**JASA ACS Reproducibility Initiative - Author Contributions Checklist Form**

## Data

**Abstract (Mandatory)**

There are two sets of simulated data, each one corresponds to scenarios 1 and 2 described in the paper and contain four replicates. Each replicate is a collection of three datasets containing: genotypic data (QTL genotypes), phenotypes along with additive genetic effects, residuals generation and other columns described below, and a file with individual QTL effects. Datasets simulated under scenario 1 contain 200 individuals, while those simulated under scenario 2 contain 380 individuals, the two scenarios considered 300 QTL, i.e., 300 regression variables. For further details on the differences between scenario 1 and 2 see the manuscript. On the other hand, the real dataset contains centered phenotypes corresponding to fertility, genotypes for 300 SNP that were found to be associated to fertility in Han and Peñagaricano (2016). This is a subset of the original one obtained after discarding half-sib families with less than 5 individuals. This editing process was necessary due to the fact that our frequentist methods require grouping individuals in families.

**Availability (Mandatory)**

Simulated datasets are available at xxxxxxxxxxxxxxxxx. Regarding real data, it is private and consequently, it cannot be made publicly available. Although nowadays some institutions have open data policies for genomic records, it is not the case of the company owning the individuals or their genotypes.

**Description (Mandatory if data available)**

* These datasets were simulated by the authors of the paper using free software and are available at S. Rahman’s Github personal website.
* All simulated datasets are publicly available.
* Simulated data are available at: Link to data/repository (e.g., *dataverse.org*, *datadryad.org*, *zenodo*, *Github* [only feasible for small datasets]
* Data were simulated using software QMSim (Sargolzaei and Schenkel, 2013) and R package BDgraph (Mohammadi and Wit, 2015). QMSim was used to simulate genotypes, whereas BDgraph was used to simulate the precision matrix of allele effects. There are two sets of simulated data, each one corresponds to scenarios 1 and 2 described in the paper and contain five replicates. Each replicate is a collection of three datasets containing genotypic data (QTL genotypes), phenotypes along with additive genetic effects, residuals generation and other variables described below, and individual QTL effects. Datasets simulated under scenario 1 contain 200 individuals, while those simulated under scenario 2 contain 380 individuals, the two scenarios considered 300 QTL, i.e., 300 regression variables. For further details on the differences between scenario 1 and 2 see the manuscript.
* File format: .txt
* Data dictionary.
  + Datasets Pop1\_data\_001,…, Pop1\_data\_004:
    - Progeny: Individual ID.
    - Sire: Individual’s sire ID.
    - Dam: Individual’s dam ID.
    - Sex: Individual’s gender, male (M) or female (F).
    - G: Generation.
    - NMPrg: number of male progenies.
    - NFPrg: number of female progenies.
    - F: inbreeding coefficient
    - Homo: percentage of homozygous loci.
    - Phen: Phenotype, i.e., record.
    - Res: Residual
    - Polygene: It corresponds to a genetic effect different from the one corresponding to simulated QTL. In this simulation, this effect was null.
    - QTL: Additive genetic effect coming from simulated QTL’s, in this simulation, the total additive genetic effect came from simulated QTL. It corresponds to the dot product between the vector containing coded genotypes (see manuscript for details) and the vector of individual QTL effects.
  + Datasets QTL\_effects\_001,…,QTL\_effects\_004: These files contain a single column corresponding to individual QTL effects.
    - V1: Individual QTL effects.
  + Datasets Pop1\_qtl\_001,…,Pop1\_qtl\_004: These files do not have column names. The firs column contains individual’s ID and it is followed by 600 numbers. Each consecutive pair corresponds to a QTL and codes the genotype; thus, 1 1 is a homozygous genotype for the first allele, 2 2 a homozygous genotype for allele 2, finally 1 2 and 2 1 indicate heterozygous genotypes.

**Optional Information (complete as necessary)**

Unique identifier / DOI

## Code

**Abstract (Mandatory)**

The code used to perform the analyses described in the manuscript consists of seven R programs. There is a file to perform data analysis based on each one of our methods. Moreover, some files perform the required computations to obtain files that are used as inputs in other programs. Codes also include the computations used in model performance assessment, as described in the paper.

