**Description of R code, MATLAB code, and data sets for:**

**Disentangling relationships in symptom networks using matrix permutation methods**

The seriation.zip file contains the Fortran, R, and MATLAB programs, as well as the data sets discussed in the manuscript. This document presents these files, describes how to run the R scripts, and provides three examples. All of our analyses were completed using a PC with a Windows 10 operating system. The Fortran executable (.dll) files were compiled on this system, and *they will not run on a Mac*. However, we do provide the Fortran source codes, which should enable Mac users to compile and implement them on that platform.

**I. Data Sets**

There are four plain text data sets corresponding to the networks examined in the paper:

(i) foodpref.prn – the 5 × 5 food-item asymmetric preference matrix (from Table 1)

(ii) foodsim.prn – the 5 × 5 food-item symmetric similarity matrix (from Table 2)

(iii) han.prn – the 19 × 19 Han depression symmetric network matrix (from Table 3)

(iv) alcohol.prn – the 11 × 11 alcohol use asymmetric matrix (from Table 5)

**II. Fortran Source Codes (.for) Files**

There are 6 Fortran source code (.for) files.

(i) dpdom.for – maximize the dominance index using dynamic programming

(ii) dprob.for – minimize the Robinson index using dynamic programming

(iii) sadom1.for – maximize the dominance index using simulated annealing

(iv) sarob1.for – minimize the Robinson index using simulated annealing

(v) subsequencedom.for – extract the tau largest subsets with perfect dominance

(vi) subsequencepath.for – extract the tau largest paths with monotonic patterning

**III. Fortran Executable (.dll) Files**

There are 6 Fortran executable (.dll) files. These are called within the R scripts and, therefore, they are transparent to the user. Again, the dll files were created on a PC using Windows 10. *They will not work on a Mac*.

(i) dpdom.dll – maximize the dominance index using dynamic programming

(ii) dprob.dll – minimize the Robinson index using dynamic programming

(iii) sadom1.dll – maximize the dominance index using simulated annealing

(iv) sarob1.dll – minimize the Robinson index using simulated annealing

(v) subsequencedom.dll – extract the tau largest subsets with perfect dominance

(vi) subsequencepath.dll – extract the tau largest paths with monotonic patterning

**IV. R Scripts**

There are 6 R scripts that call the Fortran .dll’s to run the algorithms.

(i) dpdom.R – maximize the dominance index using dynamic programming

(ii) dprob.R – minimize the Robinson index using dynamic programming

(iii) sadom1.R – maximize the dominance index using simulated annealing

(iv) sarob1.R – minimize the Robinson index using simulated annealing

(v) subsequencedom.R – extract the tau largest subsets with perfect dominance

(vi) subsequencepath.R – extract the tau largest paths with monotonic patterning

**V. MATLAB (.m) Files**

There are 6 MATLAB m-files that can run the algorithms in MATLAB.

(i) dpdom.m – maximize the dominance index using dynamic programming

(ii) dprob.m – minimize the Robinson index using dynamic programming

(iii) pidom.m – maximize the dominance index using pairwise interchange

(iv) pirob.m – minimize the Robinson index using pairwise interchange

(v) subsequencedom.m – extract the tau largest subsets with perfect dominance

(vi) subsequencepath.m – extract the tau largest paths with monotonic patterning

**VI. Some Implementations Guidelines**

(a) It is assumed this ‘seriation’ folder is downloaded to a hard drive with the reference directory “c:/seriation”. If you use a different folder name, then it will be necessary to change the directory in the ‘dyn.load’ call in the R scripts accordingly.

(b) The MATLAB codes are appreciably slower than the Fortran codes running in R.

(c) The computation time for the dynamic programming algorithms roughly doubles for each one-unit increase in *p* (the dimensions of the network matrix)

(d) The dynamic programming algorithms are constrained by computer memory (RAM). For most personal computers, the upper limit will be somewhere between *p* = 25 and *p* = 30.

(e) The simulated annealing heuristics implemented in R are better than the pairwise interchange heuristics implemented in MATLAB. These algorithms can be useful for networks where *p* > 30. They should scale for ranges of *p* up to about 200.

**VII. Running the R Scripts – Example 1 (dprob.R)**

*Step 0.* Open the R console to enter commands.

*Step 1*. Assure that the Fortran executable file (dprob.dll) is in the ‘seriation’ directory. Alternatively, a user may select a different directory, but the directory call in the ‘dyn.load’ line will have to be adjusted accordingly.

