High-dimensional Numerical Examples

Luis Ángel Rodríguez García

2022-09-18

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

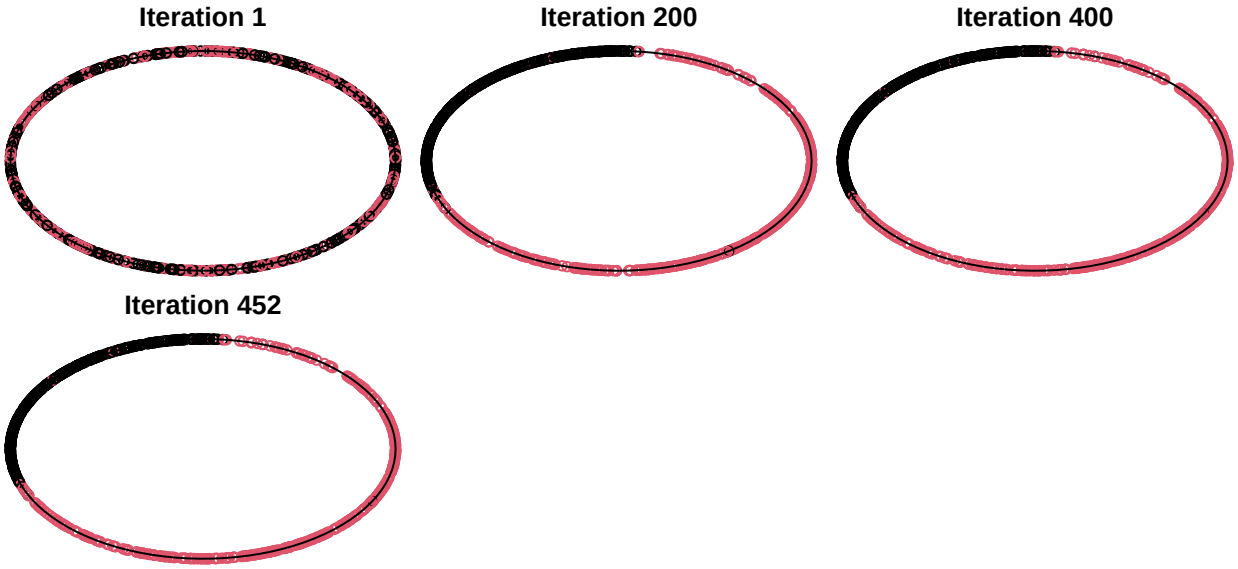
```
sc_unif_mix <- function(n, p, w_sc, w_unif, kappa = 50) {
  if (w_sc + w_unif != 1) {
    stop("w_sc and w_unif must sum 1")
  }
  n1 <- rbinom(1, n, w_sc)
  n2 <- n - n1
  r_1 <- sphunif::r_alt(n = n1, p = p, alt = "SC", kappa = kappa)
  r_2 <- sphunif::r_unif_sph(n = n2, p = p)
  data <- abind(r_1, 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 <- 800
p <- 101
w_sc <- 0.5
w_unif <- 0.5
kappa <- 1000
set.seed(42)
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 9.196822 secs
```

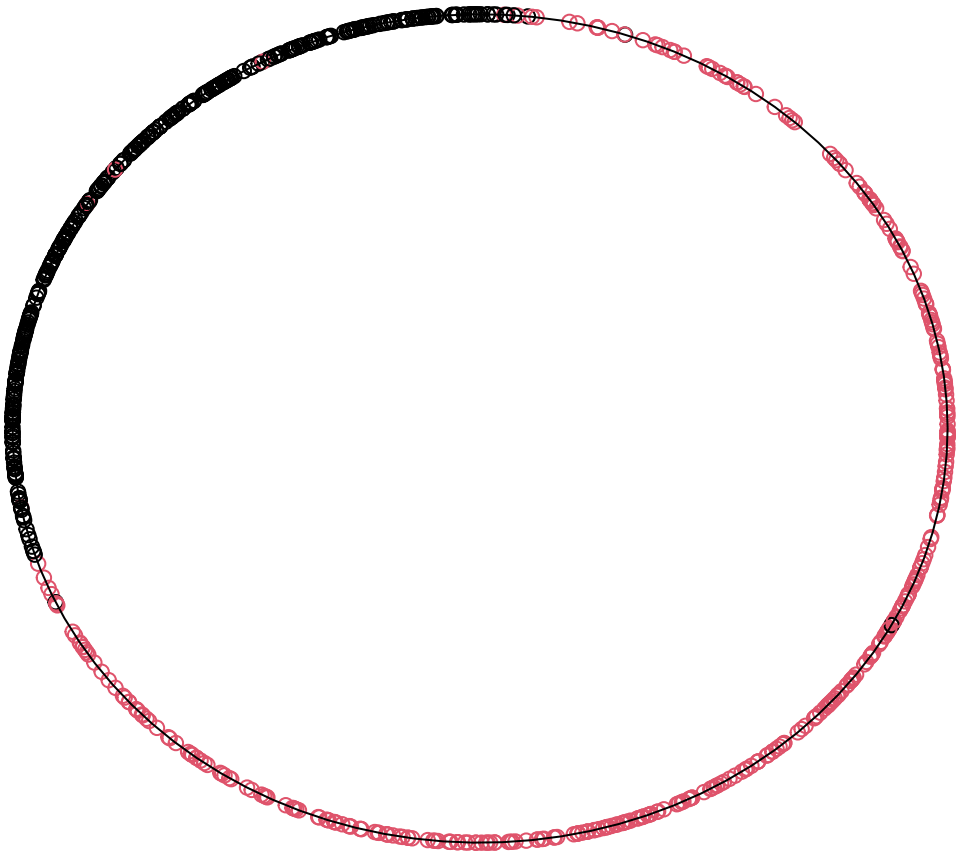
Reduction to $d = 1$ The next thing is to run the `psc-sne` algorithm with the parameter $d = 1$:

```
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,
                         eta = 100, parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.88e+01 (best: 1.88e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 6.9e-02; mom: 0.0e+00
#> It: 100 (best: 13); obj: 1.84e+01 (best: 1.71e+01); abs: 7.4e-08; rel: 4.0e-09; norm: 2.2e-01; mom: 6.3e+00
#> It: 200 (best: 200); obj: 2.68e+00 (best: 2.68e+00); abs: 6.5e-05; rel: 2.4e-05; norm: 5.7e-04; mom: 6.0e-02
#> It: 300 (best: 300); obj: 2.68e+00 (best: 2.68e+00); abs: 4.9e-05; rel: 1.8e-05; norm: 3.4e-04; mom: 1.2e-01
#> It: 400 (best: 400); obj: 2.68e+00 (best: 2.68e+00); abs: 1.7e-07; rel: 6.4e-08; norm: 2.3e-05; mom: 9.8e-03
#> It: 452 (best: 452); obj: 2.68e+00 (best: 2.68e+00); abs: 5.9e-10; rel: 2.2e-10; norm: 9.9e-07; mom: 5.1e-04
#> CONVERGENCE!
```



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

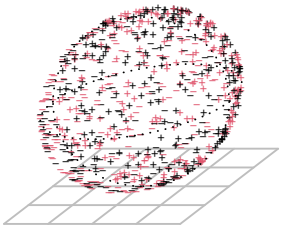
```
Y <- res_pscsne_11$best_Y
plot(Y[, 1], Y[, 2], col = sc_unif_mix_colors, xlim = c(-1, 1),
     ylim = c(-1, 1), axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```



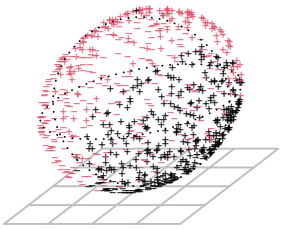
Reduction to $d = 2$ After that, let's execute psc-sne with the parameter $d = 2$:

```
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,
                         eta = 100, parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.97e+01 (best: 1.97e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.4e-01; mom: 0.0e+00
#> It: 100 (best: 8); obj: 2.09e+01 (best: 1.75e+01); abs: 2.0e+00; rel: 1.1e-01; norm: 4.6e-01; mom: 1.1e+01
#> It: 200 (best: 200); obj: 2.44e+00 (best: 2.44e+00); abs: 1.4e-04; rel: 5.9e-05; norm: 8.5e-04; mom: 8.4e-02
#> It: 300 (best: 300); obj: 2.43e+00 (best: 2.43e+00); abs: 5.7e-05; rel: 2.3e-05; norm: 4.0e-04; mom: 1.1e-01
#> It: 400 (best: 400); obj: 2.43e+00 (best: 2.43e+00); abs: 6.4e-08; rel: 2.6e-08; norm: 1.1e-05; mom: 4.5e-03
#> It: 500 (best: 500); obj: 2.43e+00 (best: 2.43e+00); abs: 3.0e-07; rel: 1.2e-07; norm: 2.4e-05; mom: 1.0e-02
#> It: 600 (best: 600); obj: 2.43e+00 (best: 2.43e+00); abs: 5.4e-10; rel: 2.2e-10; norm: 9.8e-07; mom: 4.6e-04
#> It: 600 (best: 600); obj: 2.43e+00 (best: 2.43e+00); abs: 5.4e-10; rel: 2.2e-10; norm: 9.8e-07; mom: 4.6e-04
#> CONVERGENCE!
```

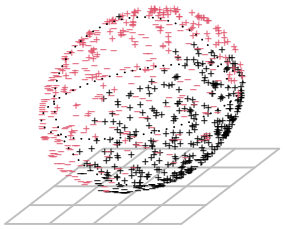
Iteration 1



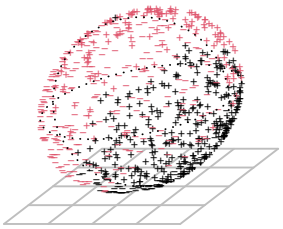
Iteration 200



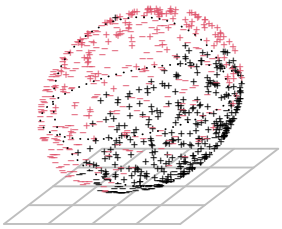
Iteration 400



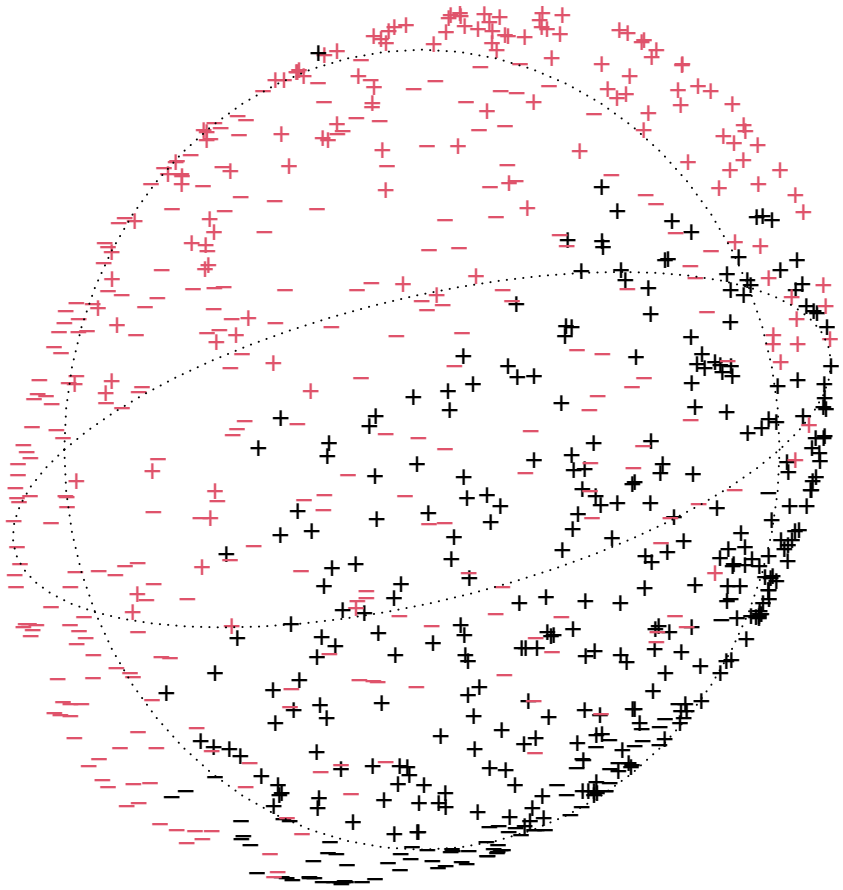
Iteration 600



Iteration 600



