**Supplementary Information**

**Estimating Networks from Continuous or Categorical Data**

For estimating networks from continuous or categorical data, we can use the function EBICglasso in combination with the function cor­\_auto. Both functions are provided in the R package *qgraph* (Epskamp et al., 2016). EBICglasso is used in a similar way as the IsingFit function, but requires as input a correlation matrix instead of a data frame. For this we can use the function cor­\_auto, which detects whether variables are continuous or categorical and then calculates the appropriate correlation matrix (e.g., polychorious correlations if the variables are categorical). If our data were continuous or categorical, we could estimate attitude networks in the following way:

ObamaWeiAdj <- EBICglasso(cor\_auto(Obama), nrow(Obama))

This command estimates a network from the data and stores its weight adjacency matrix to the object ObamaWeiAdj. The first argument provides the input correlation matrix and the second argument specifies the number of cases on which the correlation matrix was estimated. We can then use the weight adjacency matrix as input for a *qgraph* object or an *igraph* object and run the same analyses on this network as we described in the tutorial.

ObamaqGraph2 <- qgraph(ObamaWeiAdj)

ObamaiGraph2 <- graph\_from\_adjacency\_matrix(abs(ObamaWeiAdj), 'undirected', weighted = TRUE, add.colnames = FALSE)

**Inclusion of Covariates**

It is also possible to add covariates when one wants to estimate an attitude network. We would advice researches, who want to do this, to first estimate a network including the relevant attitude items and the covariates. Second, one can extract the network based only on the attitude items and run network analyses on this network and plot the network without the covariates. Adding the covariates gender and age to the Obama network can be done in this way:

ObamaCov <- data.frame(ObamaCog, ObamaAff, gender, age)

ObamaCov <- na.omit(ObamaCov)

ObamaFitCov <- mgmfit(ObamaCov, c(rep('c', 11),'g'), c(rep(2, 11),1), binary.sign = TRUE)

Note that we use the function *mgmfit*, available in the R package *mgm* (Haslbeck, 2016), because we need to estimate a network from data involving different kind of variables, as the evaluative reactions and gender are binary variables and age is a continuous variable. The first argument in the function specifies the data frame on which the network is estimated, the second argument specifies the type of the variables, with ‘c’ indicating categorical variables and ‘g’ indicating Gaussian or continuous variables, and the binary.sign argument specifies that we also want to estimate weather binary variables are positively or negatively connected (which is not done by default in the *mgmfit* function). To get a weight adjacency matrix similar to the weight adjacency matrix of the *IsingFit* function, we need to combine the following two outputs of the *mgmfit* function:

ObamaFitCov$signs[is.na(ObamaFitCov$signs)] <- 0

ObamaNetCovIn <- ObamaFitCov$wadj\*ObamaFitCov$signs

The object ObamaNetCovIn now contains the weight adjacency matrix of the evaluative reactions and the covariates. We can select the weight adjacency matrix of only the evaluative reactions in the following way:

ObamaNetCovOut <- ObamaNetCovIn[1:10,1:10]

The object ObamaNetCovOut now contains the weight adjacency matrix of the evaluative reactions that is estimated when gender and age is controlled for.

**Edge Stability**

As can be seen in Figure 2, edges differ quite strongly in their magnitude. If one assumes that the edges represent causal connections, one can use the magnitude of the edges to infer to what extent change in one evaluative reaction would cause change in another evaluative reaction. However, edges are subject to sampling error, so that, to be able to interpret these differences, we need to assess the *stability* of the network structure (Epskamp, Borsboom, & Fried, 2016). To assess the stability of the edge weights, we can use the function bootnet, available in the R package *bootnet* (Epskamp, 2016). This function creates samples based on bootstrapping and returns estimates of how strongly network parameters differ between samples. We can run the function bootnet on the Obama network in this way:

ObamaSta <- bootnet(Obama, 1000, 'IsingFit')

The first argument of the bootnet function specifies the data on which the function is run, the second argument specifies the number of bootstrapped samples, and the third argument specifies the estimation method for creating the networks. We can plot the results of the bootstrapping analysis for the stability of the edge weights in this way:

plot(ObamaSta, plot = 'interval', order = 'sample')

This function creates the plot shown in Figure S1. The first argument in the plot function specifies that we want to plot the object ObamaSta, the plot argument specifies that we want to plot the bootstrapped confidence intervals around the edge weights and the order argument specifies that we want the edges to be ordered according to their magnitude.

*Macintosh HD:Users:Jonas:Google Drive:SPPS Tutorial Paper:Figure2.pdfFigure S1*. Plot of the stability of the edge estimates of the Obama network. Points indicate the mean edge weights of the bootstrapped samples and lines indicate the width of the 95% confidence interval. See Table 1 for the abbreviations of the nodes.

As can be seen in Figure S1, the confidence intervals are relatively small, which indicates that large differences in edge weights are meaningful. The strongest positive edge is the connection between judging Barack Obama as knowledgeable and as intelligent (Kno-Int). The confidence interval of this edge only overlaps with the confidence interval of the second strongest positive edge, which is the connection between feeling hopeful and feeling proud toward Obama (Hop-Prd). It is thus certain that the edge Know-Int differs significantly from all the other edges except from the edge Hope-Prou. The edges Know-Int and Hope-Prou might also differ significantly, as their confidence intervals overlap only slightly. Edges that have largely overlapping confidence intervals should not be interpreted as differing in their magnitude.

