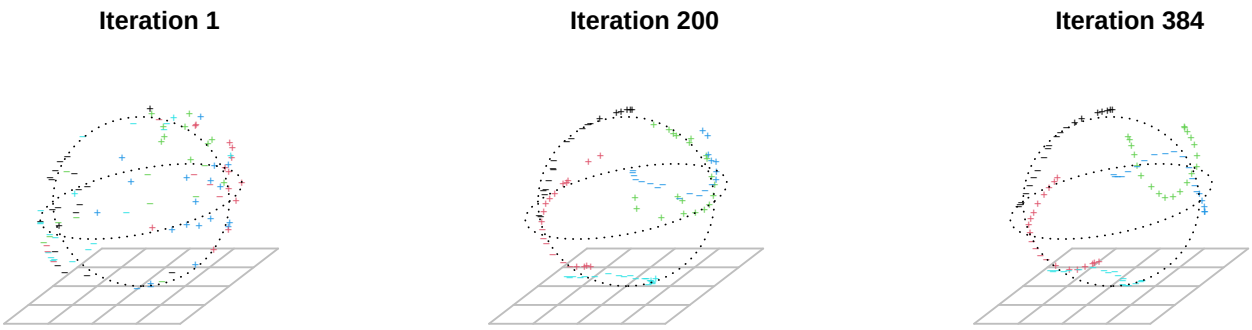
The best result related with the smallest value of the objective function is:

```
Y <- res_pscsne_21$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1),
     ylim = c(-1, 1))

polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```



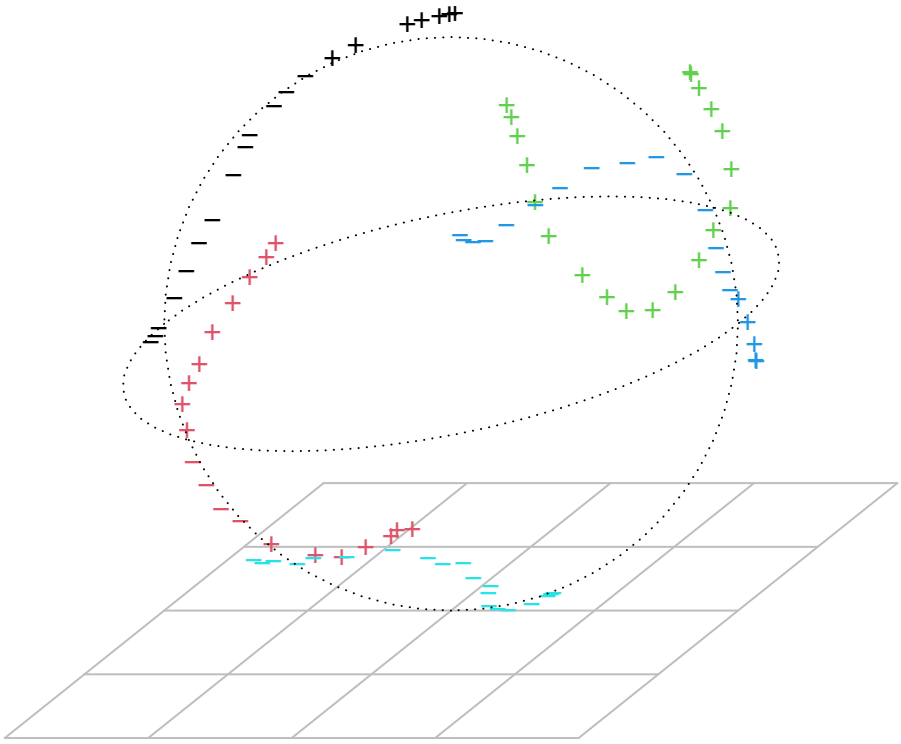
The psc-sne algorithm identifies in the reduced dimension on the circumference the five groups that are in this case strongly positive correlated within each group's variables. Let's do the same for $d = 2$:

```
psc_sne_res_22 <- psc_sne(feat_data, d = 2, rho_psc_list = rho_list,
                          colors = cols, show_prog = TRUE,
                          eta = 35, parallel_cores = num_cores_param)
#> It: 1; obj: 1.092e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 3.583e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.092e+01
#> It: 100; obj: 1.153e+01; abs: 5.007e-01; rel: 4.161e-02; norm: 9.112e-01; mom: 5.232e+00;
#> best it: 5; best obj: 9.637e+00
#> It: 200; obj: 1.491e-01; abs: 1.959e-04; rel: 1.315e-03; norm: 7.788e-02; mom: 8.639e-01;
#> best it: 165; best obj: 1.465e-01
#> It: 300; obj: 1.107e-01; abs: 1.004e-04; rel: 9.063e-04; norm: 7.161e-03; mom: 1.559e-01;
#> best it: 300; best obj: 1.107e-01
#> It: 384; obj: 1.083e-01; abs: 9.549e-13; rel: 8.817e-12; norm: 8.945e-07; mom: 3.635e-05;
#> best it: 384; best obj: 1.083e-01
```



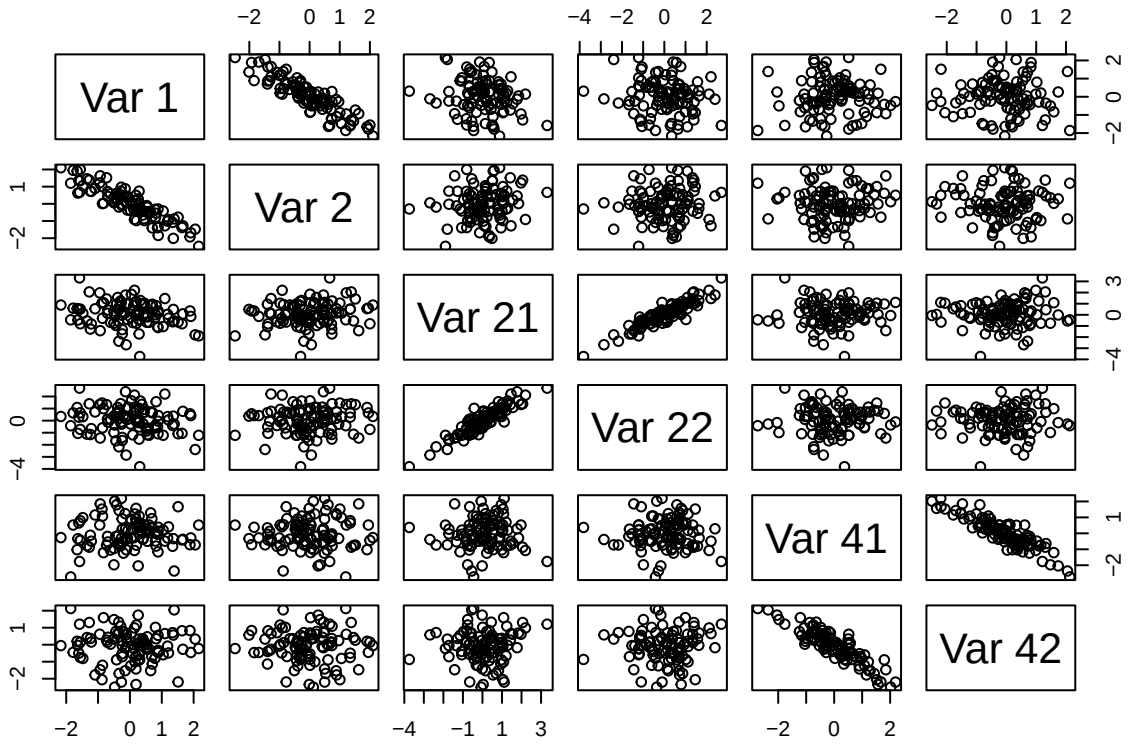
In case of the scores projected onto the sphere, results are good since the five groups are clearly separated along the sphere. Let's see the points in the interactive sphere using the package `rgl`.

