Author Contributions Checklist Form

This form documents the artifacts associated with the article (i.e., the data and code supporting the computational findings) and describes how to reproduce the findings.

# Part 1: Data

This paper **does not** involve analysis of external data (i.e., no data are used or the only data are generated by the authors via simulation in their code).

I certify that the author(s) of the manuscript have legitimate access to and permission to use the data used in this manuscript.

## Abstract

The following three files contain the dataset we analyzed in Section 8. The original data was obtained in the following paper

Bloom, Joshua S., Ian M. Ehrenreich, Wesley T. Loo, Thúy-Lan Võ Lite, and Leonid Kruglyak. "Finding the sources of missing heritability in a yeast cross." *Nature* 494, no. 7436 (2013): 234-237.

Particularly, we conduct our real data analysis with the following three files.

1. BYxRM\_GenoData.txt contains the single nucleotide polymorphisms (SNPs) data for 1008 yeast segregants.

2. BYxRM\_PhenoData.txt contains the yeast colony sizes (outcome variables) at the end point, grown under 46 growth media.

3. ReduceInd.txt contains the SNP indexes selected by the paper (Bloom et. al., 2013).

## Availability

Data **are** publicly available

Data **cannot be made** publicly available

If the data are publicly available, see the *Publicly available data* section. Otherwise, see the *Non-publicly available dat*a section, below.

### Publicly available data

Data are available online at:

http://genomics-pubs.princeton.edu/YeastCross\_BYxRM/data.shtml

Data are available as part of the paper’s supplementary material.

Data are publicly available by request, following the process described here:

We have included the file ReduceInd.txt, which is shared by the authors of Bloom et. Al. (2013) to replicate their results.

Data are or will be made available through some other mechanism, described here:

### Non-publicly available data

Discussion of lack of publicly available data:

## Description

### File format(s)

CSV or other plain text:

Software-specific binary format (.Rda, Python pickle, etc.):

Standardized binary format (e.g., netCDF, HDF5, etc.):

Other (described here):

### Data dictionary

Provided by the authors in the following file(s):

Data file(s) is (are) self-describiing (e.g., netCDF files)

Available at the following URL:

### Additional information (optional)

# Part 2: Code

## Abstract

The R code replicates the simulation in Section 7 and real data results in Section 8 of the current paper.

## Description

### Code format(s)

Script files

R  Python  Matlab

Other:

Package

R  Python  MATLAB toolbox

Other:

Reproducible report

R Markdown  Jupyter notebook

Other:

Shell script

Other (described here):

### **Supporting software requirements**

Version of primary software used

R version 4.2.1

Libraries and dependencies used by the code

Glmnet version 4.1.4; CVXR version 1.0.11; intervals version 0.15.2; MASS version 7.3.57

### Supporting system/hardware requirements (optional)

### Parallelization used

No parallel code used

Multi-core parallelization on a single machine/node

Number of cores used:

Multi-machine/multi-node parallelization

Number of nodes and cores used:

### License

MIT License (default)

BSD

GPL v3.0

Creative Commons

Other (described here):

### Additional information (optional)

# Part 3: Reproducibility workflow

## Scope

The provided workflow reproduces:

Any numbers provided in text in the paper

The computational method(s) presented in the paper (i.e., code is provided that implements the method(s))

All tables and figures in the paper

Selected tables and figures in the paper, as explained and justified here:

All tables and figures except for Figures 1 and 2 are reproduced here. Figure 1 is an illustrative figure of the maximin effect, and Figure 2 is an illustrative figure of the sampling method.

## Workflow details

### Location

The workflow is available:

As part of the paper’s supplementary material

In this Git repository:

Other:

### Format(s)

Single master code file

Wrapper (shell) script(s)

Self-contained R Markdown file, Jupyter notebook, or other literate programming approach

Text file (e.g., a readme-style file) that documents workflow

Makefile

Other (more detail in 'Instructions' below)

### Instructions

The provided workflow reproduces all the tables and figures in section 7 and section 8.

## Section 7

#### Table 1

In the folder "./code/simulations", run the script:

1. "I0to6.R" returns the RData files for (I0) (I1) ... (I6), by setting the argument "setting" as {0,1,2,3,4,5,6};

2. "I7&8.R" returns the RData files for (I7) and (I8), by setting the argument "setting" as {7,8};

3. "I9&10.R" returns the RData files for (I9) and (I10), by setting the argument "setting" as {9,10}.

After obtaining the RData file, run the function “tab1()” in the script “figs&tabs.R” to reproduce the corresponding row in the Table 1. For example, after running “I0to6.R” with argument “setting=0”, it will save the RData file for setting 0. Then apply “tab1()”, we will get the information for setting 0 in the Table 1.

#### Figure 3

In the folder "./code/simulations", run the script:

1. "I0to6.R", set "setting=1"

2. "I7&8.R", set "setting=8"

3. "I9&10.R", set "setting=10"

After obtaining the RData file for each setting, run the function “fig3()” in the script “figs&tabs.R” to reproduce the one of left, middle, or right two subfigures. For example, by running “I0to6.R” with argument “setting=1”, we obtain its corresponding RData, then run “fig3(filename, pos=’left’)” to draw the left two subfigures.

#### Table 2

In the folder "./code/simulations", run the corresponding script to each setting.

In these scripts, set the arguments "n" across {200,300,500}. For each n, it will return a RData file, then apply “tab2()” in the script “figs&tabs.R”. We will get the information including coverage error, length ratio. Take average over varying n, we obtain the Table 2.

#### Figure 4

In the folder "./code/simulations", run script "Setting1.R".

In the script, set the arguments "n" across {100, 200, 300, 500}. For each n, it will return a RData file, then apply “fig4()” in the script “figs&tabs.R”. We will get the information including coverage, CI length, and length ratio for different penalty delta. Next, draw Figure 4 with varying n.

#### Figure 5

In the folder "./code/simulations", run script "Setting3a" and "Setting3b".

In each script,

(1)the argument "setting=1": algorithm "CS Known"

(2)the argument "setting=2": algorithm "CS n=500"

(3)the argument "setting=3": algorithm "No CS N=500"

Then summarize the results in the same approach explained in "Figure 4", except for applying “fig5()” in the script “figs&tabs.R”

## Section 8

The script “./code/realdata.R” contains thorough instructions on how to replicate the tables and figures in the real data application. By following the steps outlined in the script, users can obtain the desired results.

Expected run-time

Approximate time needed to reproduce the analyses on a standard desktop machine:

<1 minute

1-10 minutes

10-60 minutes

1-8 hours

>8 hours

Not feasible to run on a desktop machine, as described here:

### Additional documentation (optional)

The code provided to replicate the results represents the initial implementation of our algorithms. The most recent version has been developed as the “MaximinInfer” R package available on CRAN or at <https://github.com/zywang0701/MaximinInfer>

# Notes (optional)