```
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$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = sc_unif_mix_colors, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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

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

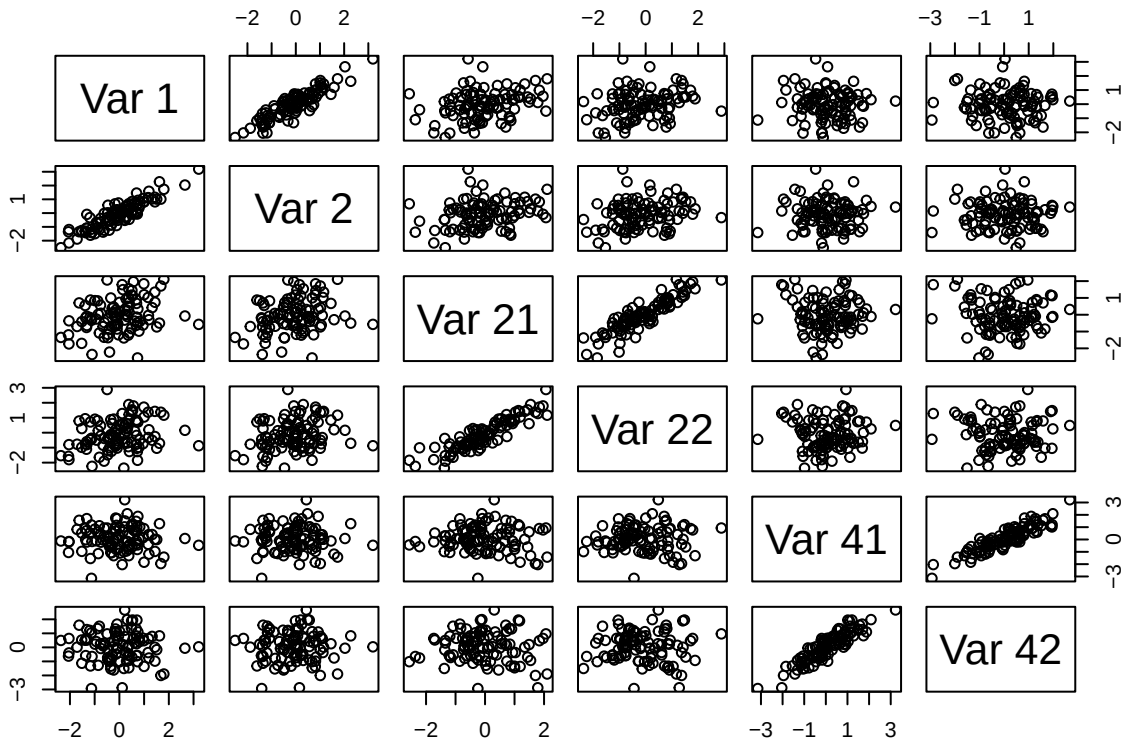
is block-wise diagonal. Take now $n = 200$ observations and center and standardize the features. Then the features now live in $\mathbb{S}^{n-1}$.

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 next to each other, we have shuffled these points so the data-set is disorganized.

```
# Visualize some features
n <- 100
g <- 10
p <- 20
set.seed(42)
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 <- rainbow(g, alpha = 1)
cols <- rep(cols, 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
set.seed(42)
indexes <- sample(1:(g * p))
feat_data <- feat_data[indexes, , , drop = FALSE]
cols <- cols[indexes]
```

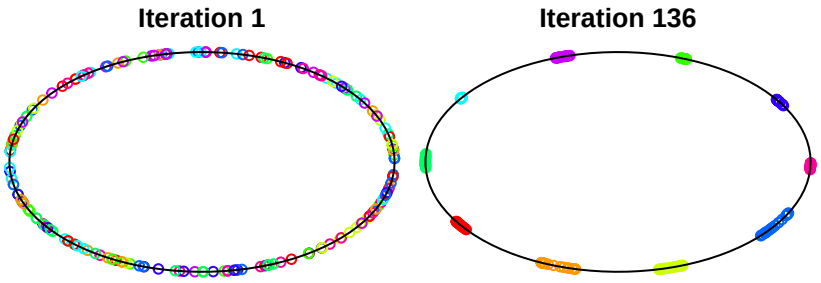
Let's calculate the rho based on a perplexity of 30

```
rho_list <- rho_optim_bst(x = feat_data, perp_fixed = 30,
                          num_cores = num_cores_param)
#> Time difference of 0.7016029 secs
```

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

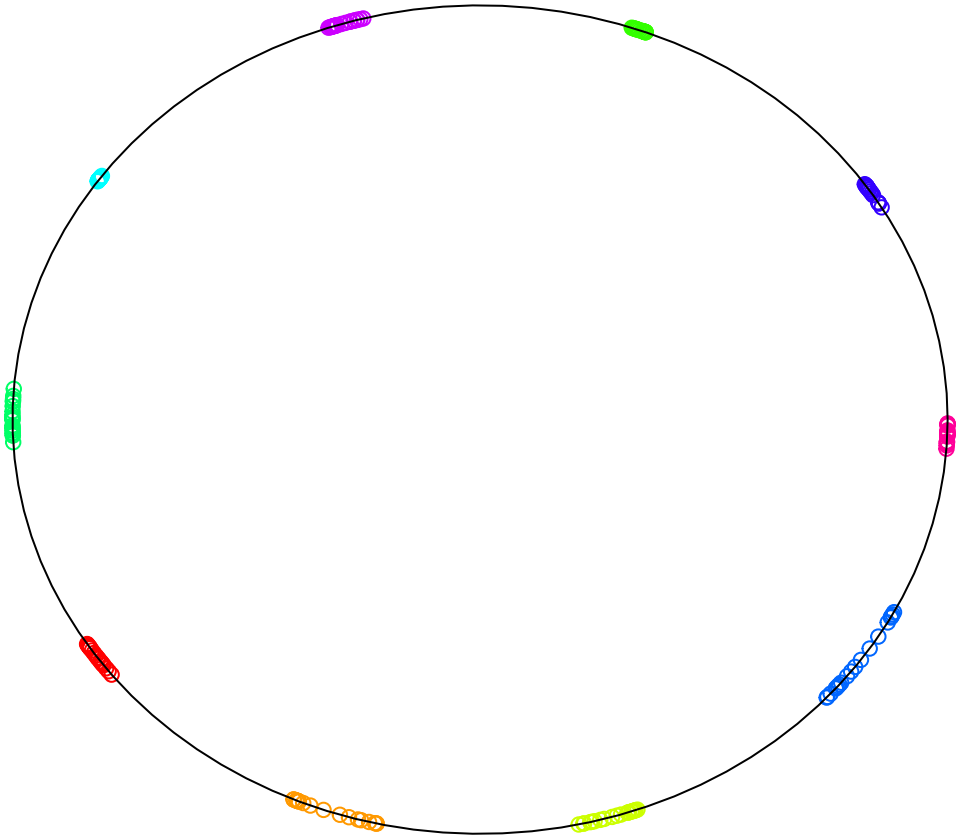
```
res_pscsne_21 <- psc_sne(X = feat_data, d = 1, rho_psc_list = rho_list,
                         colors = cols, show_prog = TRUE,
```

```
                        eta = 35, parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.36e+01 (best: 1.36e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.4e-01; mom: 0.0e+00
#> It: 100 (best: 99); obj: 1.07e+01 (best: 1.07e+01); abs: 3.1e-04; rel: 2.9e-05; norm: 3.6e-01; mom: 3.7e+00
#> It: 136 (best: 136); obj: 1.07e+00 (best: 1.07e+00); abs: 2.0e-12; rel: 1.8e-12; norm: 7.6e-07; mom: 2.3e-05
#> CONVERGENCE!
```



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),
     axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```

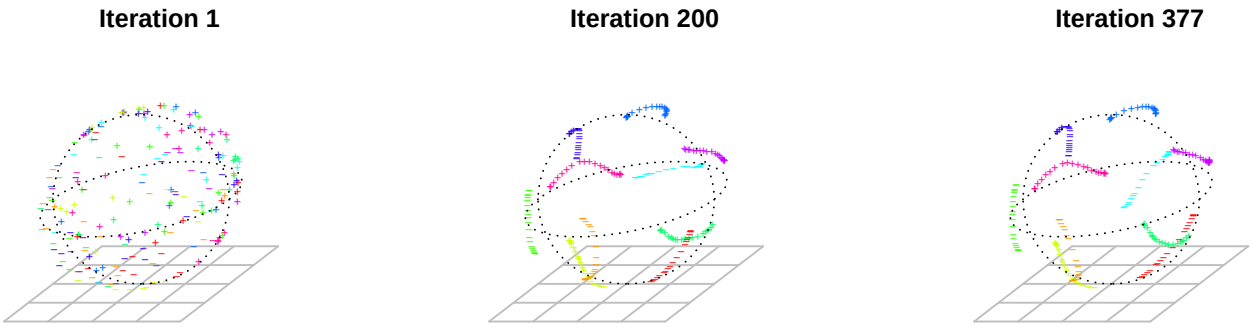


The psc-sne algorithm identifies in the reduced dimension on the circumference the five groups that are in this case strongly positive correlated within each group's variables.