```
Y <- psc_sne_res_22$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



Let's do the same but taking now negative and positive correlations for each group. For example, for the first group we are going to take a significant negative correlation, for the second a large correlation and so on.

```
# Visualize some features
n <- 100
g <- 5
p <- 20
rho_values_neg <- rep(c(-0.9, 0.9), times = g)[1:g]
x <- r_block(n = n, g = g, p = p, rho = rho_values_neg)
pairs(x[, c(1:2, p + 1:2, (2 * p) + 1:2)],
      labels = c("Var 1", "Var 2", paste("Var", p + 1), paste("Var", p + 2),
                 paste("Var", (2 * p) + 1), paste("Var", (2 * p) + 2)))
```




```
# Standardize variables -- now the vectors of observations for each variable
# (the columns) live on \sqrt{n - 1} * S^{n - 1}!
x_sca <- scale(x)
# Make the features live on S^{n - 1}
x_sca <- x_sca / sqrt(n - 1)
# Transpose matrix (features become observations)
feat_data_2 <- t(x_sca)
# Colors of the groups
cols <- rep(1:g, each = p)
# Set an array of dimension c(g * p, n, 1)
dim(feat_data_2) <- c(dim(feat_data_2), 1)
# Change the order of the data
indexes <- sample(1:(g * p))
feat_data_2 <- feat_data_2[indexes, , , drop = FALSE]
cols <- cols[indexes]
```

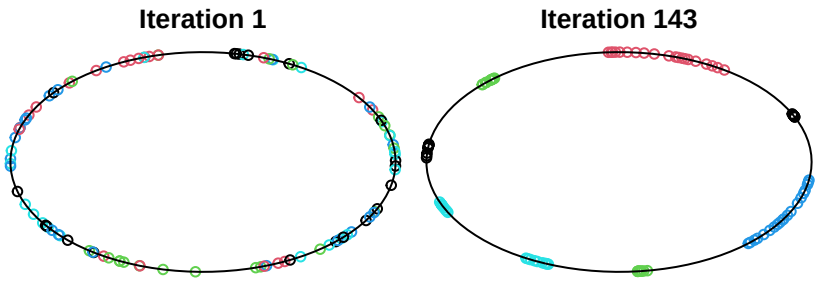
Let's calculate the rho based on a perplexity of 30

```
rho_list_2 <- rho_optim_bst(x = feat_data_2, perp_fixed = 30,
                            num_cores = num_cores_param)
#> Time difference of 2.178255 secs
```

Run psc_sne() with colors being the groups of variables. First, reduced the data on the circumference.

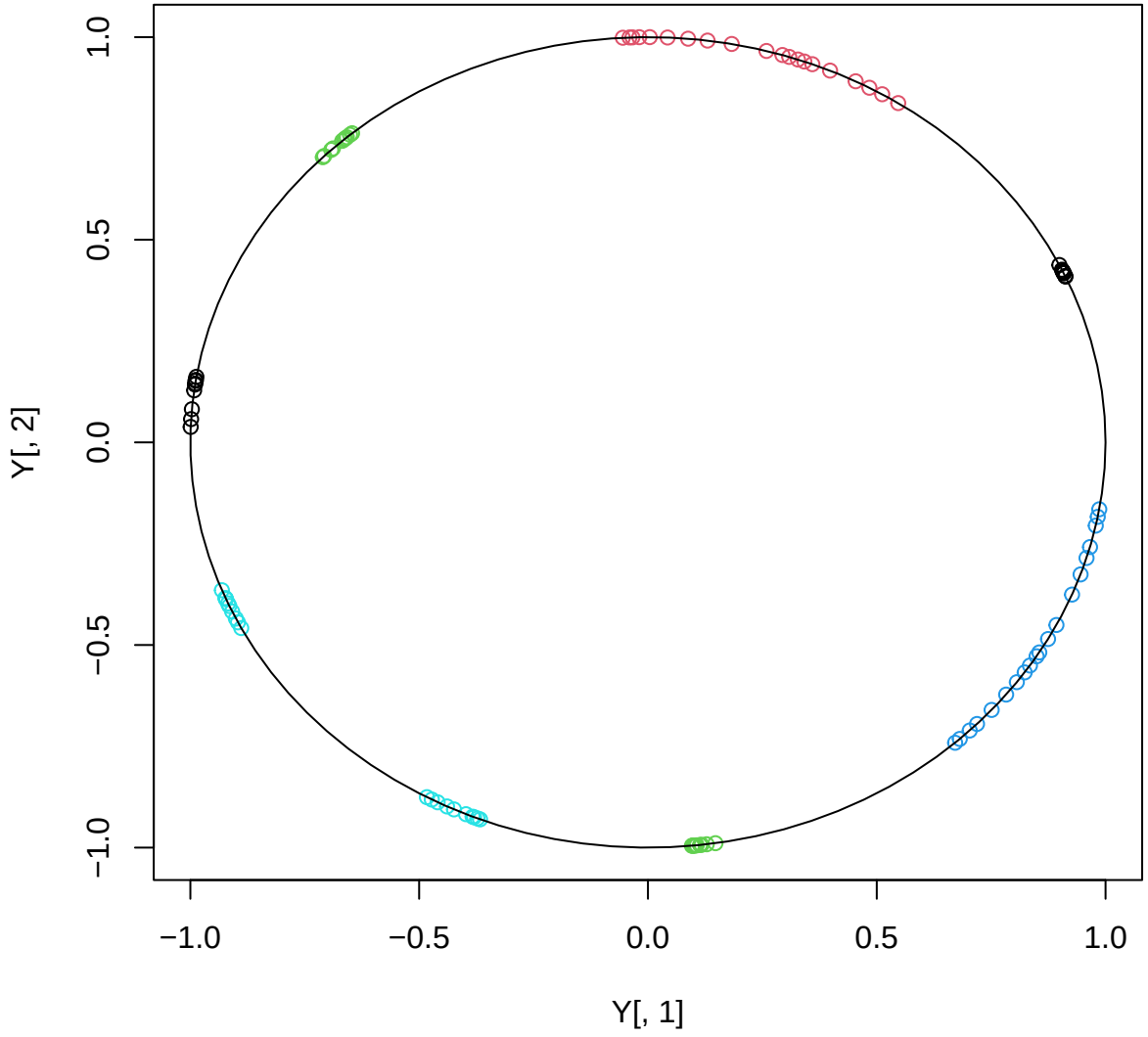
```
res_pscsne_21_neg <- psc_sne(X = feat_data_2, d = 1, rho_psc_list = rho_list_2,
                      colors = cols, show_prog = TRUE, eta = 35,
                      parallel_cores = num_cores_param)
#> It: 1; obj: 1.071e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.782e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.071e+01
#> It: 100; obj: 1.133e+01; abs: 8.916e-01; rel: 8.542e-02; norm: 4.710e-01; mom: 3.874e+00;
#> best it: 9; best obj: 9.174e+00
#> It: 143; obj: 5.778e-01; abs: 4.450e-13; rel: 7.701e-13; norm: 6.399e-07; mom: 3.156e-05;
```

```
#> best it: 143; best obj: 5.778e-01
```



The best result related with the smallest value of the objective function is:

```
Y <- res_pscsne_21_neg$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1),
     ylim = c(-1, 1))

polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```

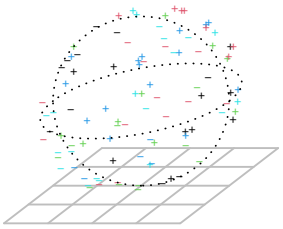


In this case, the negative correlation in some groups has an effect on the results produced by psc-sne. Each

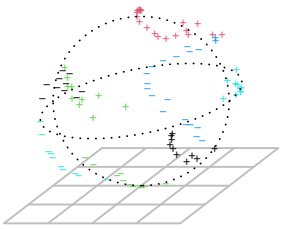
group with negative correlation is split in two where they look like being distributed more or less antipodal. Later on, reduced on the sphere $d = 2$.

