Findr User Manual

Lingfei Wang

The Roslin Institute, University of Edinburgh

For version 0.2.0

Contents

1 Introduction

1.1 Basic information

Findr (Fast Inference of Networks from Directed Regulations) is a C library for the inference of gene regulatory network structure. Interfaces for binary and python, as well as a standalone package for R are provided for major functions. Currently, only the causal inferences of pairwise gene regulations from genotype and gene expression datasets or from gene expression dataset alone are implemented. More functions will be added in later releases. Methodologies can be found in publication [1]. The latest version of this documentation can be downloaded here.

1.2 Highlights

Findr features native shared memory multithreading with OpenMP. Calculations are performed in C language as well as by invoking GSL (GNU Scientific Library). The statistical advances in [1] have enabled analytical calculation of null histograms. They together provide improved time and memory efficiencies for Findr. As demonstrated in [1] with yeast dataset, Findr is thousands of times faster than Trigger, an earlier implementation of similar methods [2].

1.3 License

Findr is licensed under GNU Affero General Public License, Version 3 (AGPL-3), which can be downloadable from https://www.gnu.org/licenses/agpl-3.0. GSL is separated licensed by its authors under GPL.

1.4 Feedback

The authors welcome feedbacks in any form and perspective. No matter you have enquiries, suggestions, or critics, in installation, usage, future development, methodology, or collaboration, you can reach us at: Lingfei.Wang.github@outlook.com.

2 Findr library

Findr library is the core of computation for the software. The binary and python interfaces require Findr to be pre-installed, and call Findr during computation. The Findr R package contains a copy of Findr library by itself and does not need to install Findr library separately. Users of Findr R package can jump to Section 3.3.

2.1 Installation

The Findr library and its interfaces can function without installation after some adjustments. However we recommend installation following the steps below. If you encounter any errors during installation that can't be resolved independently, please contact us with detailed error messages.

- **Prerequisites:** Findr is written in C and by nature supports Windows, Linux, and Mac OS X platforms. The user also needs the following to build Findr library:
  - A recent GCC compiler that supports OpenMP interface.
  - An up-to-date GNU make utility.
  - GNU Scientific Library (GSL). GSL must be installed or placed in a searchable location for include and link.

  To build Findr on Windows, Cygwin or MinGW is an option but above prerequisites must also be satisfied. However, the pre-built version of Findr based on MinGW from MSYS2 is recommended and is available for download at https://github.com/lingfeiwang/findr/releases. The pre-built version can be installed by running 'install.cmd' as administrator.
- **Download:** Findr can be downloaded with git from github with the command:

```
git clone https://github.com/lingfeiwang/findr.git
```

Alternatively it can also be downloaded from the webpage https://github.com/lingfeiwang/findr.

- **Customize makefile (optional):** You can customize how Findr is built and installed by editing its Makefile. This is useful for example when the user provides custom GSL location. Another possibility is the user does not have administrative priviledge on the computer, so a local installation is desired. In the latter case, the user must make sure the custom installation can be correctly located by binary and/or python interfaces, e.g. with environmental variables.

- **Build:** The source can be built with one line of command after downloading, in the directory it is downloaded or unzipped:

```
make
```

- **Install:** If you can gain superuser privilege, a normal install is recommended. This is also one line of command:

```
sudo make install
```

Otherwise, the user is suggested to change install location by editing the 'PREFIX' variable of Makefile. Then use the following command to install without superuser privilege:

```
make install
```

The same PREFIX should then be set for binary interface.

We strongly advise against keeping multiple versions of Findr on the same machine, including installed and not installed versions, as this may confuse interfaces of which library to use.

- **Update:** To update Findr, simply download a newer version and install it to the same location. This will replace the older version.

- **Uninstall:** If you ever need to remove Findr from your machine permanently, this can be done with the command in the directory containing Findr source:

```
sudo make uninstall
```

The 'PREFIX' variable should be adjusted beforehand if you want to uninstall from a custom location. Preferrably, the source code should have the same version as the installed library. Sudo may be needed depending on the location of installation.

2.2 General Concepts

- **Library initialization:** The library must be initialized before called. During initialization, Findr initializes GSL, and processes the following initialization parameters:
  - loglv: The logging level of Findr. For details, see **Logging** below.
  - rs: Initial seed for random number generator. By default, uses the current time.
  - nth: Maximum number of parallel threads for computation. For best performance, it should not exceed the number of CPU cores. Its default value is the number of cores automatically detected, which is not always accurate.

- **Logging:** Findr has 13 logging levels:
  - Critical (0): The calculation has to halt due to critical error(s). Output is invalid.
  - Error (1-3): Part of function is lost in calculation due to uncritical error(s). Part of output is invalid.
  - Warning (4-6): Calculation is successful but Findr has encountered abnormal data. Findr attempted to correct anomalies but errors may have been introduced in output. User is advised to inspect their input.
  - Info (7-9): Running information of Findr.
  - Debug (10-12): More verbose running information.

  During initialization, the user is requested to input the logging level (loglv), so that only message levels not greater than loglv would printed. Findr library does not have a default log level, but in binary, python interfaces and R package, the default log level is 6.

2.3 Using Findr library

Findr library is well documented in its header files and source code. Developers are encouraged to build programs that call Findr, and to modify Findr freely for specific needs under current license.

3 Interfaces and packages

We provide binary and python interfaces for the major functions of Findr. Incompatible versions between interfaces and the library can create potential errors and limit reproducibility in data analysis. For this reason, users are advised to install/update interface(s) immediately after the library, and ensure they have the same version. Version differences are warned during interface execution. The R package of Findr includes the library of the same version, so the issue is absent.

