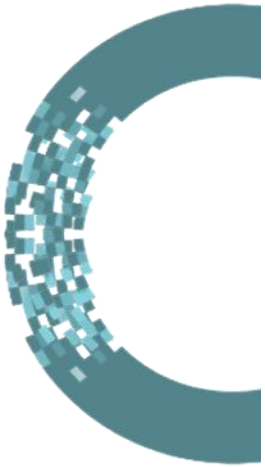
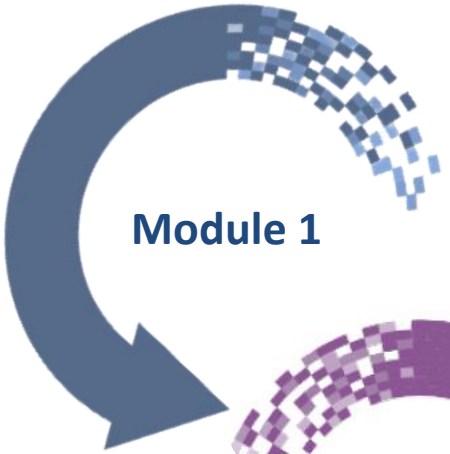
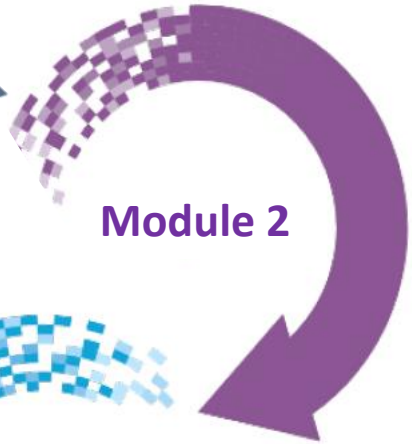
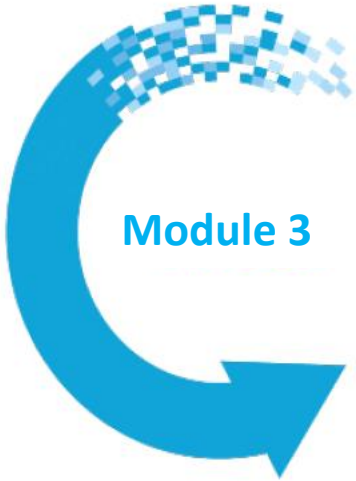
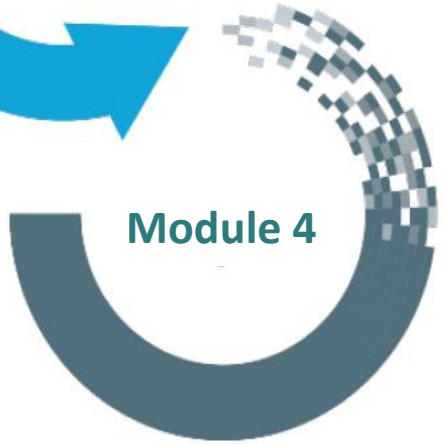