Reduction to $d = 2$ Let's do the same for $d = 2$:

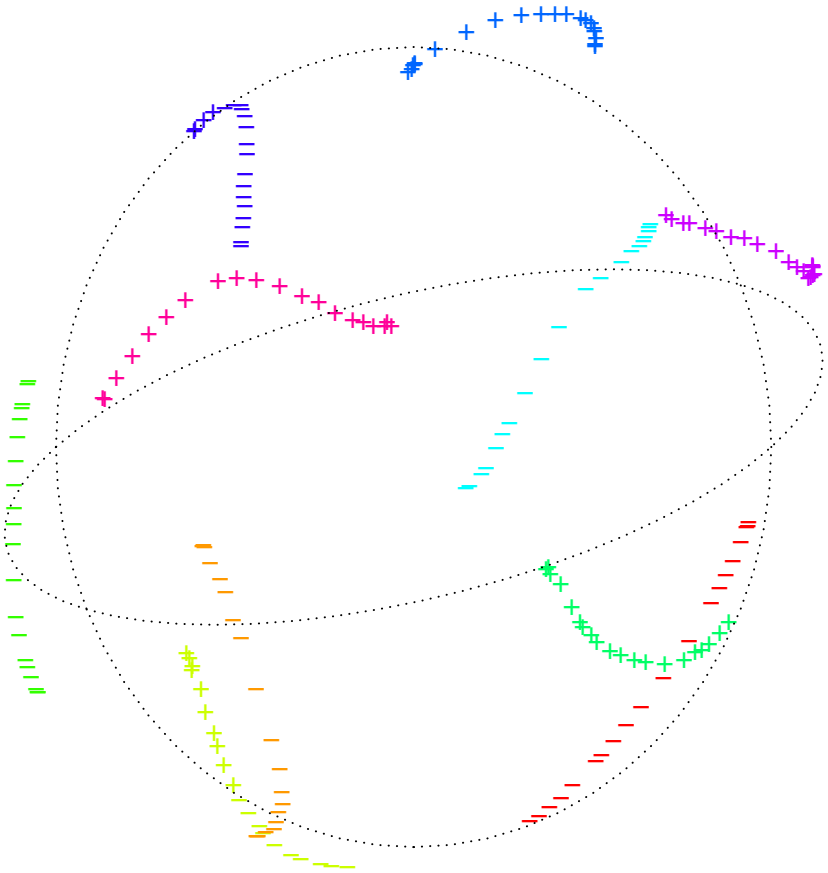
```
psc_sne_res_22 <- psc_sne(feat_data, d = 2, rho_psc_list = rho_list,
                          colors = cols, show_prog = TRUE,
                          eta = 35, parallel_cores = num_cores_param)
```

```
#> It: 1 (best: 1); obj: 1.44e+01 (best: 1.44e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 2.5e-01; mom: 0.0e+00
#> It: 100 (best: 25); obj: 1.08e+01 (best: 1.03e+01); abs: 3.0e-02; rel: 2.7e-03; norm: 7.5e-01; mom: 6.0e+00
#> It: 200 (best: 200); obj: 4.42e-01 (best: 4.42e-01); abs: 2.1e-07; rel: 4.8e-07; norm: 5.5e-05; mom: 1.9e-03
#> It: 300 (best: 300); obj: 4.42e-01 (best: 4.42e-01); abs: 6.7e-07; rel: 1.5e-06; norm: 6.5e-05; mom: 8.1e-03
#> It: 377 (best: 377); obj: 4.42e-01 (best: 4.42e-01); abs: 3.6e-10; rel: 8.1e-10; norm: 8.1e-07; mom: 5.0e-04
#> CONVERGENCE!
```



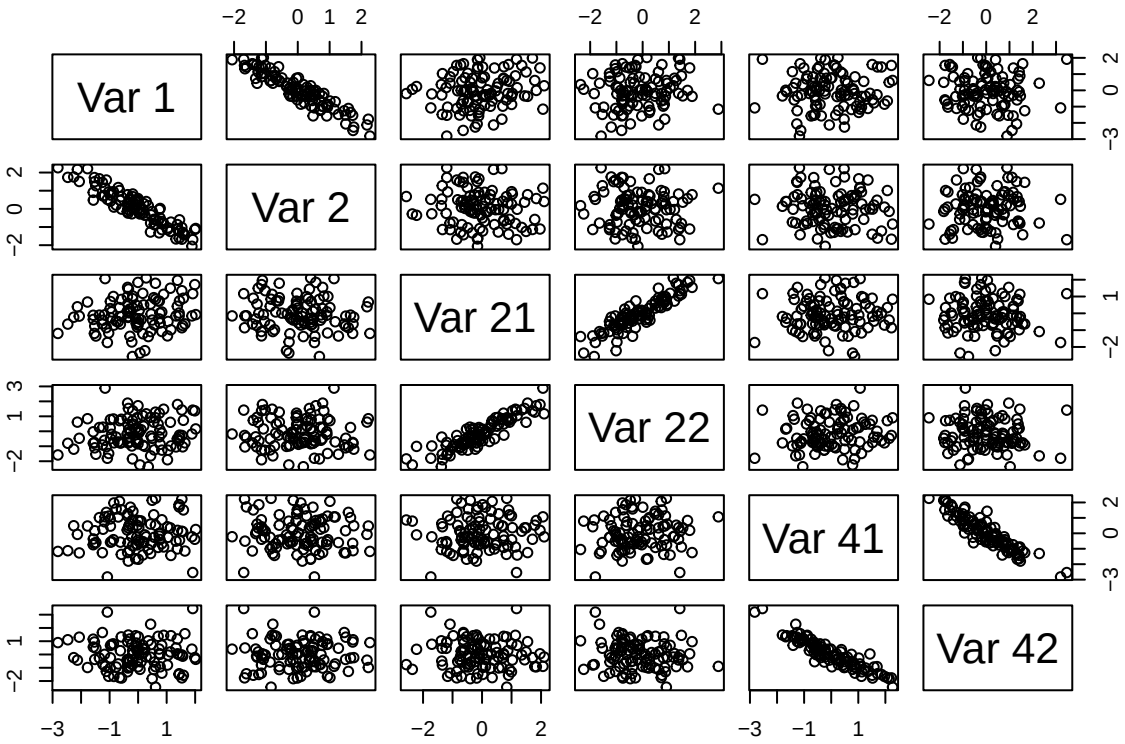
In case of the scores projected onto the sphere, results are good since the five groups are clearly separated along the sphere. Let's see the points in the interactive sphere using the package `rgl`.

```
Y <- psc_sne_res_22$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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 <- 10
p <- 20
rho_values_neg <- rep(c(-0.9, 0.9), times = g)[1:g]
set.seed(42)
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 <- rainbow(g, alpha = 1)
cols <- rep(cols, 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
set.seed(42)
indexes <- sample(1:(g * p))
feat_data_2 <- feat_data_2[indexes, , , drop = FALSE]
cols <- cols[indexes]
```

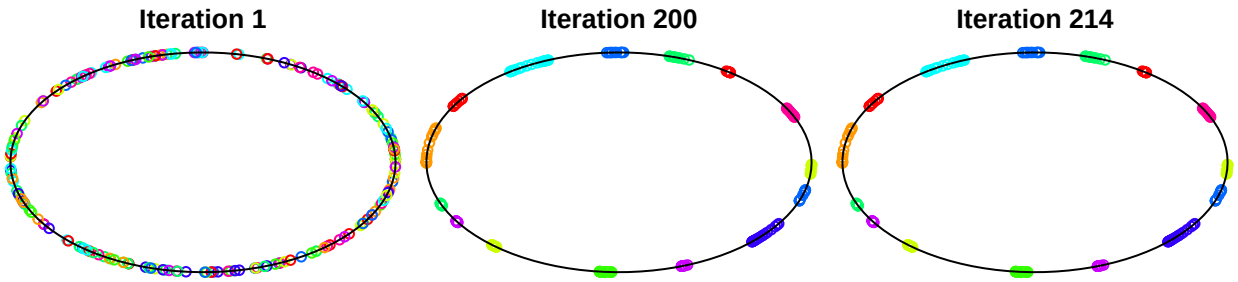
Let's calculate the rho based on a perplexity of 30:

```
rho_list_2 <- rho_optim_bst(x = feat_data_2, perp_fixed = 30,
                            num_cores = num_cores_param)
#> Time difference of 0.7078402 secs
```

Reduction to $d = 1$ Run `psc_sne()` with colors being the groups of variables. First, reduced the data on the circumference.