This set of programs permits to reproduce results shown in Table 1 and supplementary Table 1. Notice that these tables involve four replicates under two scenarios and that method Bayes DAG-Sel only works when there are more data than markers, that is, under scenario 2. Therefore, the corresponding code has to be run four times, while the remaining have to be executed eight times. For the sake of clarity, all codes are commented, each

**Description (Mandatory)**

* Code is delivered as R scripts.
* Licensing information (default is MIT License)
* All code files are available at: xxxxxxx
* Version information (e.g., for a Git repository, the number or branch+commit)
* Supporting software requirements (e.g., libraries and dependencies, including version numbers for R and Python packages)

## Instructions for Use

**Reproducibility (Mandatory)**

* This set of programs permits to reproduce results shown in Table 1 and supplementary Table 1.
* Since there are five different methods developed in the paper: Glasso-EM, CONCORD-EM, CSCS-EM, Bayes G-Sel and Bayes DAG-Sel, there are several files to implement each one of them. Thus, there are seven different code files, there is a file to perform data analysis based on each one of our methods. Analyses performed by each file are described below:

**GLasso\_EM**: Implements the GLasso-EM method and computes some parameters measuring its performance.

**CONCORD\_EM**: Implements the CONCORD-EM method and computes some parameters measuring its performance.

**CSCS\_EM**: Implements the CSCS-EM method and computes some parameters measuring its performance.

**Model\_Sel\_Bayes\_SSS**: Implements the stochastic short-gun search algorithm to carry out Bayesian graphical model selection under a G-Wishart prior as described in the manuscript.

**Model\_Sel\_Bayes\_DAG\_SSS**: Implements the stochastic short-gun search algorithm to carry out Bayesian graphical model selection under a DAG-Wishart prior as described in the manuscript.

**Gibbs\_Sampler\_General\_Graph**: Performs Bayesian estimation of the precision matrix under a Gaussian concentration graph model once an undirected graph has been selected using Bayes G-Sel.

**Gibbs\_sampler\_DAG\_Wishart**: Performs Bayesian estimation of the precision matrix under a Gaussian DAG model once a DAG has been selected using Bayes DAG-Sel.

The following is the way in which Table 1 and Supplementary Table 1 may be reproduced:

* Run GLasso\_EM for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce rows 1-4 of both tables.
* Run CONCORD\_EM for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce rows 5-8 of both tables, as well as the 15 input files for program Model\_Sel\_Bayes\_SSS.
* Run CSCS\_EM for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce rows 9-12 of both tables, as well as the 15 input files for program Model\_Sel\_Bayes\_DAG\_SSS.
* Run Model\_Sel\_Bayes\_SSS for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce row 13 of Table 1. Moreover, this program yields the input file (selected graph) for file Gibbs\_Sampler\_General\_Graph
* Run Model\_Sel\_Bayes\_DAG\_SSS for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce row 14 of Table 1. Moreover, this program yields the input file (selected graph) for file Gibbs\_sampler\_DAG\_Wishart.
* Run Gibbs\_Sampler\_General\_Graph for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce row 13 of Supplementary Table 1.
* Run Gibbs\_sampler\_DAG\_Wishart for each replicate, get the empirical mean and standard deviation of each parameter, do it for both scenarios, this would reproduce row 14 of Supplementary Table 1.

In the following, each code file will be described.

**GLasso\_EM**

-The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details in the manuscript).

-The next part creates training and testing (as indicated in the manuscript).

-The following lines create a function called “Def.Graph.Band” which is an auxiliary function to be used later.

-Then, there are two functions to create adjacency matrices.

-The next section groups data according to half-sibs families.

- Function “Rand.Sparse” is used to create an initial value for marker effects concentration matrix.

- Function “Bibi.GEMLASSO” is the main function of the code, it implements the GLasso-EM method described in the paper.

- Function “F” performs tuning of the penalty parameter using the four criteria described in the paper. Fifteen values of the penalty parameter are considered.

- The next section performs computations required to assess model performance. Two kinds of performance are assessed, predictive performance and graph selection performance.

**CONCORD\_EM**

-The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details about this matrix in the manuscript).

-The next part creates training and testing sets (as indicated in the manuscript).

-Function custConcord1 performs CONCORD using a covariance matrix as an input.

- Adj2 builds an adjacency matrix.

- The next part of the code groups data according to half-sibs families.

- Function Rand.Sparse is used to create an initial value for marker effects concentration matrix.

- “Bibi.GEMCONCORD” is the main function, it implements the CONCORD-EM method.

- Function “F” performs tuning of the penalty parameter using the four criteria described in the paper. Fifteen values of the penalty parameter are considered, the resulting graphs are also used as inputs in program Model\_Sel\_Bayes\_SSS.

-The next section performs computations required to assess model performance. Two kinds of performance are assessed, predictive performance and graph selection performance.

**CSCS\_EM**

-The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details about this matrix in the manuscript).

-The next part creates training and testing sets (as indicated in the manuscript).

- Then, there are two functions to create adjacency matrices.

-The following lines create a function called “Def.Graph.Band” which is an auxiliary function to be used later.

