HW3: Psychological Networks

Marwin Carmo

Introduction

Welcome to Homework Assignment No. 3. For this week's assignment, you'll find the dataset's `.RDS` file on Canvas.

Instructions

- Download the dataset from Canvas as a `.RDS` file
- Load that `.RDS` file into your `R` environment and begin the assignment
  - This can be done with the command, 'readRDS()"

Below, you will find a template of the questions and fields to provide answers in either `R` or text format. Please use a mix of code and text to answer each question.

```
library(qgraph)
library(GGMnonreg)
```

```
Registered S3 method overwritten by 'GGMnonreg':
  method     from
  plot.graph GGMncv
```

Questions

Question 1. Read in your graph's `.RDS` file and provide descriptive statistics of each of the variables. This can be done in the `psych` package using the `describe()` function. Describe the data verbally.

Answer

```r
data_ggm <- readRDS("data.ggm.RDS")

psych::describe(data_ggm)
```

```
    vars    n  mean   sd median trimmed  mad   min  max range  skew kurtosis
X1     1 1000 -0.08 1.01  -0.09   -0.08 0.97 -3.23 3.65  6.88  0.06     0.15
X2     2 1000  0.00 0.99   0.01    0.00 0.95 -4.24 3.56  7.79 -0.12     0.43
X3     3 1000 -0.04 1.01  -0.05   -0.04 1.00 -2.96 3.17  6.12 -0.02    -0.10
X4     4 1000 -0.05 1.02  -0.01   -0.04 0.98 -3.83 3.66  7.50 -0.14     0.32
X5     5 1000 -0.03 1.04  -0.06   -0.03 1.07 -2.95 3.39  6.34  0.00    -0.06
X6     6 1000  0.01 0.97   0.03    0.01 0.96 -2.92 3.00  5.92 -0.07    -0.01
X7     7 1000  0.03 0.99   0.00    0.02 1.02 -3.27 3.08  6.35  0.02    -0.14
X8     8 1000  0.03 1.00   0.06    0.03 1.01 -3.45 3.71  7.17 -0.06     0.03
X9     9 1000 -0.07 0.99  -0.07   -0.07 0.98 -3.32 2.96  6.28 -0.04     0.12
X10   10 1000 -0.02 1.03  -0.04   -0.03 1.02 -3.19 3.21  6.40  0.09    -0.11
X11   11 1000  0.04 0.99   0.06    0.04 0.98 -3.15 3.19  6.34 -0.06    -0.07
X12   12 1000  0.05 0.98   0.04    0.04 0.98 -2.77 4.15  6.92  0.13     0.15
X13   13 1000  0.01 1.04   0.02    0.02 1.01 -3.56 3.88  7.44  0.01    -0.07
X14   14 1000 -0.03 1.04  -0.02   -0.03 1.03 -3.56 3.21  6.77 -0.05     0.00
X15   15 1000 -0.02 1.05  -0.02   -0.02 1.00 -3.65 3.63  7.28  0.04     0.23
X16   16 1000  0.00 1.02   0.06    0.01 0.98 -3.46 3.12  6.58 -0.08     0.15
X17   17 1000  0.01 1.00   0.01    0.00 0.95 -3.12 2.78  5.90  0.05     0.01
X18   18 1000 -0.04 1.00  -0.07   -0.05 0.97 -3.38 3.47  6.86  0.09    -0.01
X19   19 1000  0.02 1.00   0.05    0.01 0.92 -2.92 3.33  6.25  0.06     0.10
X20   20 1000  0.02 1.03   0.00    0.02 1.00 -3.67 3.92  7.59  0.06     0.18
      se
X1  0.03
X2  0.03
X3  0.03
X4  0.03
X5  0.03
X6  0.03
X7  0.03
X8  0.03
X9  0.03
X10 0.03
X11 0.03
X12 0.03
X13 0.03
X14 0.03
X15 0.03
X16 0.03
```

```
X17 0.03
X18 0.03
X19 0.03
X20 0.03
```

This dataset includes 20 variables with 1,000 observations for each. These variables appear to be standardized, given the means close to zero and standard deviation approximately one. The range of the variables are also consistent with a standard normal distribution, with minimum and maximum roughly in line with the extreme 1% tails. The skewness and kurtosis are minimal, which is to be expected for approximately normally distributed and standardized data.

Question 2. Convert your raw data into a covariance matrix using the `cov()` function and then use the `solve()` function to invert the covariance matrix. `print()` your precision matrix as your answer to this question.