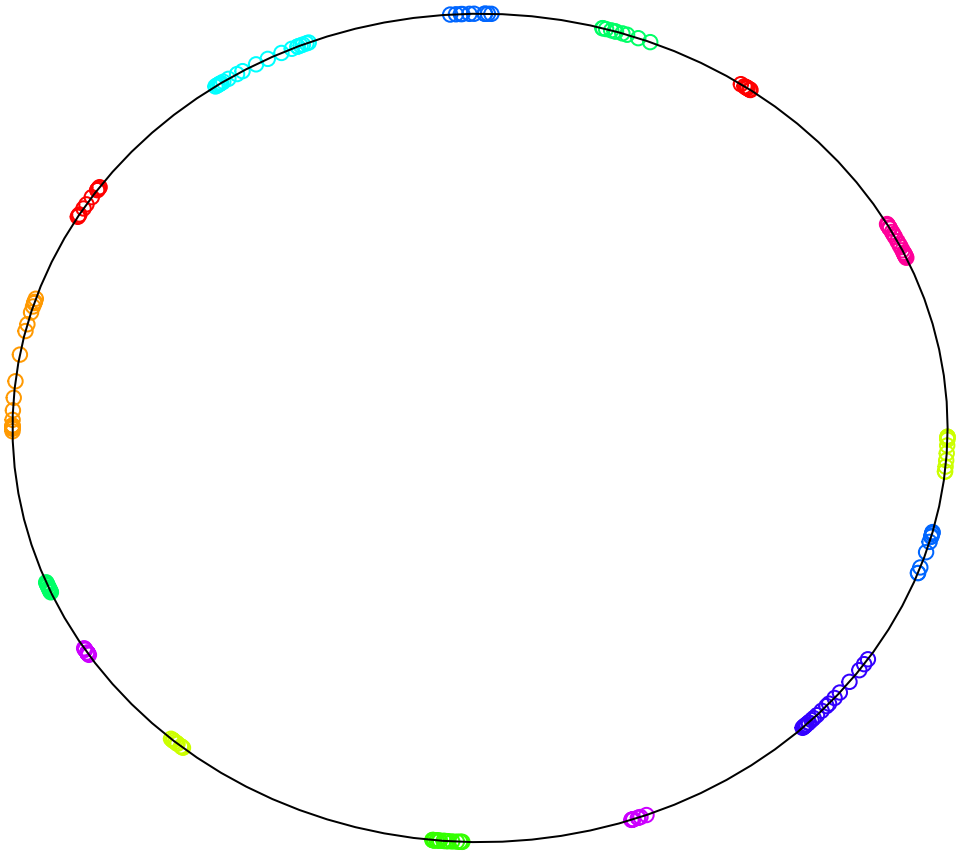
```
res_pscsne_21_neg <- psc_sne(X = feat_data_2, d = 1, rho_psc_list = rho_list_2,
                             colors = cols, show_prog = TRUE, eta = 35,
                             parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.36e+01 (best: 1.36e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.5e-01; mom: 0.0e+00
#> It: 100 (best: 10); obj: 1.34e+01 (best: 1.11e+01); abs: 1.3e-04; rel: 9.4e-06; norm: 3.9e-01; mom: 3.9e+00
#> It: 200 (best: 200); obj: 1.15e+00 (best: 1.15e+00); abs: 3.2e-10; rel: 2.8e-10; norm: 2.1e-06; mom: 8.2e-05
#> It: 214 (best: 214); obj: 1.15e+00 (best: 1.15e+00); abs: 6.8e-11; rel: 6.0e-11; norm: 9.7e-07; mom: 3.8e-05
```

```
#> CONVERGENCE!
```



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),
     axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```



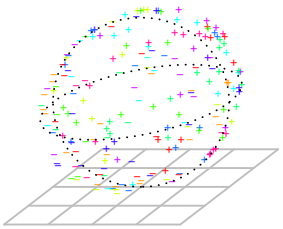
In this case, the negative correlation in some groups has an effect on the results produced by `psc_sne`. Each group with negative correlation is split in two where they look like being distributed more or less antipodal.

Reduction to $d = 2$ Now, let's reduce the data onto the sphere $d = 2$.

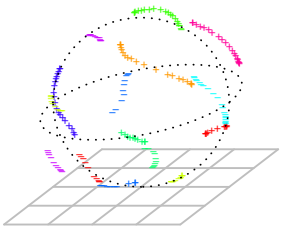
```
psc_sne_res_22_neg <- psc_sne(feat_data_2, d = 2, rho_psc_list = rho_list_2,
                              colors = cols, show_prog = TRUE, eta = 20,
                              parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.45e+01 (best: 1.45e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 2.9e-01; mom: 0.0e+00
#> It: 100 (best: 6); obj: 1.41e+01 (best: 9.94e+00); abs: 7.1e-06; rel: 5.0e-07; norm: 9.0e-01; mom: 4.8e+00
#> It: 200 (best: 200); obj: 5.70e-01 (best: 5.70e-01); abs: 4.6e-05; rel: 8.1e-05; norm: 1.1e-03; mom: 2.1e-02
```

```
#> It: 300 (best: 300); obj: 5.64e-01 (best: 5.64e-01); abs: 8.4e-07; rel: 1.5e-06; norm: 1.5e-04; mom: 1.5e-02
#> It: 400 (best: 400); obj: 5.64e-01 (best: 5.64e-01); abs: 4.2e-05; rel: 7.4e-05; norm: 6.8e-04; mom: 4.8e-02
#> It: 500 (best: 500); obj: 5.56e-01 (best: 5.56e-01); abs: 1.9e-05; rel: 3.4e-05; norm: 4.5e-04; mom: 3.3e-02
#> It: 600 (best: 600); obj: 5.49e-01 (best: 5.49e-01); abs: 9.5e-06; rel: 1.7e-05; norm: 4.1e-04; mom: 3.6e-02
#> It: 686 (best: 686); obj: 5.49e-01 (best: 5.49e-01); abs: 1.2e-10; rel: 2.1e-10; norm: 9.5e-07; mom: 1.1e-04
#> CONVERGENCE!
```

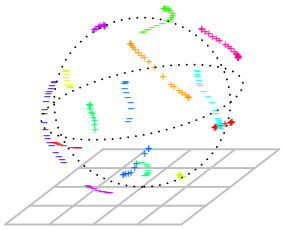
Iteration 1



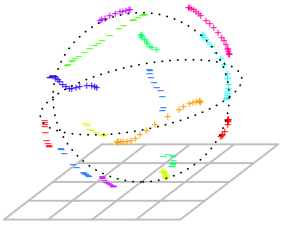
Iteration 200



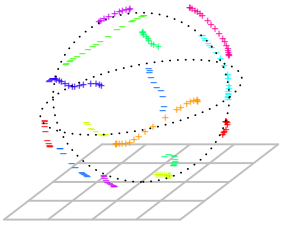
Iteration 400



Iteration 600

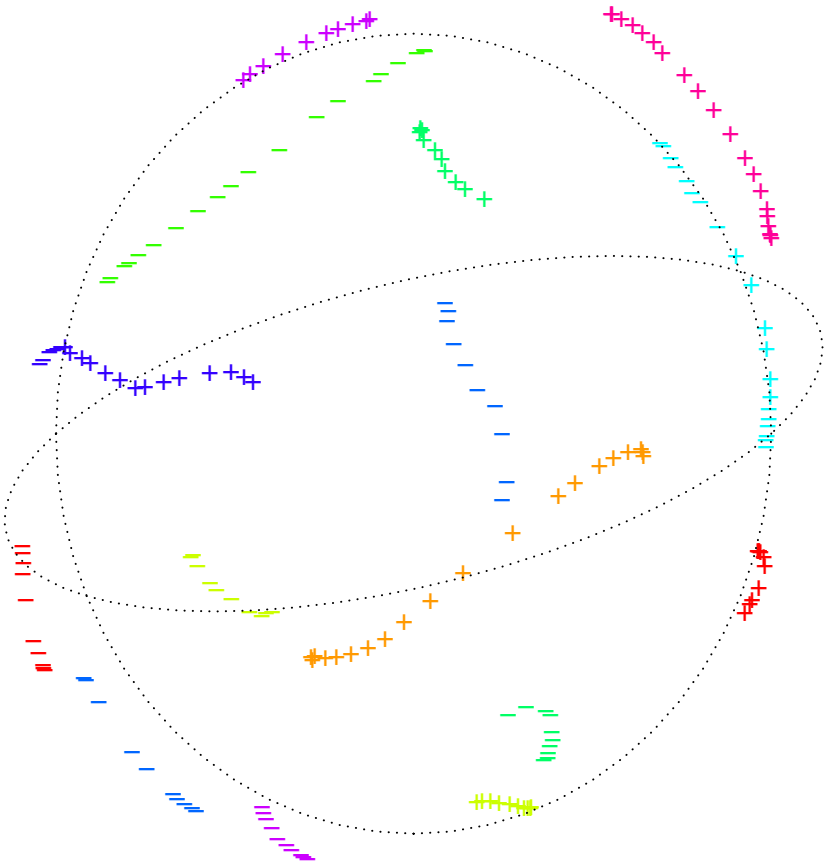


Iteration 686



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$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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 $\mathbb{S}^1$'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).

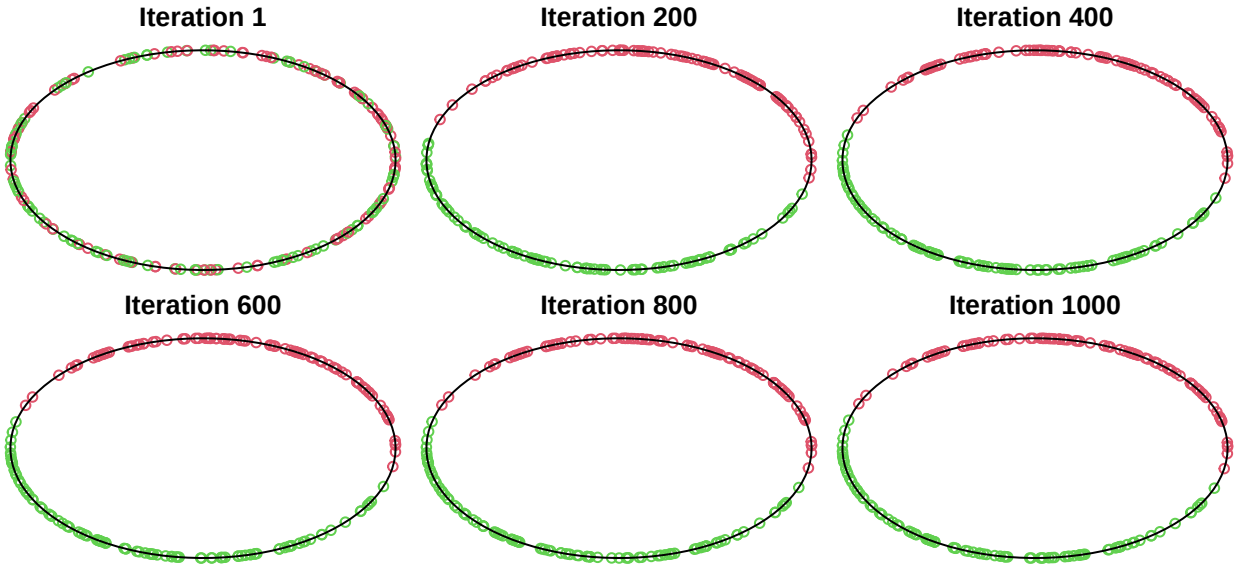
```
r1 <- 5
r2 <- 95
p <- 1
n <- 200
kappa <- 3
set.seed(42)
gen_data <- rbinom(n = n, size = n, prob = 0.5)
x_vMF_5 <- sapply(seq_len(r1), function(k1) {
  data_vMF <- lapply(seq_len(n), function(i) {
    if (gen_data[i] <= n / 2) {
      rotasym::r_vMF(n = 1, mu = c(0, 1), kappa = kappa)
    } else {
      rotasym::r_vMF(n = 1, mu = c(0, -1), kappa = kappa)
    }
  })
  do.call(rbind, data_vMF)
}, simplify = "array")
set.seed(42)
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 35.97735 secs
```