Module 4:
Resilience Analysis









Exploratory Data Analysis

Importing data

Pre-processing phase

Filtering phase

Network Analysis

Building the correlation network

Checking network integrity

Finding communities in the network

Switch Mining

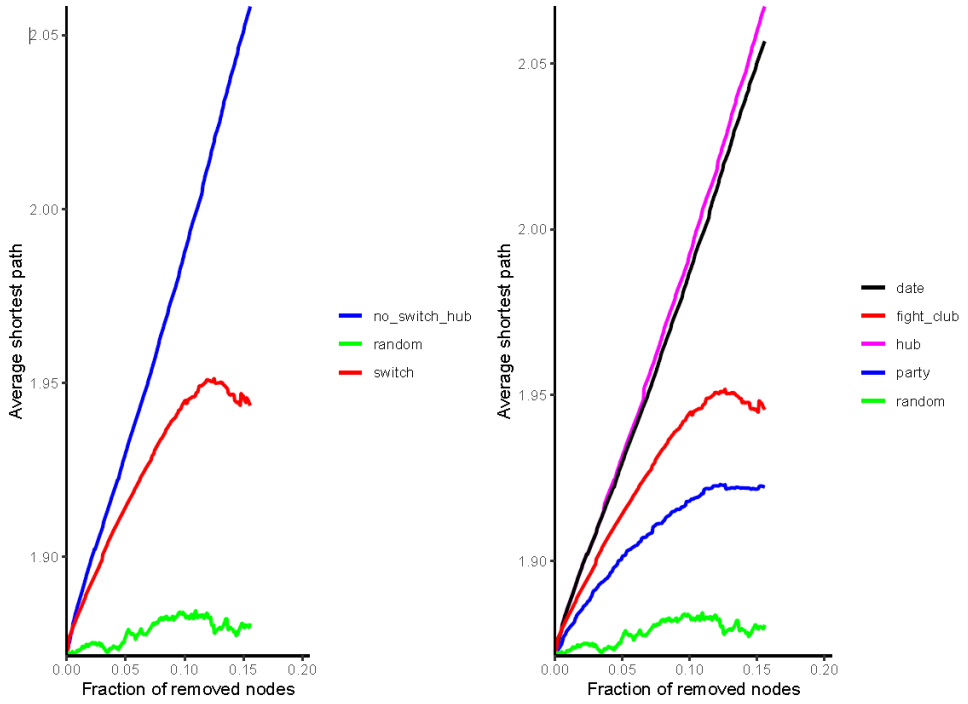
Building the heat cartography map

Identification of switch genes

Resilience Analysis

Evaluation of network resilience

Network resilience plot



- x-axis represents the cumulative fraction of removed nodes
- y-axis represents the average shortest path

Network resilience (robustness)

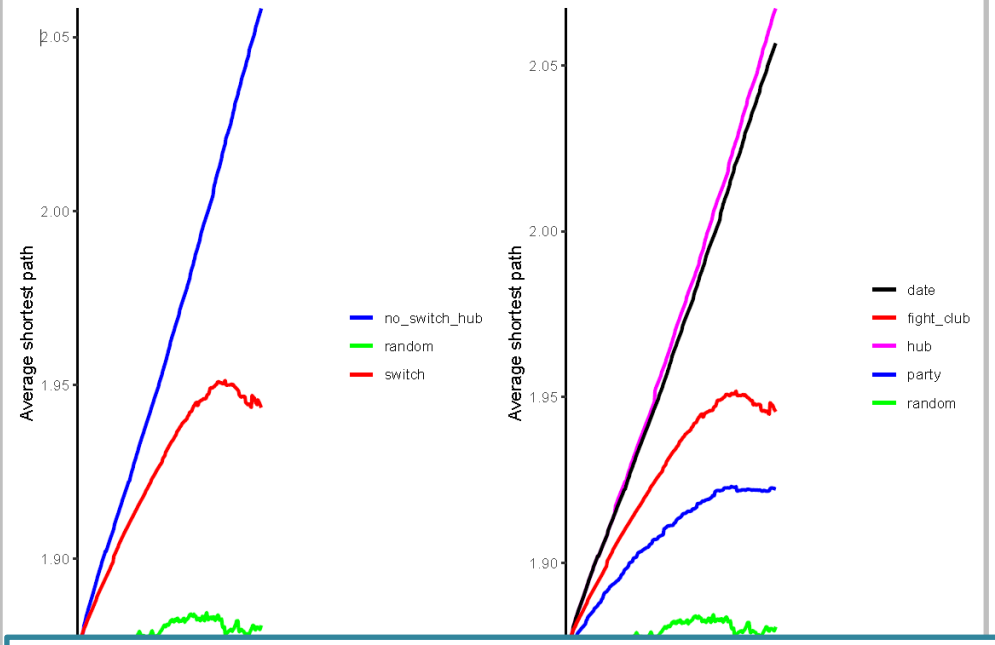
- This step evaluates the **network robustness**
- This step is **optional**: set “yes” on config.R

```
removal_node <- "yes"
```

- **Network robustness** is defined as the resilience to errors
- In complex networks, it focuses on the response of the network to the removal of nodes or links

⚠ **Caveat:** this could take long time depending on the switch numbers

Network resilience plot



Network resilience (robustness)