Answer

```r
Sigma <- cov(data_ggm)
Theta <- solve(Sigma)

print(Theta)
```

```
              X1           X2            X3            X4            X5
X1   1.140642203 -0.168585665 -0.1218757121 -0.0278852087 -0.0234309389
X2  -0.168585665  1.158202119 -0.0157569454 -0.1912457079  0.0753722014
X3  -0.121875712 -0.015756945  1.0305908743  0.0159144947 -0.0176737172
X4  -0.027885209 -0.191245708  0.0159144947  1.1067164060 -0.1908782047
X5  -0.023430939  0.075372201 -0.0176737172 -0.1908782047  0.9833954533
X6  -0.033949998 -0.168447495  0.0527974753  0.0229087352 -0.0060074213
X7  -0.131807163 -0.055096395  0.0026737553 -0.0040282594  0.0277464704
X8  -0.002907894  0.012314509  0.0235641251 -0.0268122753  0.0516045529
X9  -0.071669947 -0.160389614  0.0008209529  0.0246582825  0.0148011896
X10  0.027219854  0.014735357  0.0165726498 -0.0188103057  0.0004537419
X11  0.002599446 -0.035751236  0.0294500857 -0.0005272425  0.0457686502
X12 -0.165045540  0.041819699 -0.0841853397  0.0142758215  0.0080411705
X13 -0.013963537  0.001640164  0.0004129098  0.0051797950 -0.0053831782
X14 -0.003115909 -0.009786405 -0.0247401178 -0.1864489878 -0.0003899115
X15 -0.184454871  0.075406170  0.0371334954 -0.0779235285  0.0223776848
X16 -0.012701744  0.030860770  0.0455438005  0.0075790916 -0.0351792104
X17 -0.037607515 -0.122575915  0.0146443923  0.0115489443 -0.0018148738
```

X18	0.063930438	-0.018330704	0.0732467337	-0.0192641654	-0.1367583473
X19	-0.128701633	-0.001245002	-0.0723653983	0.0013392766	0.0036503055
X20	0.076570311	0.014743997	0.0620542776	-0.1987404340	0.0051041769

	X6	X7	X8	X9	X10
X1	-0.033949998	-0.131807163	-0.002907894	-0.0716699472	0.0272198535
X2	-0.168447495	-0.055096395	0.012314509	-0.1603896140	0.0147353570
X3	0.052797475	0.002673755	0.023564125	0.0008209529	0.0165726498
X4	0.022908735	-0.004028259	-0.026812275	0.0246582825	-0.0188103057
X5	-0.006007421	0.027746470	0.051604553	0.0148011896	0.0004537419
X6	1.140744331	-0.033705807	-0.085268704	0.0041453901	0.0136172727
X7	-0.033705807	1.127365651	-0.139447769	-0.0107729444	-0.1726196952
X8	-0.085268704	-0.139447769	1.046449861	0.0842772394	0.0098338362
X9	0.004145390	-0.010772944	0.084277239	1.0824853394	0.0315749386
X10	0.013617273	-0.172619695	0.009833836	0.0315749386	0.9805540497
X11	0.038938976	-0.109800608	-0.065765870	-0.0448124602	0.0065784031
X12	-0.040322474	-0.046741750	-0.051804578	-0.0042899813	0.0118238759
X13	-0.162309559	-0.030981867	0.045606290	-0.0201955941	-0.0230736068
X14	-0.031071866	-0.009730309	0.067201879	0.0789906743	-0.0541082427
X15	-0.014414575	-0.030446059	0.021927841	-0.0380867970	0.0311796536
X16	0.030501855	0.035371605	0.028420357	0.0147111336	-0.0495105853
X17	-0.011955708	0.018911261	-0.034449970	-0.0212665201	0.0186901241
X18	-0.077709587	0.028830925	0.014563081	-0.0602847173	-0.0472305967
X19	-0.020982350	-0.007517197	-0.006477156	0.0073093605	0.0193332782
X20	-0.062854001	0.023961539	0.061476929	-0.0074419844	-0.0175507900