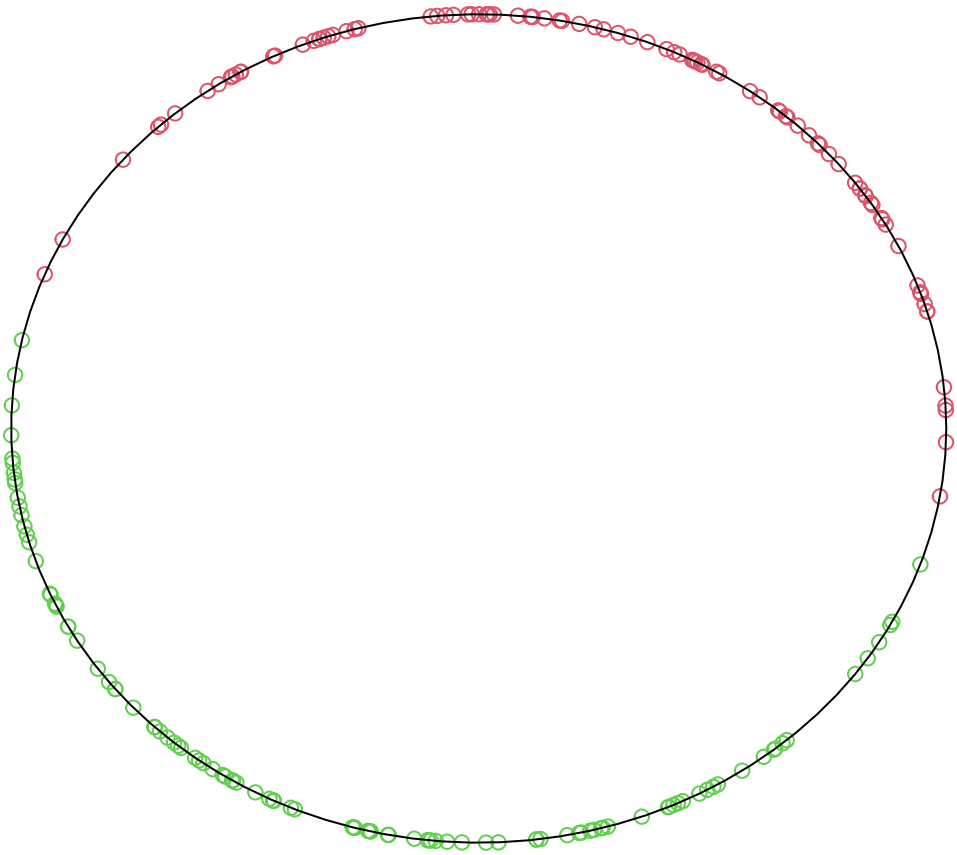
Reduction to $d = 1$ First, let's reduce the dimension to the circumference:

```
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 = 110,
                          parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.37e+01 (best: 1.37e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.2e-01; mom: 0.0e+00
#> It: 100 (best: 6); obj: 1.47e+01 (best: 1.26e+01); abs: 1.3e+00; rel: 9.5e-02; norm: 3.7e-01; mom: 6.6e+00
#> It: 200 (best: 199); obj: 1.47e+00 (best: 1.47e+00); abs: 1.7e-03; rel: 1.2e-03; norm: 3.4e-02; mom: 1.2e+00
#> It: 300 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 4.8e-04; rel: 3.3e-04; norm: 1.6e-02; mom: 8.0e-01
#> It: 400 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 3.5e-04; rel: 2.4e-04; norm: 1.6e-02; mom: 7.9e-01
#> It: 500 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 3.3e-04; rel: 2.3e-04; norm: 1.6e-02; mom: 7.6e-01
#> It: 600 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 3.0e-04; rel: 2.0e-04; norm: 1.7e-02; mom: 8.3e-01
#> It: 700 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 1.7e-04; rel: 1.2e-04; norm: 1.4e-02; mom: 7.2e-01
#> It: 800 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 5.8e-04; rel: 4.0e-04; norm: 1.6e-02; mom: 7.6e-01
#> It: 900 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 4.0e-04; rel: 2.7e-04; norm: 1.5e-02; mom: 7.2e-01
#> It: 1000 (best: 251); obj: 1.45e+00 (best: 1.45e+00); abs: 4.1e-04; rel: 2.8e-04; norm: 1.8e-02; mom: 8.5e-01
#> **NO** CONVERGENCE. Decrease eta? Change init? Increase maxit?
```



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

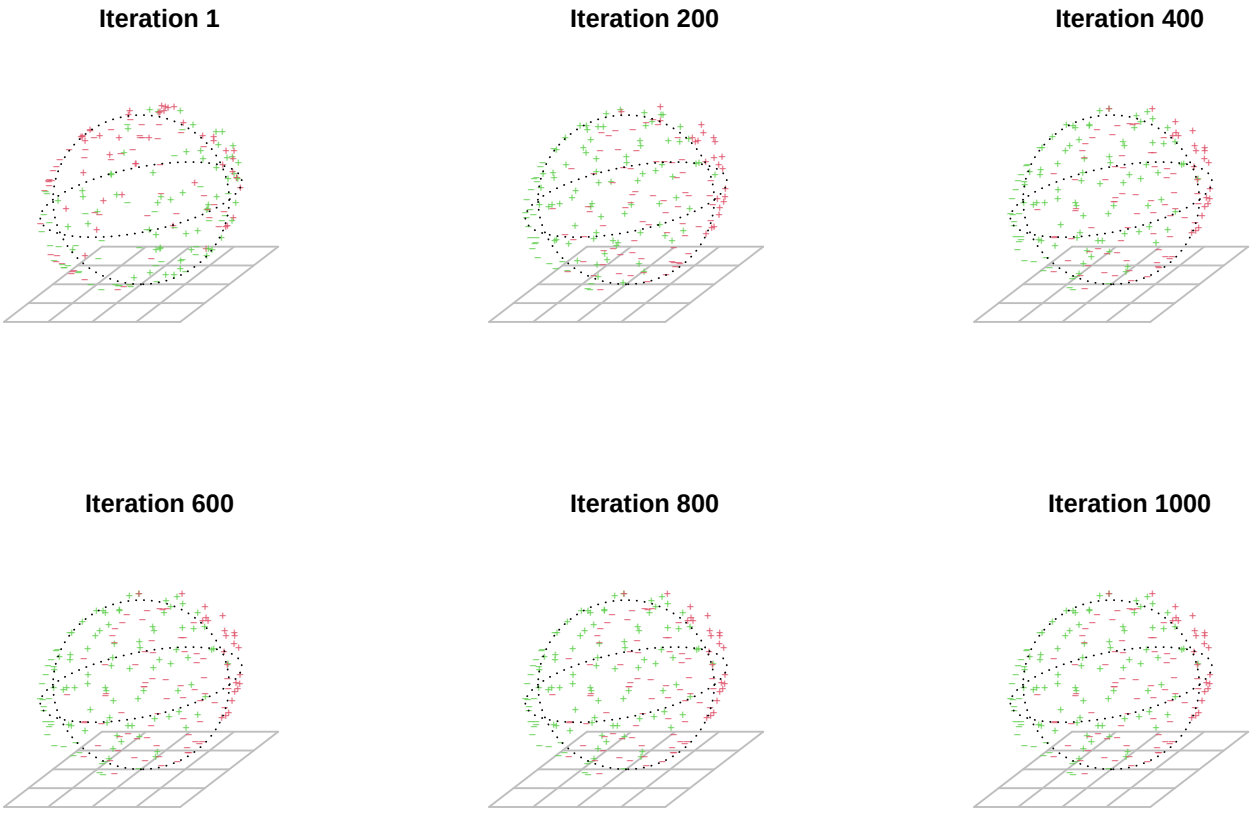
```
Y <- psc_sne_res_31$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1), ylim = c(-1, 1),
     axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```



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.

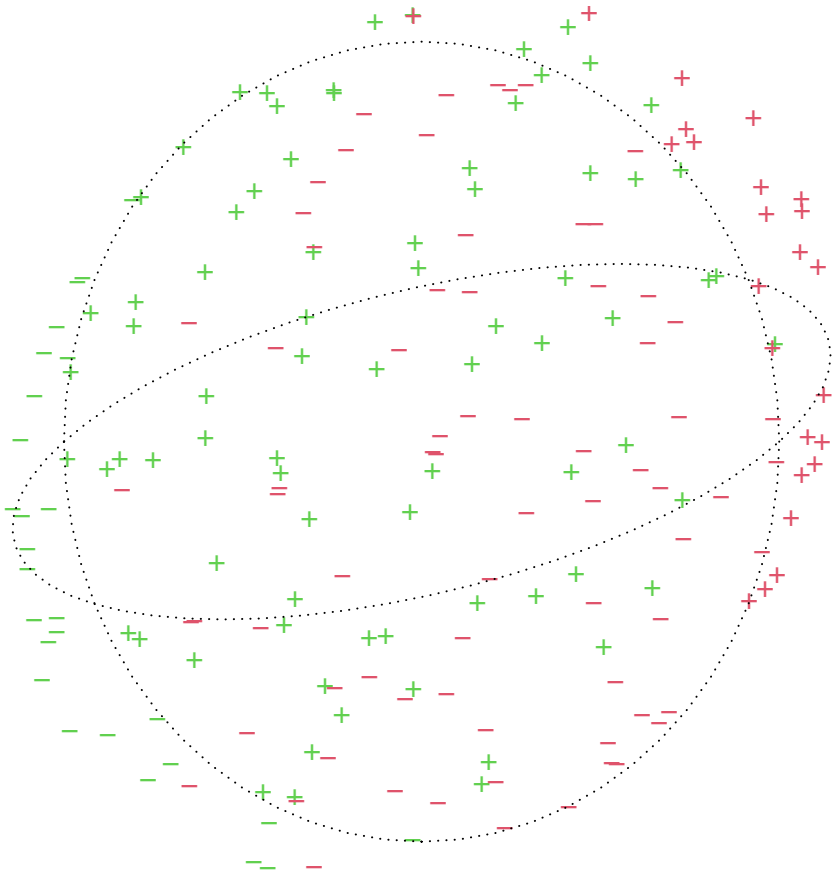
Reduction to $d = 2$ Let's reduce the data onto the sphere:

```
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 = 50,
                          parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.46e+01 (best: 1.46e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 2.8e-01; mom: 0.0e+00
#> It: 100 (best: 3); obj: 1.40e+01 (best: 1.32e+01); abs: 2.8e+00; rel: 1.7e-01; norm: 6.9e-01; mom: 7.3e+00
#> It: 200 (best: 181); obj: 1.29e+00 (best: 1.29e+00); abs: 1.6e-04; rel: 1.2e-04; norm: 4.6e-02; mom: 7.5e-01
#> It: 300 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 9.9e-06; rel: 7.8e-06; norm: 7.1e-03; mom: 1.6e-01
#> It: 400 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 500 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 600 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 700 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 800 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 900 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> It: 1000 (best: 287); obj: 1.27e+00 (best: 1.27e+00); abs: 1.0e-05; rel: 7.9e-06; norm: 7.2e-03; mom: 1.6e-01
#> **NO** CONVERGENCE. Decrease eta? Change init? Increase maxit?
```



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$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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 <- 200
r <- 100
set.seed(42)
x_s1_100_path <- r_path_s1r(n = n, r = r)
cols <- rainbow(n, alpha = 1)
# Change the order of the data
set.seed(42)
indexes <- sample(1:n)
x_s1_100_path <- x_s1_100_path[indexes, , ]
cols <- cols[indexes]
```

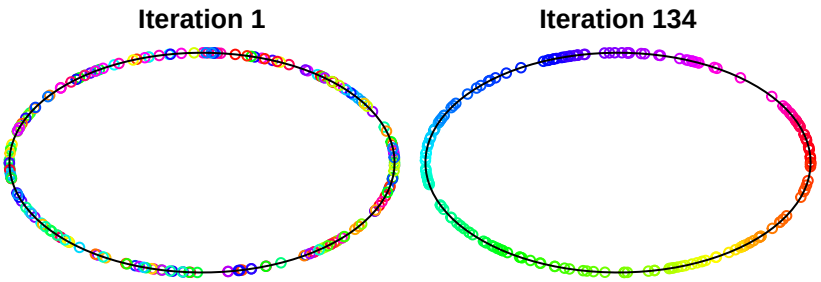
Now that we have created the data, we can calculate the rho based on a fixed perplexity.

```
rho_30_s1_100_path <- rho_optim_bst(x = x_s1_100_path, perp_fixed = 30,
                                    num_cores = num_cores_param)
#> Time difference of 35.14913 secs
```

Reduction to $d = 1$ First, let's reduce the dimension to the circumference:

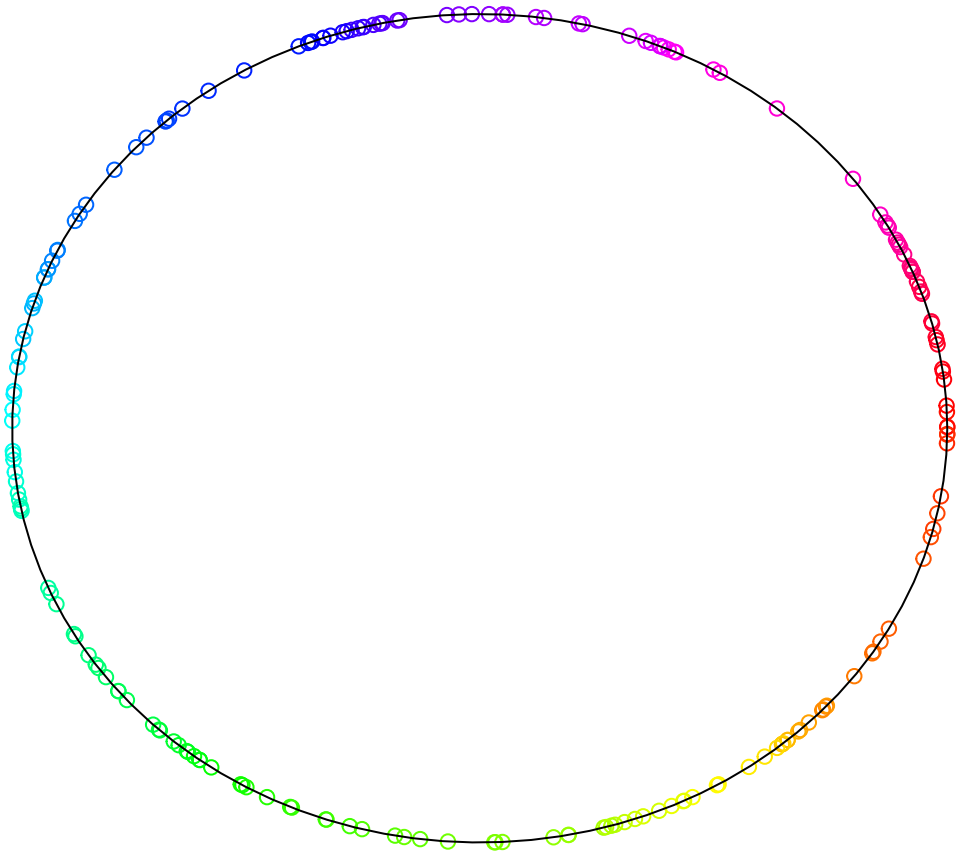
```
psc_sne_res_41 <- psc_sne(X = x_s1_100_path, d = 1,
                          rho_psc_list = rho_30_s1_100_path, colors = cols,
```

```
                        show_prog = TRUE, eta = 25,
                        parallel_cores = num_cores_param)
#> It: 1 (best: 1); obj: 1.40e+01 (best: 1.40e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 8.4e-02; mom: 0.0e+00
#> It: 100 (best: 24); obj: 9.26e+00 (best: 9.25e+00); abs: 1.1e-05; rel: 1.2e-06; norm: 2.4e-01; mom: 1.9e+00
#> It: 134 (best: 134); obj: 8.71e-01 (best: 8.71e-01); abs: 1.1e-11; rel: 1.3e-11; norm: 1.0e-06; mom: 1.2e-05
#> CONVERGENCE!
```



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

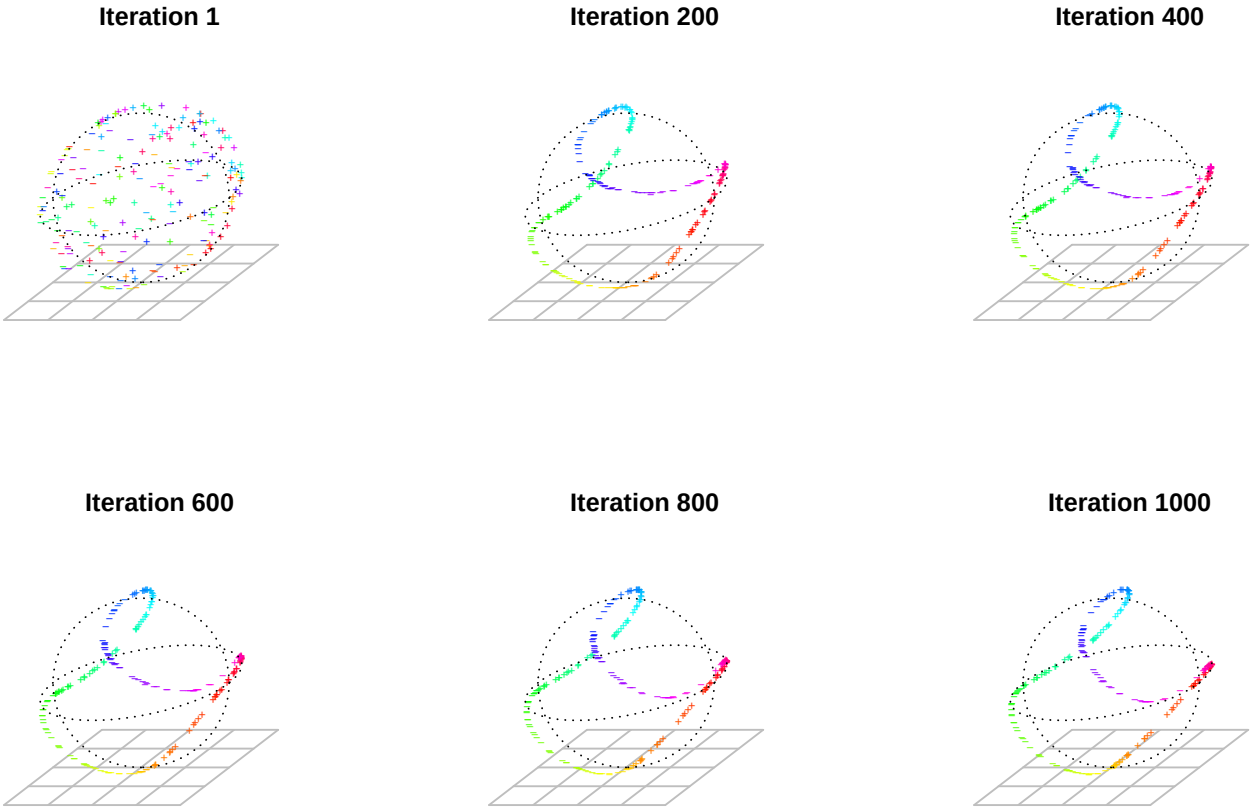
```
Y <- psc_sne_res_41$best_Y
plot(Y[, 1], Y[, 2], col = cols, xlim = c(-1, 1), ylim = c(-1, 1),
     axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```



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.

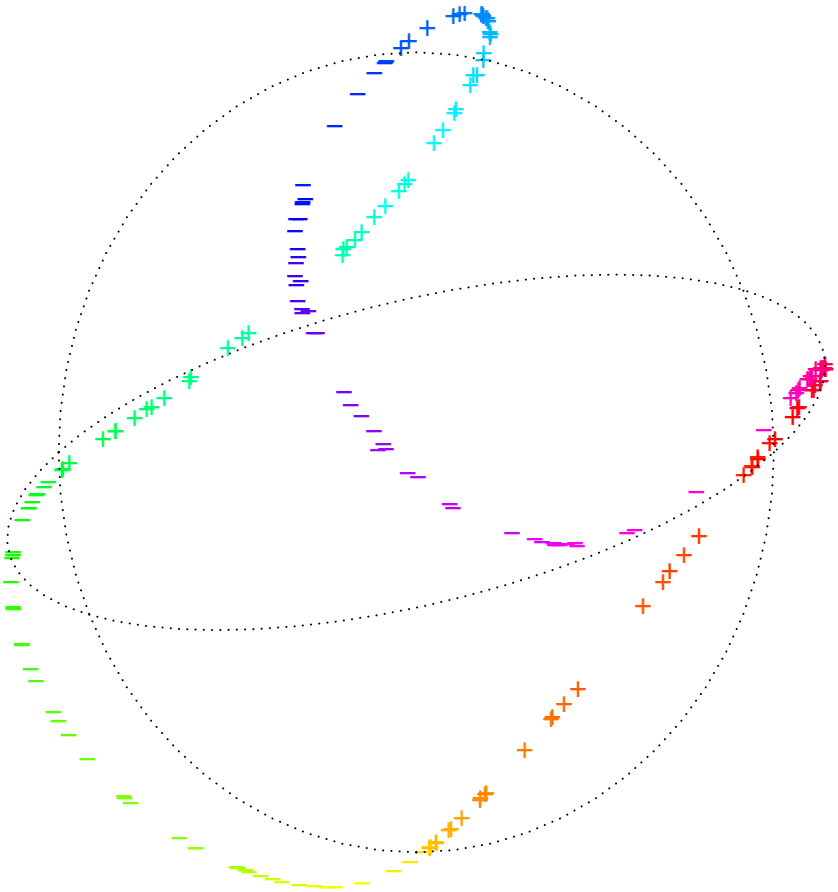
Reduction to $d = 2$ Let's see if results onto the sphere are better:

```
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 (best: 1); obj: 1.51e+01 (best: 1.51e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.7e-01; mom: 0.0e+00
#> It: 100 (best: 10); obj: 9.12e+00 (best: 8.67e+00); abs: 1.1e-02; rel: 1.2e-03; norm: 8.1e-01; mom: 5.2e+00
#> It: 200 (best: 200); obj: 2.64e-01 (best: 2.64e-01); abs: 2.0e-07; rel: 7.4e-07; norm: 6.3e-05; mom: 1.6e-03
#> It: 300 (best: 300); obj: 2.63e-01 (best: 2.63e-01); abs: 6.9e-07; rel: 2.6e-06; norm: 7.5e-05; mom: 7.4e-03
#> It: 400 (best: 400); obj: 2.63e-01 (best: 2.63e-01); abs: 8.8e-07; rel: 3.3e-06; norm: 8.4e-05; mom: 8.4e-03
#> It: 500 (best: 500); obj: 2.63e-01 (best: 2.63e-01); abs: 8.2e-07; rel: 3.1e-06; norm: 8.1e-05; mom: 8.1e-03
#> It: 600 (best: 600); obj: 2.63e-01 (best: 2.63e-01); abs: 6.5e-07; rel: 2.5e-06; norm: 7.2e-05; mom: 7.2e-03
#> It: 700 (best: 700); obj: 2.63e-01 (best: 2.63e-01); abs: 5.0e-07; rel: 1.9e-06; norm: 6.3e-05; mom: 6.3e-03
#> It: 800 (best: 800); obj: 2.63e-01 (best: 2.63e-01); abs: 3.8e-07; rel: 1.4e-06; norm: 5.5e-05; mom: 5.5e-03
#> It: 900 (best: 900); obj: 2.63e-01 (best: 2.63e-01); abs: 2.8e-07; rel: 1.0e-06; norm: 4.7e-05; mom: 4.7e-03
#> It: 1000 (best: 1000); obj: 2.63e-01 (best: 2.63e-01); abs: 1.9e-07; rel: 7.3e-07; norm: 3.9e-05; mom: 3.9e-03
#> **NO** CONVERGENCE. Decrease eta? Change init? Increase maxit?
```



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$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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)
```



Again, we can see the 1d-structure related to the time-index generated in the path since the closed rainbow colors are also located nearby.

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

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

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

Therefore, 95 spheres that follow a uniform distribution are generated in $\mathbb{S}^2$. The remaining spheres are formed by data that follow a small circle distribution located in the equator, two small circle distributions rotated randomly and two rotated spherical spiral distributions.

```
n <- 200
r1 <- 1
r2 <- 2
r3 <- 2
r4 <- 95
# Calculate the north pole of the sphere
```

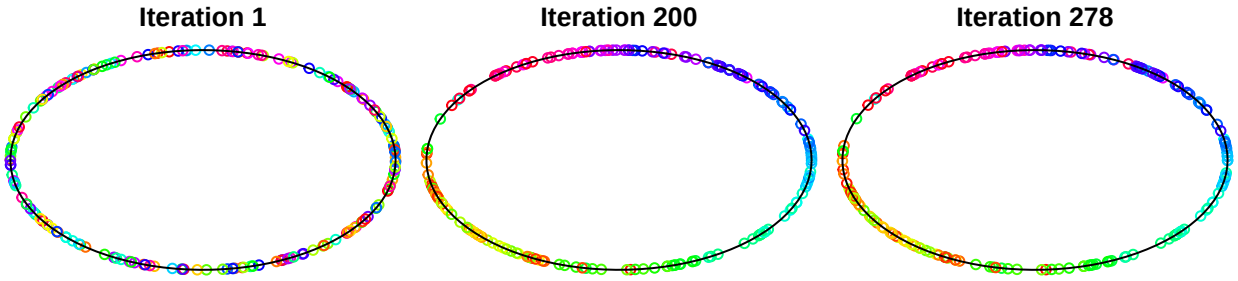
```
north_pole <- cbind(c(0, 0, 1))
# small circle in the equator (1 sample)
set.seed(42)
samp_1 <- r_path_s2r(n = n, r = r1, sigma = 0.08, Theta = north_pole)
# small circle rotated 1 and 2 (2 samples)
set.seed(42)
samp_2_3 <- r_path_s2r(n = n, r = r2, sigma = 0.1)
# spherical spiral 1 and 2 (2 samples)
set.seed(42)
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)
set.seed(42)
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
set.seed(42)
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 38.24012 secs
```

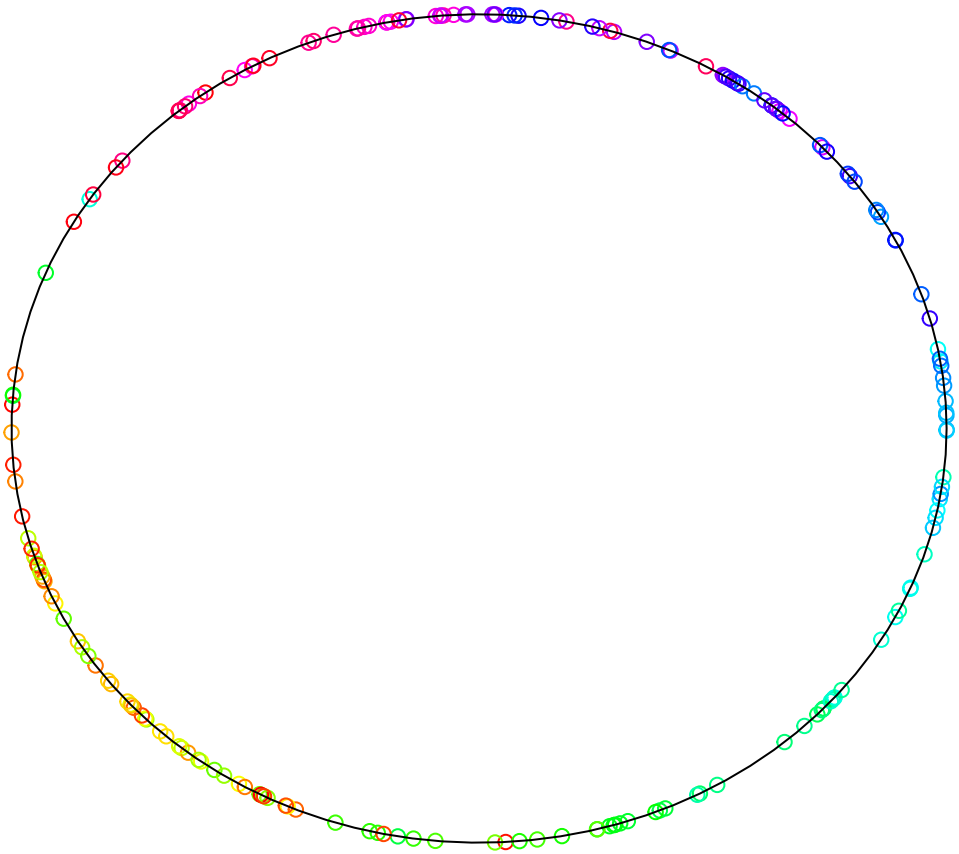
Reduction to $d = 1$ First, let's reduce the dimension to the circumference:

```
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 (best: 1); obj: 1.39e+01 (best: 1.39e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 1.2e-01; mom: 0.0e+00
#> It: 100 (best: 13); obj: 1.33e+01 (best: 1.19e+01); abs: 1.8e-15; rel: 1.3e-16; norm: 4.0e-01; mom: 3.0e+00
#> It: 200 (best: 200); obj: 1.43e+00 (best: 1.43e+00); abs: 2.1e-05; rel: 1.5e-05; norm: 6.4e-04; mom: 1.9e-02
#> It: 278 (best: 278); obj: 1.43e+00 (best: 1.43e+00); abs: 8.4e-11; rel: 5.9e-11; norm: 9.7e-07; mom: 7.8e-05
#> CONVERGENCE!
```