```
psc_sne_res_22_neg <- psc_sne(feat_data_2, d = 2, rho_psc_list = rho_list_2,
                              colors = cols, show_prog = TRUE, eta = 35,
                              parallel_cores = num_cores_param)
#> It: 1; obj: 1.139e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 3.909e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.139e+01
#> It: 100; obj: 1.108e+01; abs: 2.456e+00; rel: 1.815e-01; norm: 8.051e-01; mom: 5.326e+00;
#> best it: 10; best obj: 8.879e+00
#> It: 200; obj: 2.585e-01; abs: 8.441e-04; rel: 3.254e-03; norm: 1.173e-01; mom: 1.287e+00;
#> best it: 154; best obj: 2.559e-01
#> It: 300; obj: 1.748e-01; abs: 4.036e-04; rel: 2.314e-03; norm: 4.303e-02; mom: 6.602e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 400; obj: 1.751e-01; abs: 2.333e-04; rel: 1.334e-03; norm: 4.422e-02; mom: 6.678e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 500; obj: 1.750e-01; abs: 3.240e-04; rel: 1.855e-03; norm: 4.393e-02; mom: 6.662e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 600; obj: 1.750e-01; abs: 2.988e-04; rel: 1.710e-03; norm: 4.399e-02; mom: 6.636e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 700; obj: 1.751e-01; abs: 3.612e-04; rel: 2.067e-03; norm: 4.400e-02; mom: 6.645e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 800; obj: 1.751e-01; abs: 3.330e-04; rel: 1.906e-03; norm: 4.413e-02; mom: 6.652e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 900; obj: 1.750e-01; abs: 2.834e-04; rel: 1.622e-03; norm: 4.411e-02; mom: 6.665e-01;
#> best it: 274; best obj: 1.736e-01
#> It: 1000; obj: 1.750e-01; abs: 3.044e-04; rel: 1.743e-03; norm: 4.402e-02; mom: 6.649e-01;
#> best it: 274; best obj: 1.736e-01
```

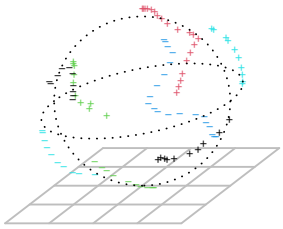
Iteration 1



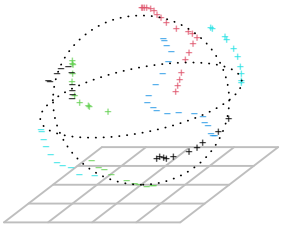
Iteration 200



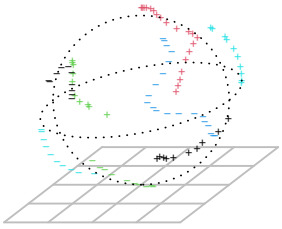
Iteration 400



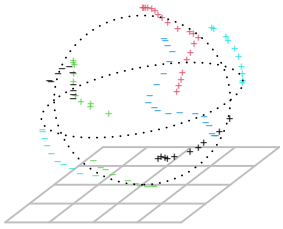
Iteration 600



Iteration 800

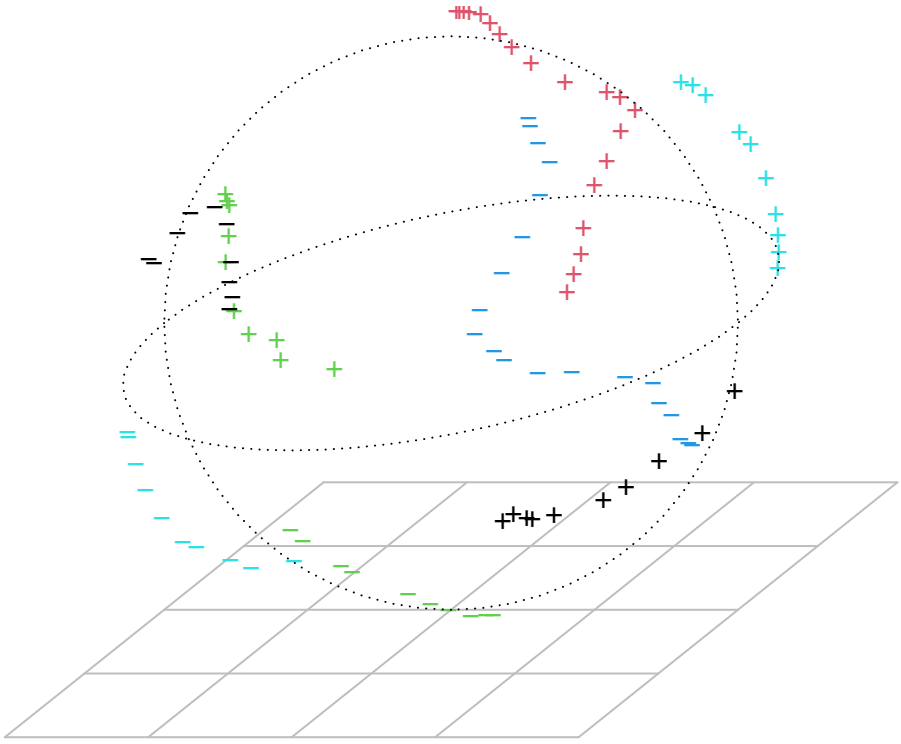


Iteration 1000



In case of the reduced dimension $d = 2$, we can confirm the results obtained in the circumference. Let's see these results on the sphere.

```
Y <- psc_sne_res_22_neg$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



$(\mathbb{S}^1)^{100}$

Generating samples that are uniform and independent in all 's except in the first five, where two groups are defined by two antipodal vMF distributions at north/south

A medium concentration (not massive, so that one has to rely on the combined information of the first ones).

```
r1 <- 5
r2 <- 95
p <- 1
n <- 100
kappa <- 1
gen_data <- rbinom(n = n, size = n, prob = 0.5)
x_vMF_5 <- sapply(seq_len(r1), function(k1) {
  data_vMF <- lapply(seq_len(n), function(i) {
    if (gen_data[i] <= n / 2) {
      rotasym::r_vMF(n = 1, mu = c(0, 1), kappa = kappa)
    } else {
```