**Illustration of Shortest Path Lengths**

To make the concept of shortest path lengths less abstract we use two examples as illustrations. As a first example, take the shortest path length (*l*) between the nodes Kno and Int, which is equal to the inverse of their direct strong connection, because no indirect connection between these two nodes represents a shorter path (*lKnoInt* = 2.66-1 = 0.38). Conversely, the shortest path between the nodes Led and Int does not reflect the direct connection between these two nodes, as the direct connection between the two nodes is weak. The shortest path between these two nodes runs through the node Kno. The shortest path length between Led and Int then is the sum of the inverses of the edge weights between the nodes Kno and Int and between the nodes Led and Kno (*lLedInt* = 1.38-1 + 2.66-1 = 1.10), which is smaller than the path based on the direct connection between Led and Int.

**Centrality Stability**

Centrality indices are relatively complicated functions of a potentially large number of statistical estimates, each of which is subject to sampling error. As a result, these indices can vary quite substantially across samples and their values should be interpreted with care. Fortunately, we can again use the bootnet function to investigate the stability of the different centrality indices. In contrast to assessing the stability of the edges, we cannot use bootstrapping to assess the stability of the centrality indices (Epskamp et al., 2016). An alternative to bootstrapping is to investigate how strongly the centrality estimates are affected when subsets of the sample are used to estimate the network. We can specify the bootnet function to do this in the following way:

ObamaCenSta <- bootnet(Obama, 1000, ‘IsingFit’, ‘person’)

Setting the fourth argument of the bootnet function to ‘person’ results in the calculation of the centrality estimates for different subsamples of the data. We can plot the results of this analysis in the following way:

plot(ObamaCenSta, subsetRange = c(100,50))

*Macintosh HD:Users:Jonas:Google Drive:SPPS Tutorial Paper:Figure4.pdfFigure S2*. Stability plot of the centrality estimates of the Obama network. The lines represent the mean correlations between the given sampled percentage and the complete sample and the shades represent the area between the 2.5% and 97.5% percentile of the sampled estimates.

This command creates the plot shown in Figure S2. The argument subsetRange specifies the samples we want the plot to show. In this case the plot shows the correlation between the centrality estimates of the whole sample and samples based on 50 to 100 % of the data. As can be seen in Figure S2, both the strength and closeness estimates are highly stable. Even for the samples including only 50 % of the individuals of the complete sample, the correlation with the centrality indices of the complete sample stays close to 1. It is thus safe to interpret the estimated strength and closeness estimates of the Obama network. For the betweenness estimates the story is a bit different as the correlation drops continuously when less individuals are included. The correlation, however, stays positive and also of relatively high magnitude. To further investigate the stability of the betweenness estimates, we can plot how much each betweeness estimate fluctuates in the different samples. We can do this by using this command:

plot(ObamaCenSta, ‘betweenness’, perNode = TRUE, subsetRange = c(100,50))

This command creates the plot in Figure S3. The second argument of the plot function specifies that we only want to plot the betweeness estimates and the perNode argument specifies that we want to plot the betweenness estimates of every node. As can be seen in Figure S3, the ordering of the betweeness estimates is sufficiently stable: Only nodes that have the same betweeness estimates in the complete sample change their position in the different samples. It is thus also safe to interpret the betweenness estimates of the Obama network.

Often results of the stability analyses might not be as positive as in our current example. We would advise researchers to first test whether the correlations between the centrality estimates of the complete sample and the subsamples stay close to 1 (as is the case here for the closeness and strength indices). If this turns out to be the case it is safe to interpret the ordering of the centrality indices. In the case that correlations clearly drop below 1 (as is the case here for the betweenness indices), we advice researchers to investigate whether the ordering of the centrality indices remains the same for the different subsamples. Differences between indices that change their position in the different subsamples should not be interpreted, but one can still interpret the differences between the indices that do not change their position (e.g., one node stands out as highest in betweenness while the other nodes change their positions).

*Macintosh HD:Users:Jonas:Google Drive:SPPS Tutorial Paper:Figure5.pdfFigure S3*. Stability plot of the betweenness estimates of the Obama network. See Table 1 for the abbreviations of the nodes.

References

Epskamp, S. (2016). bootnet: Bootstrap methods for various network estimation routines (R package Version 0.2) [Computer Software]. Available at https://cran.r- project.org/web/packages/bootnet/index.html

Epskamp, S., Borsboom, D. & Fried, E.I. (2016). *Estimating psychological networks and their stability: A tutorial paper*. Retrieved from https://arxiv.org/abs/1604.08462

Epskamp, S., Costantini, G., Cramer, A. O. J., Waldorp, L. J., Schmittmann, V. D., & Borsboom. D. (2016). qgraph: Graph plotting methods, psychometric data visualization and graphical model estimation (R package Version 1.3.4) [Computer Software]. Available at https://cran.r-project.org/web/packages/qgraph/index.html

Haslbeck, J. (2016). mgm: Estimating time-varying mixed graphical models (R package Version 1.1-7) [Computer Software]. Available at https://cran.r- project.org/web/packages/mgm/index.html