ClusterByAlpha

Load libraries

```
library("ggplot2") # general plotting
library("ggpubr") # combining plots
library("ggpubr") # combining plots
library("Spectrum") # spectral clustering
library("hitandrun") # sampling from hypersphere
```

Specify the experiment parameters

```
set.seed(17) # seed for reproducibility
ntimes <- 100 # number of noise replicates to investigate cluster performance
npoints <- 25 # number of points per ground truth cluster to be investigated
maxdim <- 10000 # maximal dimension of the data set to be investigated
dims <- round(exp(seq(log(10), log(maxdim), length.out=4))) # dimensions to study from log-scale
sigma <- 1 # magnitude of noise: per dimension we sample from gaussian with std sigma
alphas <- c(3, 4, 5) # factors controling the growth rate of the ground truth diameters
```

Construct the ground data sets according to the growth rates

```
group <- factor(c(rep(1, npoints), rep(2, npoints)))
datasets <- lapply(alphas, function(alpha){
  if(alpha==Inf) center <- rep(1, maxdim) else center <- (1:maxdim)**(-1 / alpha)
  center[1:2] <- 5 * center[1:2]
  S1 <- (norm(center, type="2") / 4) * runif(npoints) * hypersphere.sample(maxdim, npoints)
  S2 <- t(replicate(npoints, center)) +
    (norm(center, type="2") / 4) * runif(npoints) * hypersphere.sample(maxdim, npoints)
  data.frame(rbind(S1, S2))
})
```

We view the magnitude of noise and spectral clustering result for various data sets

```
PlotList <- list() # empty list for storing plots
N <- matrix(rnorm(2 * maxdim * npoints, sd=sigma), ncol=maxdim) # noise matrix identical for all data
for(alpha_idx in 1:length(alphas)){
  # compute clustering and visualizations for this growth rate determined by alpha (noise-free)
  if(maxdim >= 1000) stringdim <- paste0(maxdim / 1000, "k") else stringdim <- as.character(maxdim)
  thisX <- datasets[[alpha_idx]]
  thisX$pred <- factor(Spectrum(data.frame(t(thisX)), maxk=2, # compute spectral clustering
                                silent=TRUE, showres=FALSE)$assignments)
  PlotList[[length(PlotList) + 1]] <- ggplot(data=thisX, # compute and store plot
                                             aes(x=X1, y=X2, fill=pred)) +
    geom_point(size=5, aes(shape=group)) +
    ggtitle(paste0("without noise \n alpha = ", alphas[alpha_idx], ", dim = ", stringdim)) +
    theme_bw() +
    theme(text=element_text(size=13), plot.title=element_text(hjust=0.5, size=17),
          legend.title=element_text(size=20), legend.text=element_text(size=20)) +
    scale_fill_manual(values=c("#F8766D", "#00BFC4")) +
    scale_shape_manual(values=c(21, 24)) +
    labs(fill="predicted cluster", shape="true cluster") +
```

```
    guides(fill=guide_legend(override.aes=list(colour=c("#F8766D", "#00BFC4"))))

  # add noise to data
  XN <- datasets[[alpha_idx]] + N
  for(dim in dims){
    thisX <- XN[,1:dim]
    # compute clustering and visualization for this growth rate and dimension (noisy)
    if(dim >= 1000) stringdim <- paste0(dim / 1000, "k") else stringdim <- as.character(dim)
    thisX$pred <- factor(Spectrum(data.frame(t(thisX)), maxk=2, # compute spectral clustering
                                  silent=TRUE, showres=FALSE)$assignments)
    PlotList[[length(PlotList) + 1]] <- ggplot(data=thisX, # compute and store plot
                                               aes(x=X1, y=X2, fill=pred)) +
      geom_point(size=5, aes(shape=group)) +
      ggtitle(paste0("with noise \n alpha = ", alphas[alpha_idx], ", dim = ", stringdim)) +
      theme_bw() +
      theme(text=element_text(size=13), plot.title=element_text(hjust=0.5, size=17),
            legend.title=element_text(size=20), legend.text=element_text(size=20)) +
      scale_fill_manual(values=c("#F8766D", "#00BFC4")) +
      scale_shape_manual(values=c(21, 24)) +
      labs(fill="predicted cluster", shape="true cluster") +
      guides(fill=guide_legend(override.aes=list(colour=c("#F8766D", "#00BFC4"))))
  }
}

do.call("ggarrange", c(PlotList, nrow=length(alphas), ncol=length(dims) + 1, common.legend=TRUE))
```