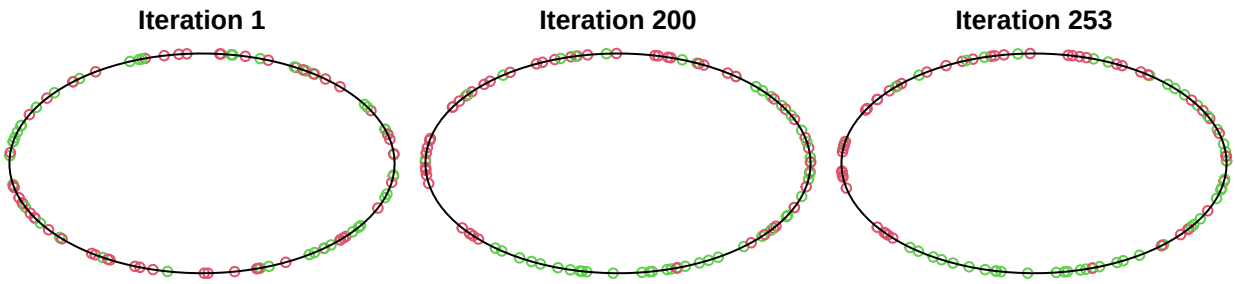
```
      rotasym::r_vMF(n = 1, mu = c(0, -1), kappa = kappa)
    }
  })
  do.call(rbind, data_vMF)
}, simplify = "array")
x_uniform_95 <- sphunif::r_unif_sph(n = n, p = p + 1, M = r2)
x_s1_100 <- abind(x_vMF_5, x_uniform_95, along = 3)
```

Let's calculate the rho based on a fixed perplexity of 30.

```
rho_30_s1_100 <- rho_optim_bst(x = x_s1_100, perp_fixed = 30,
                               num_cores = num_cores_param)
#> Time difference of 40.54312 secs
```

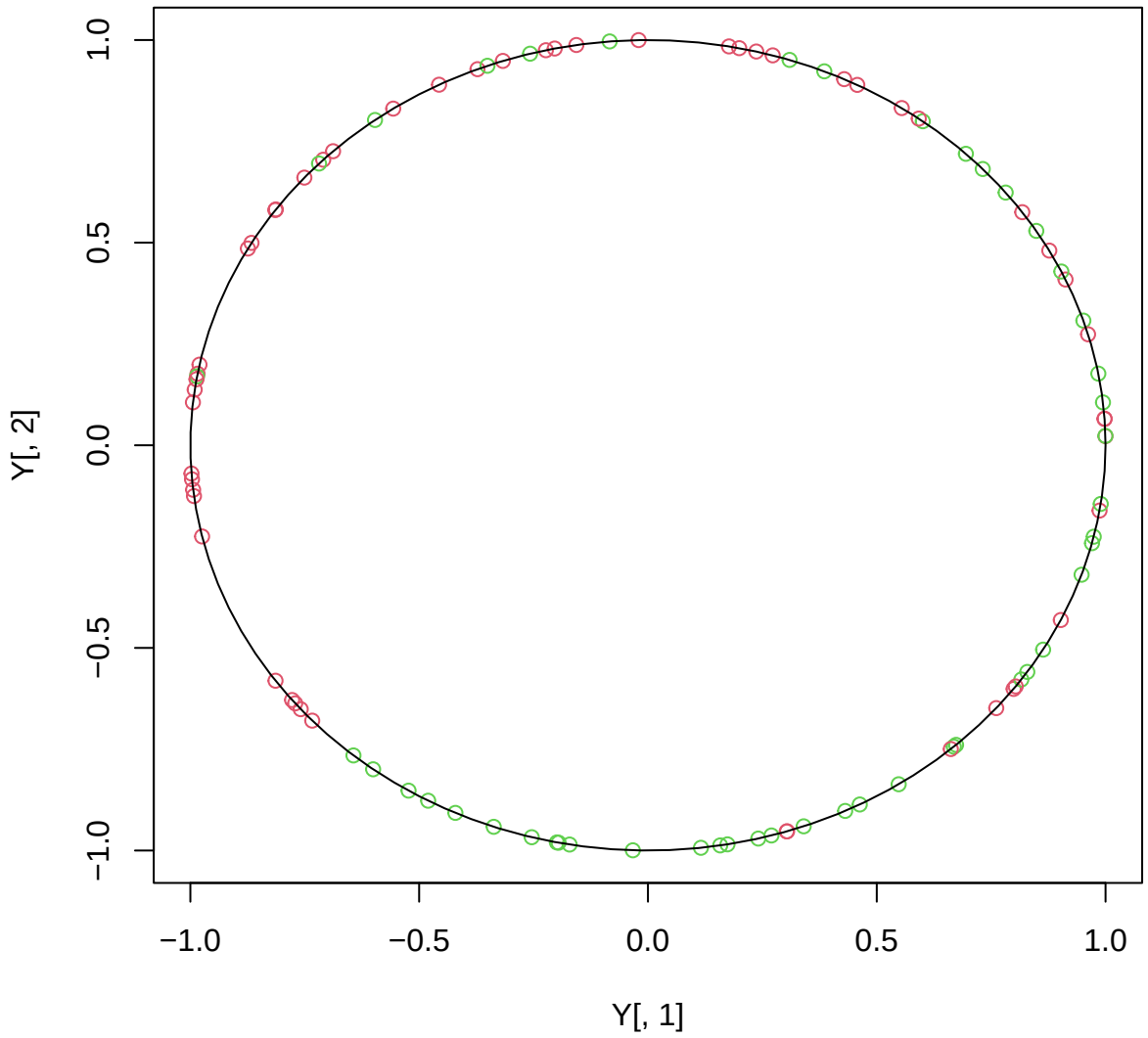
First, let's reduce the dimension to the circumference:

```
cols <- ifelse(gen_data <= n / 2, 2, 3)
psc_sne_res_31 <- psc_sne(X = x_s1_100, d = 1, rho_psc_list = rho_30_s1_100,
                          colors = cols, show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.108e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.443e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.108e+01
#> It: 100; obj: 1.052e+01; abs: 7.090e-01; rel: 6.316e-02; norm: 5.522e-01; mom: 3.521e+00;
#> best it: 7; best obj: 1.009e+01
#> It: 200; obj: 9.977e-01; abs: 4.695e-05; rel: 4.705e-05; norm: 9.854e-04; mom: 2.272e-02;
#> best it: 200; best obj: 9.977e-01
#> It: 253; obj: 9.972e-01; abs: 4.215e-11; rel: 4.227e-11; norm: 7.467e-07; mom: 6.189e-04;
#> best it: 252; best obj: 9.972e-01
```