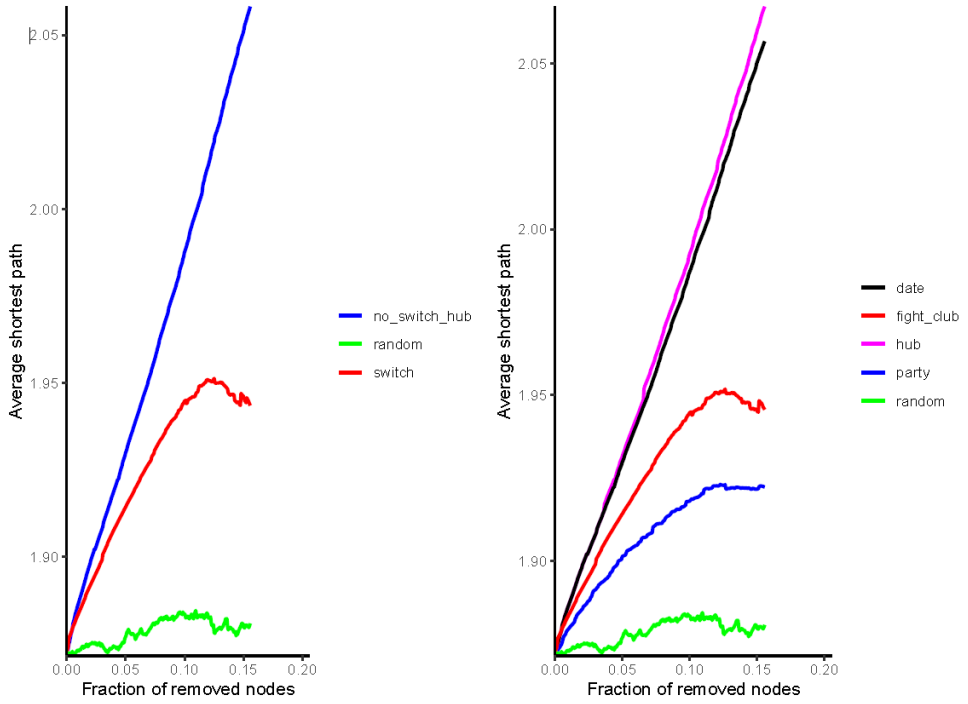
- This step evaluates the **network robustness**
- This step is **optional**: set "yes" on config.R

```
removal_node <- "yes"
```

- **Network robustness** is defined as the resilience to errors
- In complex networks, it focus on the

Scale-free networks have few hubs and many non-hub nodes, and thus they are amazingly resistant to a random removal of nodes, while the removal of hubs causes an effect known as "vulnerability to attack" to allude to the fact that the integrity of the network is destroyed

Network resilience plot

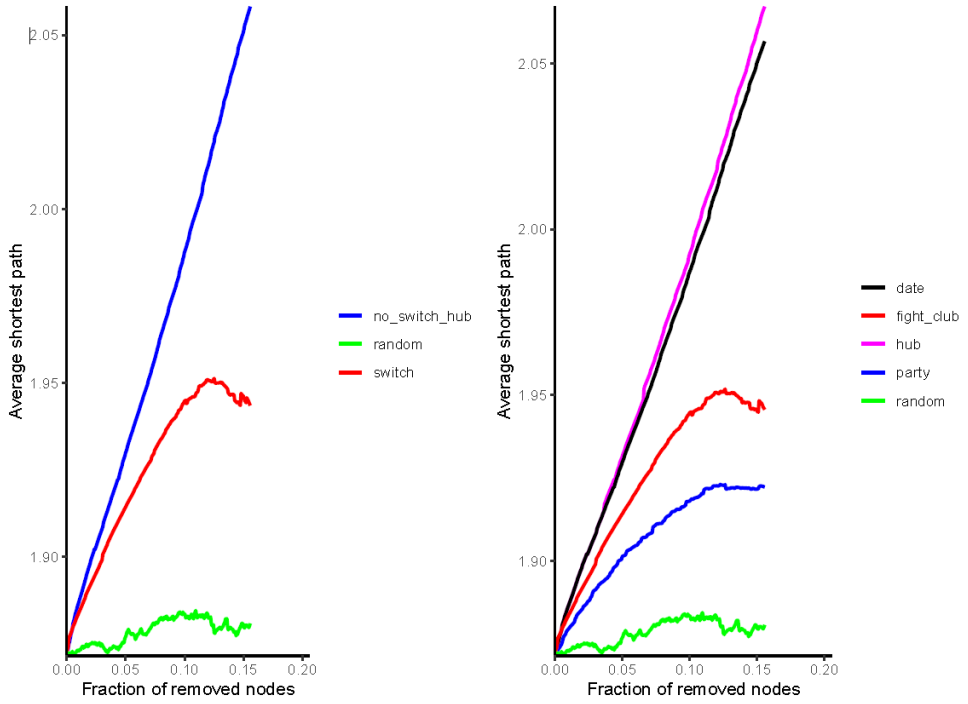


- x-axis represents the cumulative fraction of removed nodes
- y-axis represents the average shortest path

Network resilience (robustness)