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),
     axes = FALSE, xlab = "", ylab = "")
th <- seq(0, 2 * pi, length.out = 100)
polygon(x = cos(th), y = sin(th))
```

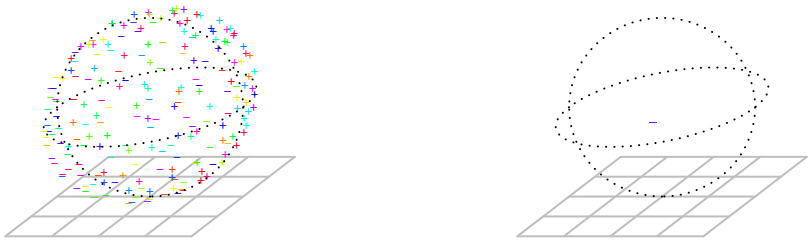


Results show more or less the time-index structure of the first 5 spheres even with the noise of the remainder spheres. Nevertheless, it is not possible to distinguish any shape.

Reduction to $d = 2$ 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 = 5,
                          parallel_cores = num_cores_param, maxit = 2000)
#> It: 1 (best: 1); obj: 1.58e+01 (best: 1.58e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 2.5e-01; mom: 0.0e+00
#> It: 100 (best: 24); obj: 1.30e+01 (best: 1.21e+01); abs: 5.3e-15; rel: 4.1e-16; norm: 1.9e-12; mom: 1.5e-11
#> It: 102 (best: 102); obj: 1.86e+00 (best: 1.86e+00); abs: 2.2e-16; rel: 1.2e-16; norm: 2.5e-12; mom: 2.5e-12
#> CONVERGENCE!
```

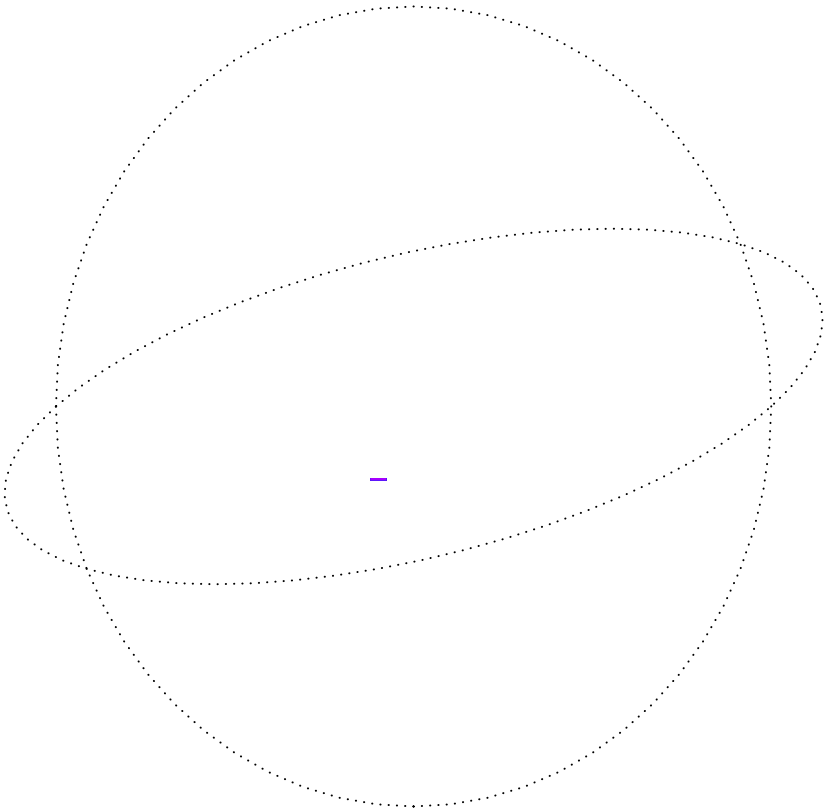
Iteration 1 **Iteration 102**



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)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
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)
```

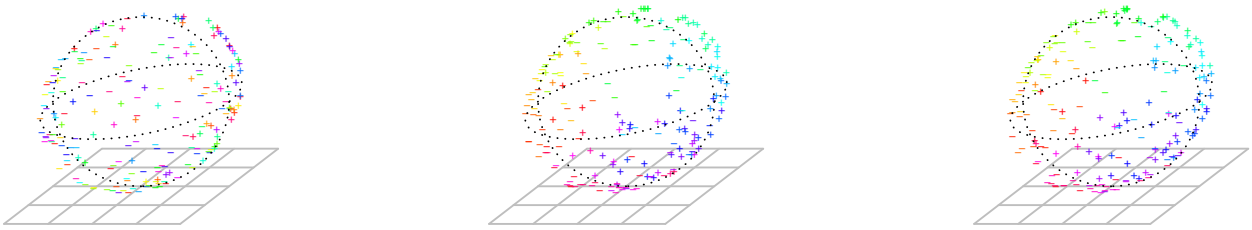


In this case, the resultant data show again the insights obtained in the reduced dimension when $d = 1$. It is not possible to see a clear shape, for example, a small circle or a spiral. The fact that there are lot of noise and other distributions involved, makes not possible to distinguish any clear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

Nevertheless, we are going to execute `psc_sne` with other parameters configuration to see if results are better. Instead of having the initialization of data points evenly spaced, now the initial values are randomly distributed:

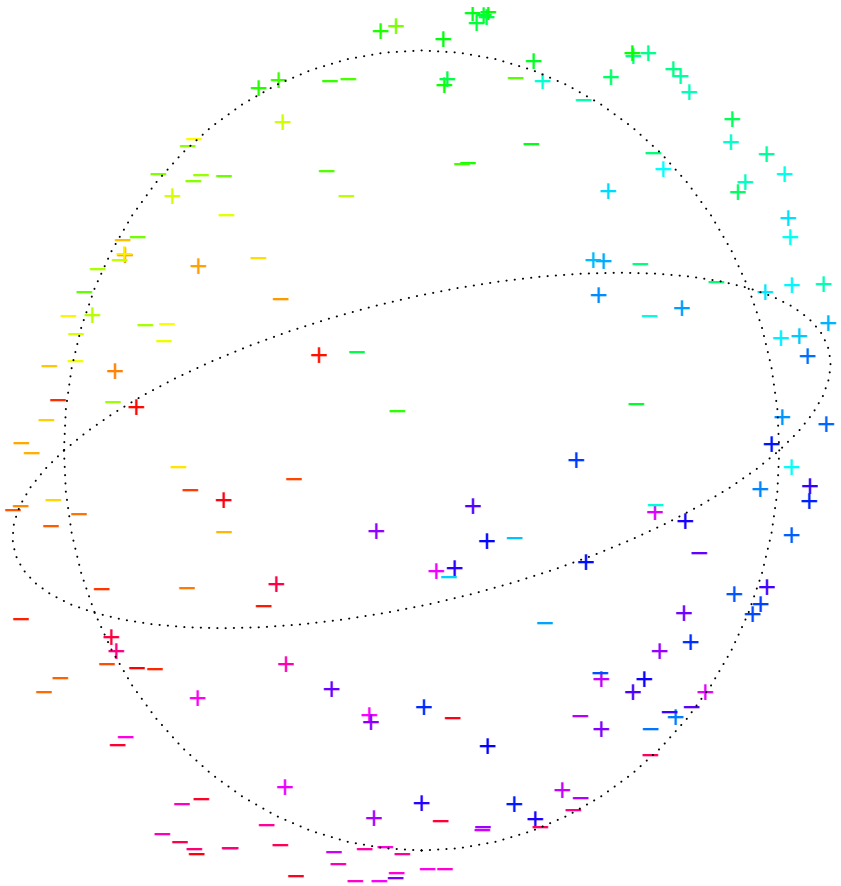
```
set.seed(42)
psc_sne_res_52 <- psc_sne(X = x_s2_100, d = 2, rho_psc_list = rho_30_s2_100,
                          colors = cols, show_prog = TRUE, eta = 30,
                          parallel_cores = num_cores_param, init = "random")
#> It: 1 (best: 1); obj: 1.50e+01 (best: 1.50e+01); abs: 0.0e+00; rel: 0.0e+00; norm: 2.5e-01; mom: 0.0e+00
#> It: 100 (best: 27); obj: 1.62e+01 (best: 1.35e+01); abs: 2.6e+00; rel: 1.9e-01; norm: 8.7e-01; mom: 5.7e+00
#> It: 200 (best: 200); obj: 1.24e+00 (best: 1.24e+00); abs: 1.4e-05; rel: 1.2e-05; norm: 4.9e-04; mom: 1.5e-02
#> It: 300 (best: 300); obj: 1.23e+00 (best: 1.23e+00); abs: 2.8e-08; rel: 2.3e-08; norm: 3.0e-05; mom: 2.0e-03
#> It: 330 (best: 330); obj: 1.23e+00 (best: 1.23e+00); abs: 2.9e-11; rel: 2.3e-11; norm: 9.7e-07; mom: 7.5e-05
#> CONVERGENCE!
```

Iteration 1 **Iteration 200** **Iteration 330**



Onto the sphere:

```
Y <- psc_sne_res_52$best_Y
sd3 <- scatterplot3d::scatterplot3d(
  Y, xlim = c(-1, 1), ylim = c(-1, 1), zlim = c(-1, 1),
  color = cols, xlab = "", ylab = "", zlab = "", axis = FALSE,
  pch = c("+", "-")[ifelse(sign(Y[, 2]) == 1, 1, 2)], grid = FALSE,
  mar = c(0, 0, 0, 0)
)
sd3$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)
```



There is not any advantage in using other initialization value. It is possible to see that points from the path that were generated next to each other are approximately close. Then, as it was saying, although there is not

any clear shape we can differentiate the time-index of the path.