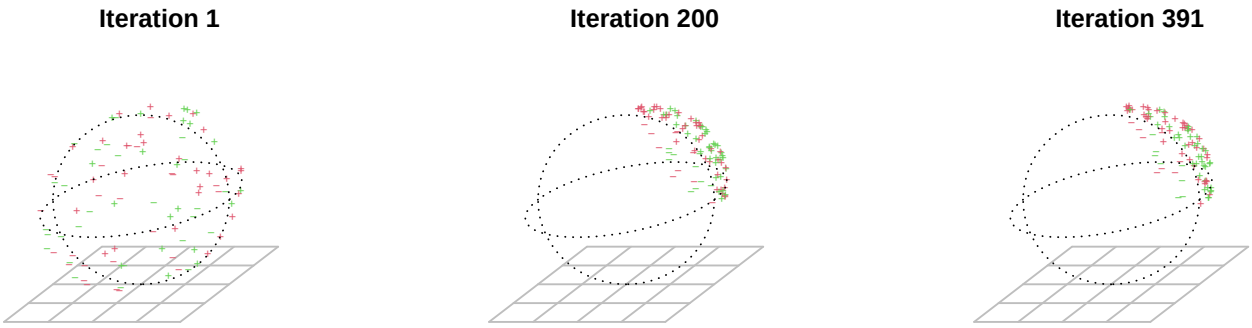
The best result related with the smallest value of the objective function is:

```
Y <- psc_sne_res_31$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1),
     ylim = c(-1, 1))

polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```



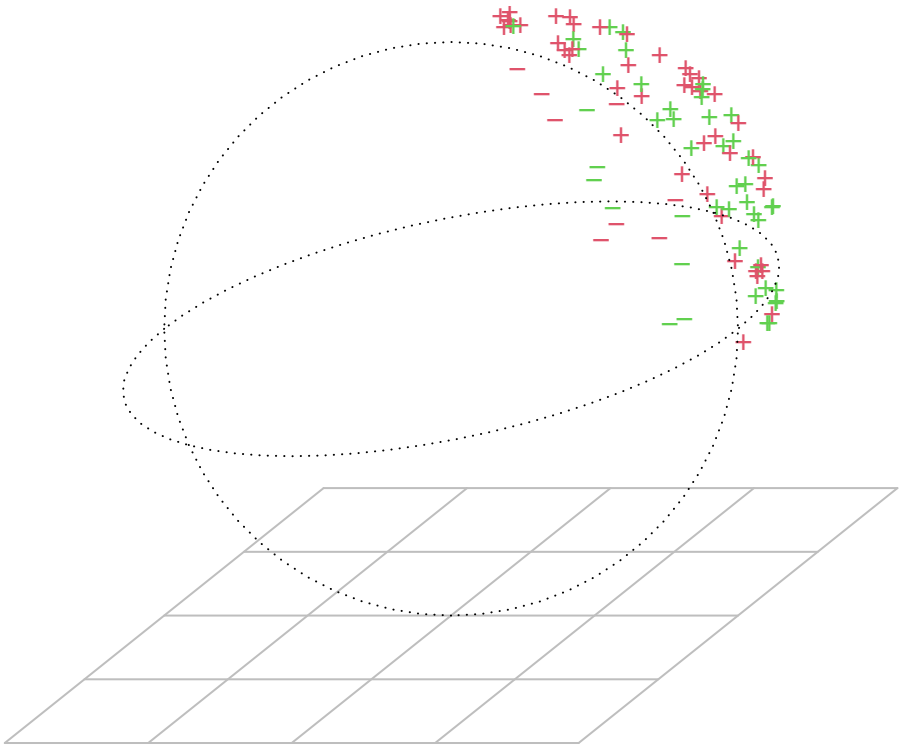
We can see the two different antipodal groups defined in the first five circumferences so the psc-sne algorithm can detect these von Misses-Fisher distributions even with the noise added with the remainder 95 uniform distributions onto the circumference. Later on, let's reduce the dimension to the sphere:

```
psc_sne_res_32 <- psc_sne(X = x_s1_100, d = 2, rho_psc_list = rho_30_s1_100,
                          colors = cols, show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.157e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 3.621e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.157e+01
#> It: 100; obj: 1.401e+01; abs: 3.158e+00; rel: 2.910e-01; norm: 1.143e+00; mom: 4.568e+00;
#> best it: 55; best obj: 1.059e+01
#> It: 200; obj: 9.552e-01; abs: 7.470e-05; rel: 7.821e-05; norm: 2.845e-02; mom: 2.339e-01;
#> best it: 129; best obj: 9.534e-01
#> It: 300; obj: 9.516e-01; abs: 2.732e-05; rel: 2.870e-05; norm: 5.635e-04; mom: 3.206e-02;
#> best it: 300; best obj: 9.516e-01
#> It: 391; obj: 9.487e-01; abs: 8.938e-12; rel: 9.421e-12; norm: 8.281e-07; mom: 4.271e-05;
#> best it: 391; best obj: 9.487e-01
```



In this case, results show two groups antipodal in the sphere. There are some not well identified points but this is something expected since there are 95 points distributed uniformly. Let's visualize the results on the sphere:

```
Y <- psc_sne_res_32$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



A path in $(\mathbb{S}^1)^{100}$ indexed by time.

For example, generated by

$$\theta_i(t) = (\alpha_i + 2\pi k_i t) \mod 2\pi, \quad t \in [0,1], k_i \in \mathbb{Z}, \alpha_i \in [0, 2\pi).$$

These are wrapped straight lines with different slopes and starting points.

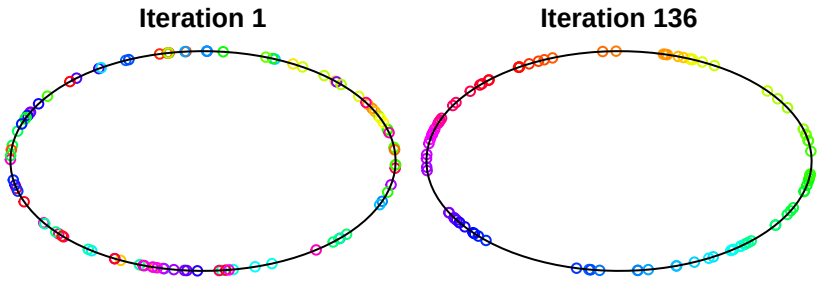
```
n <- 100
r <- 100
x_s1_100_path <- r_path_s1r(n = n, r = r)
cols <- rainbow(n, alpha = 1)
# Change the order of the data
indexes <- sample(1:n)
x_s1_100_path <- x_s1_100_path[indexes, , ]
cols <- cols[indexes]
```

Now that we have created the data, we can calculate the rho based on a fixed perplexity.

```
rho_30_s1_100_path <- rho_optim_bst(x = x_s1_100_path, perp_fixed = 30,
                                    num_cores = num_cores_param)
#> Time difference of 42.04632 secs
```