- Network robustness is performed by studying the effect on the network connectivity of **removing different types of nodes by decreasing degree**
- The total number of nodes to be removed must be equal to the total number of switch genes and the cumulative node deletion is carried out by type (i.e., total hubs, party hubs, date hubs, fight-club hubs, switch genes, and randomly chosen nodes).
- The plot shows the **average shortest path length** as function of the cumulative fraction of removed nodes

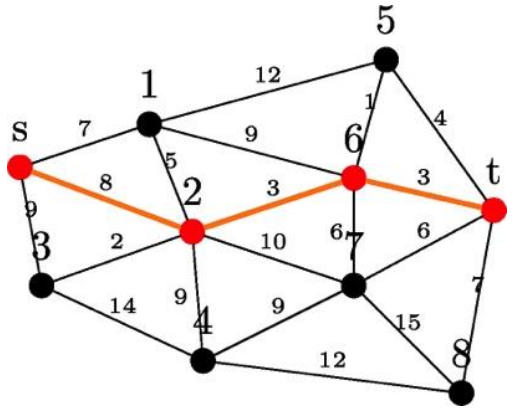
Network resilience plot



- x-axis represents the cumulative fraction of removed nodes
- y-axis represents the average shortest path

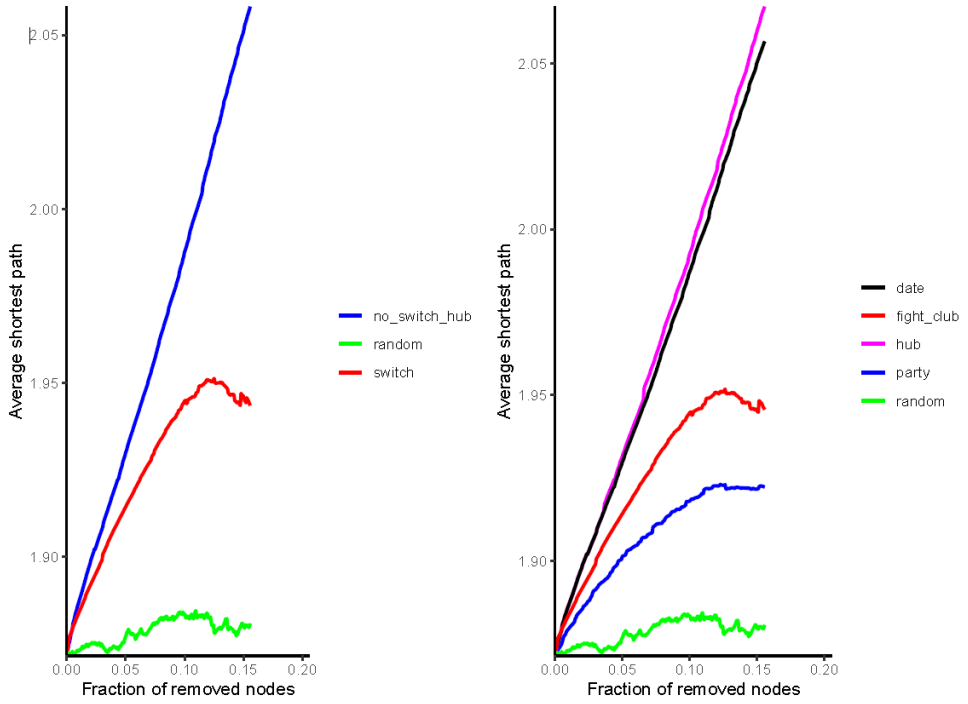
Average shortest path

- the **shortest path** between two nodes is the minimum number of edges connecting them



- the **average shortest path** of a network is the average of the shortest paths for all possible pairs of network nodes