- The next part of the code groups data according to half-sibs families.

- Function “Rand.Sparse” is used to create an initial value for marker effects concentration matrix.

- Function “CSCS2” implements the CSCS method.

- “Bibi.GEMDAG” is the main function, it implements the CSCS-EM method.

- Function “F” performs tuning of the penalty parameter using the four criteria described in the paper. Fifteen values of the penalty parameter are considered, the resulting graphs are also used as inputs in program Model\_Sel\_Bayes\_DAG\_SSS

-The next section performs computations required to assess model performance. Two kinds of performance are assessed, predictive performance and graph selection performance.

- Finally, there are some lines to save the 15 DAG’s obtained after running function “F”.

**Model\_Sel\_Bayes\_SSS**

This program takes as input each one of the 15 graphs obtained when running the CONCORD-EM method with different penalty parameters. Hereinafter, these will be referred to as input graphs. Thus, the SSS algorithm described in Ben-David et al. (2015) can be run in parallel because computations starting with each one of this 15 graphs can be carried out at the same time.

**-** The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details about this matrix in the manuscript).

**-**Then, phenotypes and total additive genetic affects are read.

-The following part splits data into training and testing (as indicated in the manuscript).

-Then, the ith graph (i=1,2,…,15) is read.

-The next lines correspond to functions creating adjacency matrices.

-The next function is called “MCint2” and it performs Monte Carlo Integration.

-Then, there is a collection of four connected functions to compute the posterior graph score used to select a graph.

-The following lines select the graph with largest score. Note that this corresponds to the graph with the largest score for the ith initial graph; therefore, after running this program for each one of the 15 input graphs, the 15 scores of the 15 output graphs have to be compared in order to select an overall winner, i.e., the selected graph.

-Finally, graph selection performance is evaluated as described in the paper.

**Model\_Sel\_Bayes\_DAG\_SSS**

This program takes as input the 15 graphs obtained when running the CSCS-EM method with different penalty parameters. In this case, there is no need to run the SSS algorithm in parallel because (approximate) closed form expressions for scores are available.

**-** The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details about this matrix in the manuscript).

**-**Then, phenotypes and total additive genetic affects are read.

-The following part splits data into training and testing (as indicated in the manuscript).

-The following lines read the 15 DAG’s obtained after running CSCS-EM method, i.e., after running program “CSCS\_EM” described above.

-The next lines correspond to a set of functions to compute the posterior score described in the manuscript.

-Then, the SSS algorithm is implemented. Notice that this is not the same code used in program “Model\_Sel\_Bayes\_SSS” because the scores are computed in a different way. Moreover, the code contains a new function called “Largestscoregraph” which yields the selected graph, that is, the overall winner. As explained above, this is so because score computation is much faster under Bayes DAG-Sel and, as a consequence, there is no need for parallel computation.

-Finally, the selected graph is saved in the working directory.

**Gibbs\_Sampler\_General\_Graph**

This program takes as input the adjacency matrix corresponding to the graph selected by Bayes G-Sel, that is, it is used after running Model\_Sel\_Bayes\_SSS**.**

-The first function reads the file with genotypes, it transforms the output format of QMSim software into the design matrix used in whole genome regression (see details about this matrix in the manuscript).

- Then, phenotypes and total additive genetic affects are read.

-The following part splits data into training and testing (as indicated in the manuscript).

-The next section creates a function to sample from the full conditional distribution under a G-Wishart prior.

-The following part creates the main function of the code, it is called “BibiGraph.W” and implements the Gibbs sampler described in the paper.

- The final part of this code is devoted to evaluating graph selection and predictive performance as described in the paper.

**Gibbs\_sampler\_DAG\_Wishart**

This program is used after a DAG has been selected, i.e., after running xxxxxx. Therefore, it takes the adjacency matrix corresponding to the selected DAG as an input.

- Function create.W reads genotypes and builds the design matrix used in whole-genome regressions (see details about this matrix in the manuscript).

- The next section splits the dataset into training and test sets as defined in the manuscript.

- Then, a file called “Nodes” which contains the edges of a given DAG.

-Function SampLD.DAG yields random samples from the full conditional distribution under a DAG-Wishart prior.

-The next function builds adjacency matrices.

-The next section contains the main function “BibiDAG.W” which implements the Gibbs sampler under a DAG-Wishart prior.

-The following portion of the code is devoted to evaluating graph selection and predictive performance as described in the paper.

* Expected run-time of the workflow (and information about particularly slow steps in workflow, if any). If possible, give the approximate time to run on a standard desktop machine.

## Notes

Other relevant information, in particular how reviewers can access the data and code if not yet made publicly available.