*Step 2*. From the ‘file’ menu in R, select ‘Source R Code’, and then select ‘dprob.R’ from the appropriate directory.

*Step 3*. Read in the network matrix, ‘han.prn’ to variable ‘A’ and use the ‘as.matrix’ command to make sure it is acknowledged as double precision.

*Step 4*. Type dprob(A) at the prompt.

*Step 5*. View the main outputs: (i) the value of the Robinson index (robidx), (ii) the total time (ttime), and, most importantly, the optimal permutation (x).

Following this process should display the following output in R:

R version 3.6.3 (2020-02-29) -- "Holding the Windsock"

Copyright (C) 2020 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

> source("C:\\SERIATION\\dprob.R")

> A <- read.table("c:/seriation/han.prn")

> A <- as.matrix(A)

> dprob(A)

> robidx

[[1]]

[1] 100.77

> ttime

[[1]]

[1] 1.25

> x

[,1]

[1,] 1

[2,] 15

[3,] 17

[4,] 19

[5,] 18

[6,] 16

[7,] 8

[8,] 6

[9,] 9

[10,] 10

[11,] 14

[12,] 7

[13,] 4

[14,] 2

[15,] 5

[16,] 3

[17,] 13

[18,] 12

[19,] 11

**VIII. Running the R Scripts – Example 2 (dpdom.R)**

*Step 0.* Open the R console to enter commands.

*Step 1*. Assure that the Fortran executable file (dpdom.dll) is in the ‘seriation’ directory. Alternatively, a user may select a different directory, but the directory call in the ‘dyn.load’ line will have to be adjusted accordingly.

*Step 2*. From the ‘file’ menu in R, select ‘Source R Code’, and then select ‘dpdom.R’ from the appropriate directory.

*Step 3*. Read in the network matrix, ‘alcohol.prn’ to variable ‘A’ and use the ‘as.matrix’ command to make sure it is acknowledged as double precision.

*Step 4*. Type dpdom(A) at the prompt.

*Step 5*. View the main outputs: (i) the value of the dominance index (domidx), (ii) the total time (ttime), and, most importantly, (iii) the optimal permutation (x).

Following this process should display the following output in R:

R version 3.6.3 (2020-02-29) -- "Holding the Windsock"

Copyright (C) 2020 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

> source("C:\\SERIATION\\dpdom.R")

> A <- read.table("c:/seriation/alcohol.prn")

> A <- as.matrix(A)

> dpdom(A)

> domidx

[[1]]

[1] 46.68

> ttime

[[1]]

[1] 0

> x

[,1]

[1,] 4

[2,] 7

[3,] 3

[4,] 9

[5,] 5

[6,] 11

[7,] 6

[8,] 1

[9,] 8

[10,] 10

[11,] 2

**IX. Running the R Scripts – Example 3 (sarob1.R)**

*Step 0.* Open the R console to enter commands.

*Step 1*. Assure that the Fortran executable file (sarob1.dll) is in the ‘seriation’ directory. Alternatively, a user may select a different directory, but the directory call in the ‘dyn.load’ line will have to be adjusted accordingly.

*Step 2*. From the ‘file’ menu in R, select ‘Source R Code’, and then select ‘sarob1.R’ from the appropriate directory.

*Step 3*. Read in the network matrix, ‘han.prn’ to variable ‘A’ and use the ‘as.matrix’ command to make sure it is acknowledged as double precision.

*Step 4*. Select the desired number of restarts, R (10 more than sufficient).

*Step 5*. Type dprob(A) at the prompt.

*Step 6*. View the main outputs: (i) the value of the Robinson index (robidx), (ii) the total time (ttime), (iii) the percentage of restarts for which the best value of the Robinson index was found (attract) and, most importantly, (iv) the optimal permutation (x).

Following this process should display the following output in R:

R version 3.6.3 (2020-02-29) -- "Holding the Windsock"

Copyright (C) 2020 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

> A <- read.table("c:/seriation/han.prn")

> A <- as.matrix(A)

> R <- 10

> source("C:\\SERIATION\\sarob1.R")

> sarob1(A,R)

> robidx

[[1]]

[1] 100.77

> attract

[[1]]

[1] 100

> ttime

[[1]]

[1] 7.625

> x

[,1]

[1,] 11

[2,] 12

[3,] 13

[4,] 3

[5,] 5

[6,] 2

[7,] 4

[8,] 7

[9,] 14

[10,] 10

[11,] 9

[12,] 6

[13,] 8

[14,] 16

[15,] 18

[16,] 19

[17,] 17

[18,] 15

[19,] 1