Network resilience plot

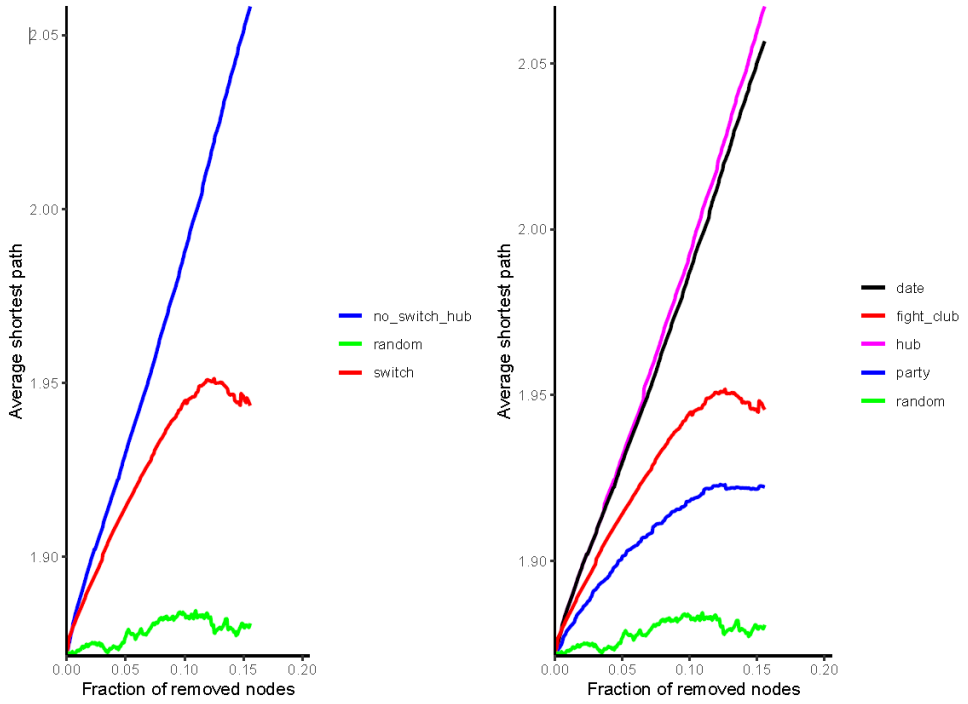


- x-axis represents the cumulative fraction of removed nodes
- y-axis represents the average shortest path

Network resilience computation

```
computeConnectivity <- function(network){

  graph <- graph_from_data_frame(network, directed = F)

  graph <- simplify(graph, remove.multiple = TRUE, remove.loops = TRUE,
                    edge.attr.comb = igraph_opt("edge.attr.comb"))

  node <- names(V(graph))

  degree <- degree(graph, v = V(graph))

  component <- components(graph)

  ind <- which(component$membership == which.max(component$csize))

  matrix_sp <- distances(graph, v = V(graph)[ind], to = V(graph)[ind])

  matrix_sp[upper.tri(matrix_sp)] <- NA
  diag(matrix_sp) <- NA

  mean_sp <- mean(matrix_sp, na.rm = T)

  res <- list(node = node, degree = degree, mean_sp = mean_sp)

  return(res)

}
```

Network resilience plot

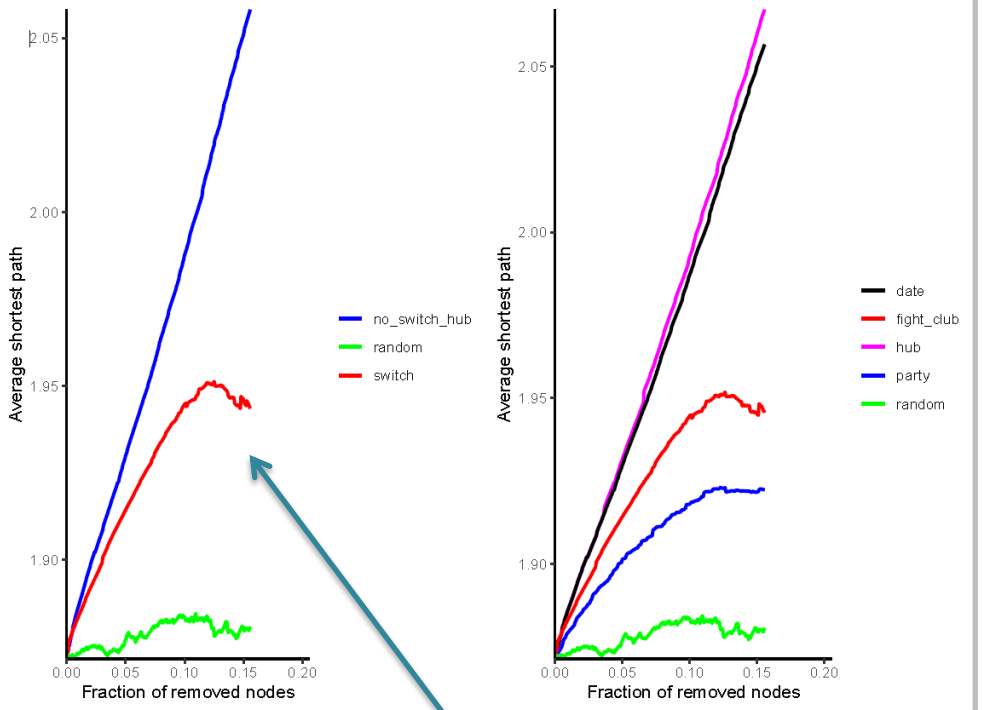


- x-axis represents the cumulative fraction of removed nodes
- y-axis represents the average shortest path