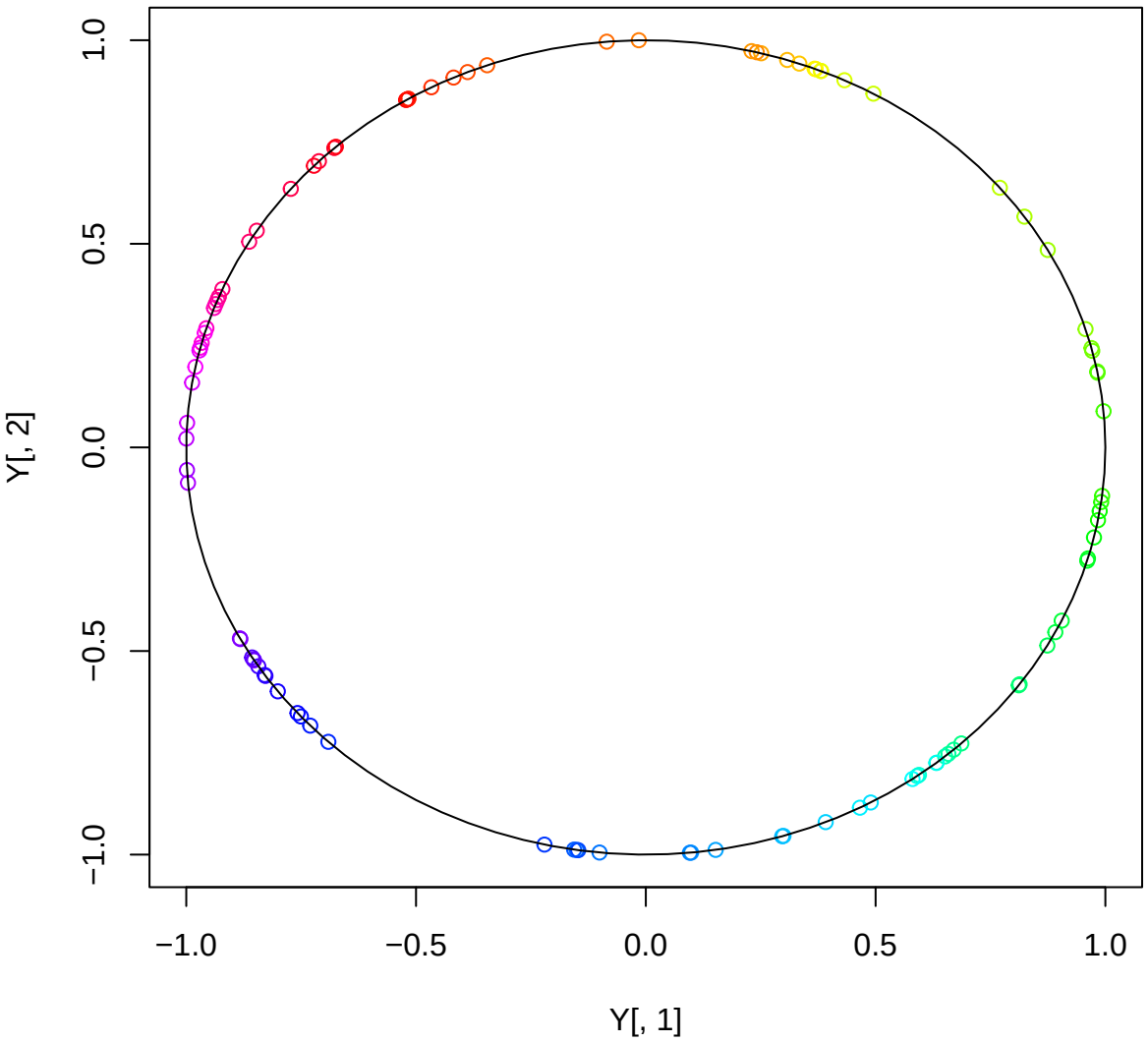
First, let's reduce the dimension to the circumference:

```
psc_sne_res_41 <- psc_sne(X = x_s1_100_path, d = 1,
                          rho_psc_list = rho_30_s1_100_path, colors = cols,
                          show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.058e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.610e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.058e+01
#> Warning in selectChildren(ac[!fin], -1): error 'No child processes' in select
#> It: 100; obj: 7.843e+00; abs: 2.566e-02; rel: 3.282e-03; norm: 4.360e-01; mom: 3.047e+00;
#> best it: 19; best obj: 7.751e+00
#> It: 136; obj: 3.587e-01; abs: 3.543e-12; rel: 9.878e-12; norm: 8.144e-07; mom: 1.369e-05;
#> best it: 136; best obj: 3.587e-01
```



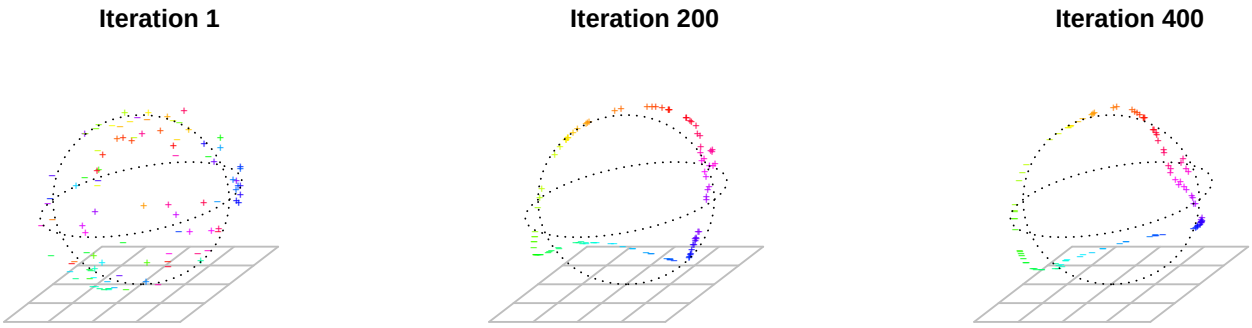
The best result related with the smallest value of the objective function is:

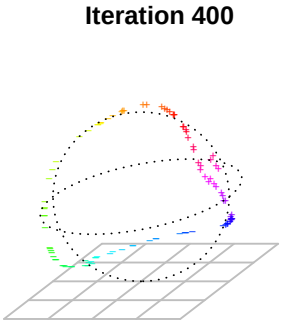
```
Y <- psc_sne_res_41$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1),
     ylim = c(-1, 1))

polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```



We can see that the points in the circumference are more or less well arranged. Nevertheless, it looks like this reduced dimension is not appropriate since results are not as satisfactory as other cases. Let's see if results onto the sphere are better:

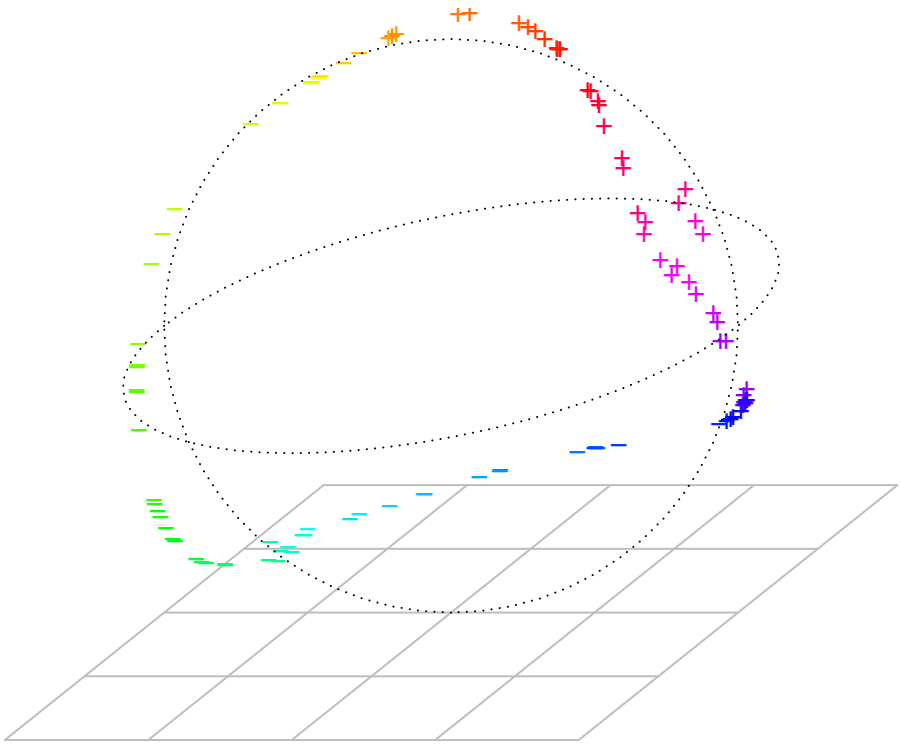
```
psc_sne_res_42 <- psc_sne(X = x_s1_100_path, d = 2,
                          rho_psc_list = rho_30_s1_100_path, colors = cols,
                          show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.116e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 3.055e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.116e+01
#> It: 100; obj: 8.923e+00; abs: 7.172e-01; rel: 7.439e-02; norm: 9.398e-01; mom: 4.863e+00;
#> best it: 3; best obj: 8.196e+00
#> It: 200; obj: 4.962e-02; abs: 1.350e-05; rel: 2.720e-04; norm: 5.199e-04; mom: 1.295e-02;
#> best it: 200; best obj: 4.962e-02
#> It: 300; obj: 4.833e-02; abs: 3.856e-06; rel: 7.979e-05; norm: 1.653e-04; mom: 1.964e-02;
#> best it: 300; best obj: 4.833e-02
#> It: 400; obj: 4.826e-02; abs: 1.010e-10; rel: 2.092e-09; norm: 9.509e-07; mom: 1.194e-04;
#> best it: 400; best obj: 4.826e-02
#> It: 400; obj: 4.826e-02; abs: 1.010e-10; rel: 2.092e-09; norm: 9.509e-07; mom: 1.194e-04;
#> best it: 400; best obj: 4.826e-02
```





In this case, results are well sorted along the sphere since the gradient of the rainbow colors are well defined. Let's visualize the results in the interactive sphere:

```
Y <- psc_sne_res_42$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



$(\mathbb{S}^2)^{100}$

Generate samples that are uniform and independent in all $\mathbb{S}^2$ except in the first five, where there is a common time-indexed path given by

$$
\begin{aligned}
x_1(t) &= \text{equator}, \\
x_2(t) &= \text{small-circle-rotated-1}, \\
x_3(t) &= \text{small-circle-rotated-2}, \\
x_4(t) &= \text{spherical-spiral-1}, \\
x_5(t) &= \text{spherical-spiral-2}.
\end{aligned}
$$

Basically, in this scenario we are generating 95 spheres, $\mathbb{S}^2$, that follows all of them a uniform distribution. The remainder spheres are a small circle distribution located in the equator, two small circles rotated randomly and two rotated spiral distribution.

```
n <- 100
r1 <- 1
r2 <- 2
r3 <- 2
r4 <- 95
# Calculate the north pole of the sphere
north_pole <- cbind(c(0, 0, 1))
# small circle in the equator (1 sample)
samp_1 <- r_path_s2r(n = n, r = r1, sigma = 0.08, Theta = north_pole)
# small circle rotated 1 and 2 (2 samples)
samp_2_3 <- r_path_s2r(n = n, r = r2, sigma = 0.1)
# spherical spiral 1 and 2 (2 samples)
samp_4_5 <- r_path_s2r(n = n, r = r3, c = 3, spiral = TRUE, sigma = 0.01)
```

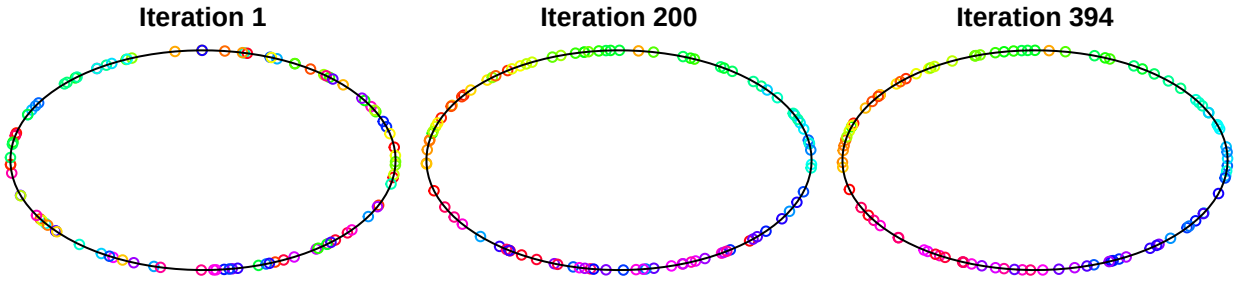
```
# Data following an uniform distribution on the sphere (95 samples)
samp_95 <- r_unif_sph(n = n, p = 3, M = r4)
# Join the data by the third dimension
x_s2_100 <- abind(samp_1, samp_2_3, samp_4_5, samp_95, along = 3)
# Create rainbow colors, alpha is the level of opacity
cols <- rainbow(n, alpha = 1)
# Change the order of the data
indexes <- sample(1:n)
x_s2_100 <- x_s2_100[indexes, , ]
cols <- cols[indexes]
```

Now that we have created the artificial dataset, we can calculate the rho based on a fixed perplexity.

```
rho_30_s2_100 <- rho_optim_bst(x = x_s2_100, perp_fixed = 30,
                               num_cores = num_cores_param)
#> Time difference of 41.95634 secs
```

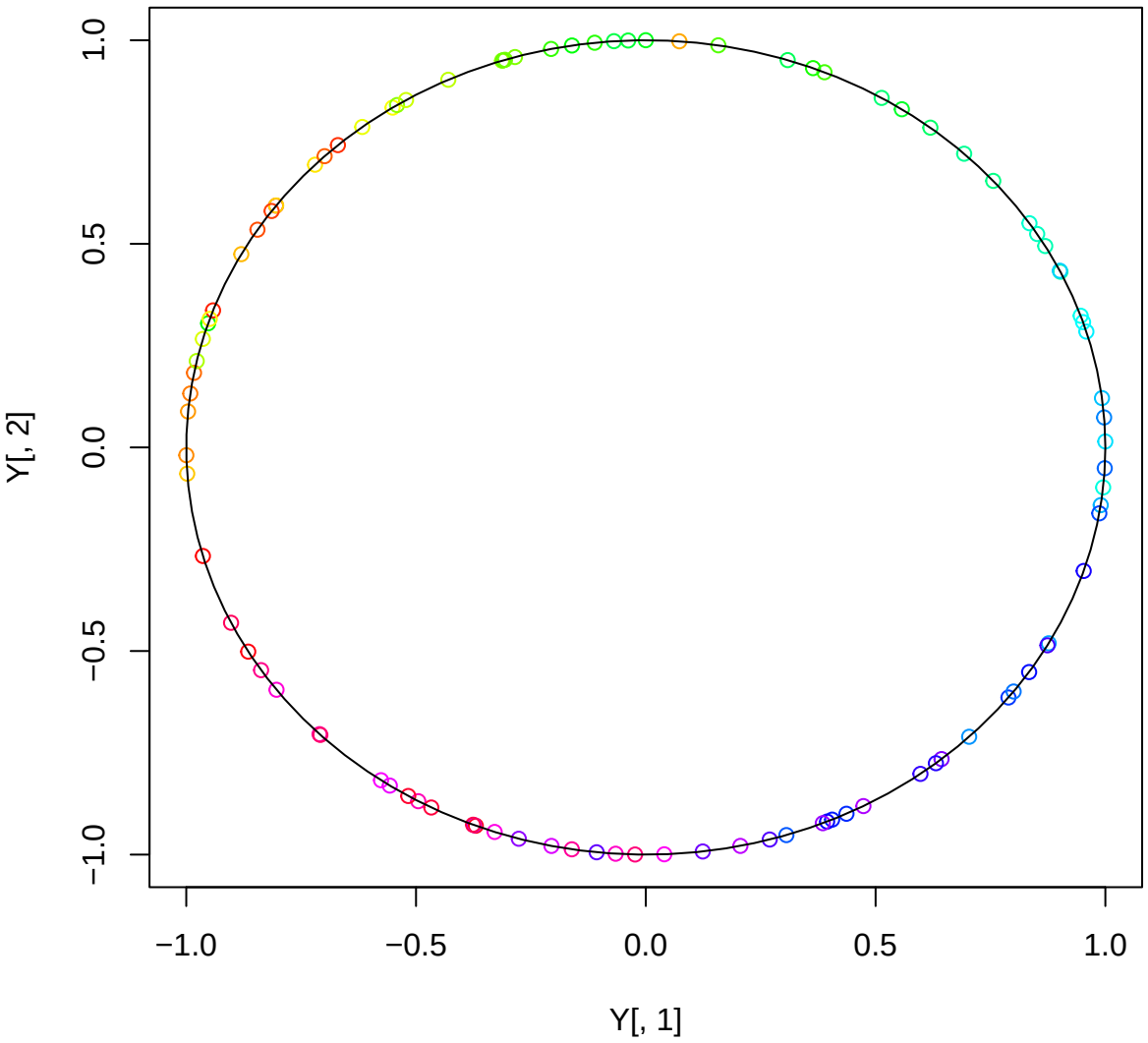
First, let's reduce the dimension to the circumference:

```
psc_sne_res_51 <- psc_sne(X = x_s2_100, d = 1, rho_psc_list = rho_30_s2_100,
                          colors = cols, show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.088e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 1.723e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.088e+01
#> It: 100; obj: 1.053e+01; abs: 7.743e-01; rel: 6.851e-02; norm: 5.596e-01; mom: 3.568e+00;
#> best it: 12; best obj: 9.810e+00
#> It: 200; obj: 8.800e-01; abs: 1.499e-06; rel: 1.704e-06; norm: 1.680e-04; mom: 5.171e-03;
#> best it: 200; best obj: 8.800e-01
#> It: 300; obj: 8.711e-01; abs: 3.419e-05; rel: 3.925e-05; norm: 1.181e-03; mom: 5.710e-02;
#> best it: 300; best obj: 8.711e-01
#> It: 394; obj: 8.575e-01; abs: 1.590e-12; rel: 1.854e-12; norm: 8.973e-07; mom: 4.327e-05;
#> best it: 394; best obj: 8.575e-01
```