	X11	X12	X13	X14	X15
X1	0.0025994460	-0.165045540	-0.0139635367	-0.0031159089	-0.184454871
X2	-0.0357512362	0.041819699	0.0016401642	-0.0097864048	0.075406170
X3	0.0294500857	-0.084185340	0.0004129098	-0.0247401178	0.037133495
X4	-0.0005272425	0.014275821	0.0051797950	-0.1864489878	-0.077923529
X5	0.0457686502	0.008041170	-0.0053831782	-0.0003899115	0.022377685
X6	0.0389389761	-0.040322474	-0.1623095595	-0.0310718662	-0.014414575
X7	-0.1098006081	-0.046741750	-0.0309818671	-0.0097303086	-0.030446059
X8	-0.0657658700	-0.051804578	0.0456062900	0.0672018789	0.021927841
X9	-0.0448124602	-0.004289981	-0.0201955941	0.0789906743	-0.038086797
X10	0.0065784031	0.011823876	-0.0230736068	-0.0541082427	0.031179654
X11	1.0534864680	0.011139830	-0.0070333793	-0.0627967927	-0.030643817
X12	0.0111398296	1.119217325	-0.0291399181	-0.0313772506	0.050230293
X13	-0.0070333793	-0.029139918	0.9570911754	0.0196635237	0.007370914
X14	-0.0627967927	-0.031377251	0.0196635237	0.9882528277	-0.062792453
X15	-0.0306438168	0.050230293	0.0073709142	-0.0627924530	0.967822140
X16	0.0335044019	-0.193775278	-0.0279329446	0.0274106634	0.010635762
X17	0.0289412687	0.036562430	0.0233251338	0.0391766939	-0.026738145
X18	-0.0289636152	-0.019342032	0.0498547306	-0.0160571406	-0.014964382

```
X19 -0.0324246166  0.040643072  0.0001138895  0.0048138300 -0.007364727
X20 -0.0188569312 -0.037268210  0.0015181391 -0.0196794903  0.069255437
             X16          X17          X18           X19          X20
X1  -0.012701744 -0.037607515  0.063930438 -0.1287016334  0.076570311
X2   0.030860770 -0.122575915 -0.018330704 -0.0012450023  0.014743997
X3   0.045543800  0.014644392  0.073246734 -0.0723653983  0.062054278
X4   0.007579092  0.011548944 -0.019264165  0.0013392766 -0.198740434
X5  -0.035179210 -0.001814874 -0.136758347  0.0036503055  0.005104177
X6   0.030501855 -0.011955708 -0.077709587 -0.0209823496 -0.062854001
X7   0.035371605  0.018911261  0.028830925 -0.0075171967  0.023961539
X8   0.028420357 -0.034449970  0.014563081 -0.0064771562  0.061476929
X9   0.014711134 -0.021266520 -0.060284717  0.0073093605 -0.007441984
X10 -0.049510585  0.018690124 -0.047230597  0.0193332782 -0.017550790
X11  0.033504402  0.028941269 -0.028963615 -0.0324246166 -0.018856931
X12 -0.193775278  0.036562430 -0.019342032  0.0406430723 -0.037268210
X13 -0.027932945  0.023325134  0.049854731  0.0001138895  0.001518139
X14  0.027410663  0.039176694 -0.016057141  0.0048138300 -0.019679490
X15  0.010635762 -0.026738145 -0.014964382 -0.0073647273  0.069255437
X16  1.010276654 -0.007430023  0.026484834 -0.0085591062  0.033679608
X17 -0.007430023  1.037132522 -0.045807770 -0.0334744101 -0.046677643
X18  0.026484834 -0.045807770  1.048193273 -0.0090251294  0.035859378
X19 -0.008559106 -0.033474410 -0.009025129  1.0229291466  0.031032517
X20  0.033679608 -0.046677643  0.035859378  0.0310325167  1.008704497
```

Question 3. Using the following equation, write an R function or hard-code a way to convert your precision matrix into the matrix of partial correlations.

$$\mathrm{Cor}(x_i, x_j | x_{\_\_-i,-j}) = -\frac{k_{ij}}{\sqrt{k_{ii} \cdot k_{jj}}}$$

where $k_{ij} \in= K = \Sigma^{-1}$

> Show your R code and check your result using `cov2pcor()` from the `gRbase` package.

Answer

```r
precis_to_pcor <- function(m) {

  P <- matrix(NA, nrow(m), ncol(m))
```

```r
  for (i in 1:nrow(m)) {
    for (j in 1:ncol(m)) {
      if(i==j) {
        P[i, j] <- 1
      } else {
        k_ij <- m[i, j]
        k_ii <- m[i, i]
        k_jj <- m[j, j]
        P[i, j] <- -k_ij/sqrt(k_ii*k_jj)
      }
    }
  }
  return(P)
}

# Checking if both functions give the same results

sum(round(precis_to_pcor(Theta), 3) == round(gRbase::cov2pcor(Sigma), 3)) == length(Sigma)
```

```
[1] TRUE
```

```r
# They match!
```