Network resilience plot

```
getResiliencePlot <- function(df,output_file){

  M <- max(df$frac) + 0.05

  p1 <- ggplot(df, aes(x = frac)) + scale_x_continuous(expand = c(0, 0), limits = c(0,M)) +
    scale_y_continuous(expand = c(0, 0)) +
    geom_line(aes(y = no_switch_hub, color = "no_switch_hub"), size = 1) +
    geom_line(aes(y = switch, color = "switch"), size = 1) +
    geom_line(aes(y = random, color = "random"), size = 1) +
    scale_colour_manual(name = "", values = c(no_switch_hub = "blue", "switch" = "red", "random" = "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
          panel.background = element_blank(), #axis.title = element_text(face = "bold"),
          axis.line = element_line(colour = "black"),
          legend.key = element_rect(fill = "white", colour = "white")) +
    labs(x = "Fraction of removed nodes", y = "Average shortest path")

  p2 <- ggplot(df, aes(x = frac)) + scale_x_continuous(expand = c(0, 0), limits = c(0,M)) +
    scale_y_continuous(expand = c(0, 0)) +
    geom_line(aes(y = hub, color = "hub"), size = 1) +
    geom_line(aes(y = date, color = "date"), size = 1) +
    geom_line(aes(y = party, color = "party"), size = 1) +
    geom_line(aes(y = fight_club, color = "fight_club"), size = 1) +
    geom_line(aes(y = random, color = "random"), size = 1) +
    scale_colour_manual(name = "", values = c(hub = "magenta", "date" = "black", "party" = "blue", "fight_club"
= "red", "random" = "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
          panel.background = element_blank(), #axis.title = element_text(face = "bold"),
          axis.line = element_line(colour = "black"),
          legend.key = element_rect(fill = "white", colour = "white")) +
    labs(x = "Fraction of removed nodes", y = "Average shortest path")

  grid.arrange(p1, p2, ncol=2)

  p <- arrangeGrob(p1, p2, ncol=2)

  ggsave(output_file,p)
```

Network resilience plot



Network resilience plot

```
getResiliencePlot <- function(df,output_file){

  M <- max(df$frac) + 0.05

  p1 <- ggplot(df, aes(x = frac)) + scale_x_continuous(expand = c(0, 0), limits = c(0,M)) +
    scale_y_continuous(expand = c(0, 0)) +
    geom_line(aes(y = no_switch_hub, color = "no_switch_hub"), size = 1) +
    geom_line(aes(y = switch, color = "switch"), size = 1) +
    geom_line(aes(y = random, color = "random"), size = 1) +
    scale_colour_manual(name = "", values = c(no_switch_hub = "blue", "switch" = "red", "random" = "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
          panel.background = element_blank(), #axis.title = element_text(face = "bold"),
          axis.line = element_line(colour = "black"),
          legend.key = element_rect(fill = "white", colour = "white")) +
    labs(x = "Fraction of removed nodes", y = "Average shortest path")

  p2 <- ggplot(df, aes(x = frac)) + scale_x_continuous(expand = c(0, 0), limits = c(0,M)) +
    scale_y_continuous(expand = c(0, 0)) +
    geom_line(aes(y = hub, color = "hub"), size = 1) +
    geom_line(aes(y = date, color = "date"), size = 1) +
    geom_line(aes(y = party, color = "party"), size = 1) +
    geom_line(aes(y = fight_club, color = "fight_club"), size = 1) +
    geom_line(aes(y = random, color = "random"), size = 1) +
    scale_colour_manual(name = "", values = c(hub = "magenta", "date" = "black", "party" = "blue", "fight_club'
= "red", "random" = "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
          panel.background = element_blank(), #axis.title = element_text(face = "bold"),
          axis.line = element_line(colour = "black"),
          legend.key = element_rect(fill = "white", colour = "white")) +
    labs(x = "Fraction of removed nodes", y = "Average shortest path")

  grid.arrange(p1, p2, ncol=2)

  p <- arrangeGrob(p1, p2, ncol=2)

  ggsave(output_file,p)
```

The behavior of switch genes is deeply different from the random demonstrating the importance of the switch for the integrity of the network.