The best result related with the smallest value of the objective function is:

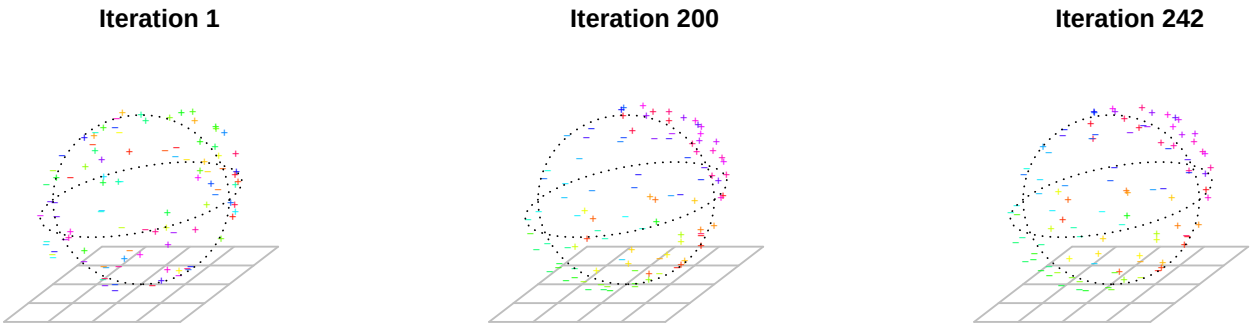
```
Y <- psc_sne_res_51$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1),
     ylim = c(-1, 1))

polygon(x = cos(seq(0, 2 * pi, length.out = 100)),
        y = sin(seq(0, 2 * pi, length.out = 100)))
```



Results show more or less the time-index structure of the first 5 spheres even with the noise of the remainder spheres. Nevertheless, it is not possible to distinguish any shape.

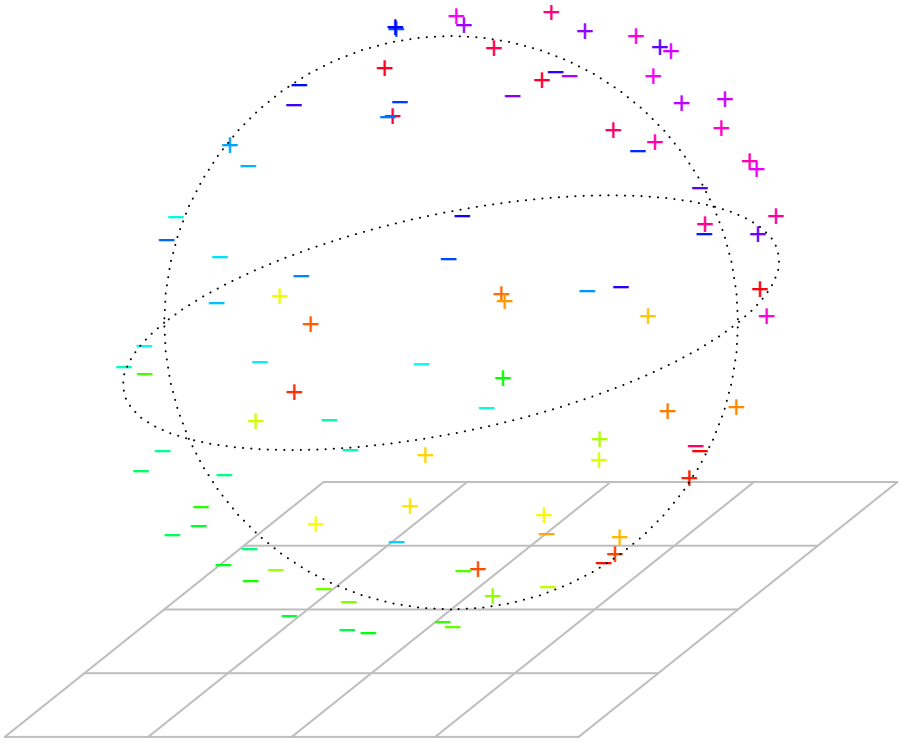
Afterwards, let's get the results on the sphere:

```
psc_sne_res_52 <- psc_sne(X = x_s2_100, d = 2, rho_psc_list = rho_30_s2_100,
                          colors = cols, show_prog = TRUE, eta = 25,
                          parallel_cores = num_cores_param)
#> It: 1; obj: 1.178e+01; abs: 0.000e+00; rel: 0.000e+00; norm: 3.393e-01; mom: 0.000e+00;
#> best it: 1; best obj: 1.178e+01
#> It: 100; obj: 1.081e+01; abs: 3.392e+00; rel: 2.388e-01; norm: 9.245e-01; mom: 5.269e+00;
#> best it: 3; best obj: 1.058e+01
#> It: 200; obj: 7.953e-01; abs: 2.388e-03; rel: 2.993e-03; norm: 7.234e-03; mom: 1.697e-01;
#> best it: 200; best obj: 7.953e-01
#> It: 242; obj: 7.884e-01; abs: 4.530e-11; rel: 5.746e-11; norm: 9.188e-07; mom: 2.906e-05;
#> best it: 242; best obj: 7.884e-01
```



Let's visualize the results on the sphere:

```
Y <- psc_sne_res_52$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c('+', '-')[ifelse(sign(Y[, 2]) == 1, 1, 2)]
)
sd3$points3d(DirStats::to_sph(th = meridian[, 1], ph = meridian[, 2]),
             type = "l", lty = 3)
sd3$points3d(DirStats::to_sph(th = equator[, 1], ph = equator[, 2]),
             type = "l", lty = 3)
```



In this case, the resultant data show again the insights obtained in the reduced dimension when $d = 1$. It is not possible to see a clear shape, for example, a small circle or a spiral. That makes sense since it is not possible to distinguish any clear cluster since there are lot of noise and other distributions involved.