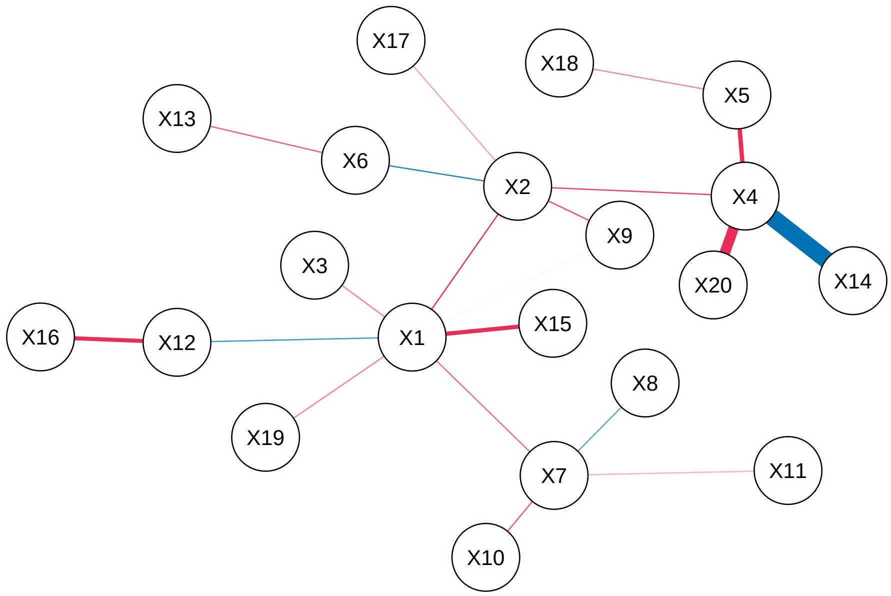
Question 4. Estimate your graph using EBICglasso and with the NHST graph. Print your results.

Answer

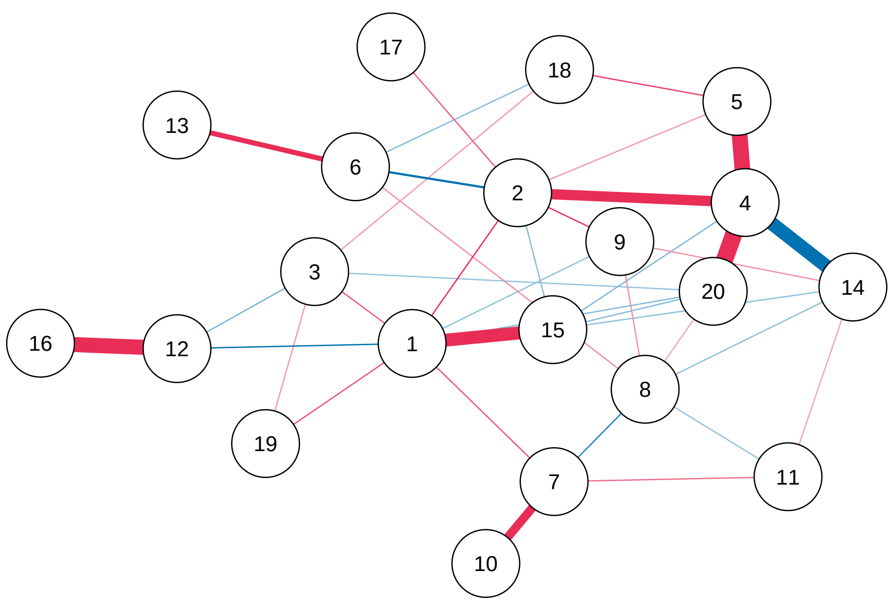
```r
NHSTgraph <- ggm_inference(data_ggm, alpha = 0.05,progress=FALSE)

L <- averageLayout(Sigma, NHSTgraph$wadj)

gEBIC <- qgraph(Sigma,
            graph = "glasso",
            sampleSize = nrow(data_ggm),
            layout = L,
            tuning = 0.5,
            edge.color = c("#0072B2", "#e82d57"))
```




```r
gNHST <- qgraph(NHSTgraph$wadj,
                sampleSize = nrow(data_ggm),
                layout = L,
                edge.color = c("#0072B2", "#e82d57"))
```



***Question 5*. What paths are similar between the two models? Which are different? Are you inclined to trust one network more than the other? Why?**

Answer

Most of the strongest edges are present in both graphs. Conversely, many weaker edges in the NHST graph, displayed as faint lines, were shrunken to zero in the EBIC estimation. The relationships estimated with these two methods are quite similar, and the choice of one over the other depends on the preference for the level of sparsity. The EBIC model (particularly with the tuning parameter set to 0.5) may be preferred when the research question is less concerned with detecting small effect sizes. That is, when increased specificity is prioritized over sensitivity. In contrast, the NHST network includes all the connections identified by EBIC, along with additional weaker edges. This network may be preferred when the goal is to maximize sensitivity rather than specificity.