The binary interface, python interface, and R package reveals the same set of functionality of Findr, under different naming and language conventions. User of a specific interface or package can proceed to installation and usage instructions of the respective section below, and then look for desired functions in Section 4.

3.1 Binary interface

The binary interface follows the same build, install, and uninstall steps as Findr library itself, after downloading with the command:

```
git clone https://github.com/lingfeiwang/findr-bin.git
```

or from the webpage https://github.com/lingfeiwang/findr-bin.

After installation, type 'findr' would execute the binary. However, if Findr library is installed in a custom location, user should change the Makefile of binary interface correspondingly, such as the 'PREFIX' variable. If the binary interface is installed in a custom location, the user needs to add the install location to 'PATH' environmental variable.

The binary interface only provides minimal validity checks on input data, although errors in calculation are notified in logs (see Section 2.2). Binary interface should be invoked as:

```
findr loglv rs nth method method_args...
```

The first three parameters are Findr library initialization parameters, as explained in Section 2.2 and Section 4.1. For automatic configuration, input zeros for all of them. *Method* stands for the name of data analysis routine the user want to call, and *method_args* are its arguments which will be passed on to the method function. Available methods are listed in Section 4 in red.

Binary interface receives bulk data (vectors and row-major matrices) as input from files, and exports bulk output data to files. Both raw and csv input formats are accepted. By default, raw format is used where float type data are in 32-bit raw format, and genotype data are in 8-bit unsigned integer format. To use csv format for all input and output files, append '_csv' to the method name. Csv format files should contain one row in each line, use space as column separator, and not contain row or column names.

On Windows, we recommend downloading the pre-built binary interface at https://github.com/lingfeiwang/findr-bin/releases. The pre-built version can be installed by running 'install.cmd' as administrator.

3.2 Python interface

The python interface requires numpy, as it uses numpy.ndarray as input and output format. It also requires pre-installed Findr library. To install Findr's python interface, execute the following from command line (with proper privilege):

```
pip install findr
```

or download and install it from github with

```
git clone https://github.com/lingfeiwang/findr-python.git
cd findr-python
sudo python setup.py install
```

Alternatively, the python interface can be downloaded from https://github.com/lingfeiwang/findr-python.

To use Findr, first obtain an initialized library object with:

```
import findr
l=findr.lib(path=None,loglv=6,rs=0,nth=0)
```

Path gives the exact location to search for Findr shared library in addition to default locations. Other three optional parameters are for library initialization and can be omitted for ordinary use. After above lines, Findr routines can be called as l's method. Exposed python functions of Findr are listed in Section 4, in which python method names are in green.

Bulk input and output data are formatted in numpy.ndarray for vectors and matrices, with 32-bit float for expression data and 8-bit unsigned integer for genotype data by default. Function returns are in dictionary type, in which each key contains an independent returned object. All functions share a returned key 'ret', which indicates a success in calculation with 0 and failure otherwise.

3.3 R package

Findr's R package is a standalone package which includes Findr library and GSL. It is locally compiled by *nix toolset, such as GCC and GNU make. On Windows, we recommend downloading the pre-built release of Findr's R package. Rterm on Windows supports logging, but

not RGui, due to compatibility issues. On RGui, only return value indicates if the execution is successful.

On Mac OS and Linux, the R package can be downloaded and installed with

```
git clone https://github.com/lingfeiwang/findr-R.git
cd findr-R
R CMD INSTALL findr
```

Alternative, it can be downloaded at https://github.com/lingfeiwang/findr-R. The prebuilt Windows release can be downloaded at https://github.com/lingfeiwang/findr-R/releases.

To use Findr, first initialize the library with:

```
library(findr)
findr.lib(loglv=6,rs=0,nth=0)
```

The three optional parameters are for library initialization and can be omitted for ordinary use. After above lines, Findr routines can then be called. Exposed R functions of Findr are listed in Section 4, in which R function names are in blue.

Bulk input and output data are formatted in R matrix or vector format, with double type for expression data and integer for genotype data. Function returns are in list type, in which each element contains an independent returned object.

4 Major functions

4.1 Library initialization

- Library initialization: Initializes the library and provides log level, initial random seed, and maximum thread count.
  (automatic through three initial parameters: loglv, rs, and nth, whose 0 indicates to use default values)
  l=findr.lib(path=None,loglv=6,rs=0,nth=0)
  findr.lib(loglv=6,rs=0,nth=0)

path	Extra location to search for Findr shared library in addition to default search paths. Only accepted in python interface.
loglv	Log level of library Findr. Only messages whose levels are not greater than the designated log level will be printed. Level 0: critical, 1-3: errors, 4-6: warnings, 7-9: information, 10-12: debug. Default value is level 6.
rs	Initial random seed. Default value means to use current time.
nth	Maximum number of parrallel threads in calculation. For best performance, this should not exceed the number of CPU cores. Default value indicates using the number of cores automatically detected, which is not always accurate.

4.2 Inference of pairwise regulations

- Inference of correlation $A \cdots B$ probability with expression data only, by converting log likelihood ratios to probabilities per A. Methodology can be found in [1].
  pij_rank_a ft ft2 nt nt2 ns fp nodiag
  ans=l.pij_rank_a(dt,dt2,nodiag=False)
  ans=findr.pij_rank_a(dt,dt2,nodiag=FALSE)

ft dt dt	Input matrix of expression levels of A, in file path, numpy.ndarray, or matrix format. Element [i,j] is the expression level of gene i of sample j. The matrix has dimension (nt,ns).
ft2 dt2 dt2	Input matrix of expression levels of B, in file path, numpy.ndarray, or matrix format. Element [i,j] is the expression level of gene i of sample j. The matrix has dimension (nt2,ns).
nt	Number of genes for A.
nt2	Number of genes for B.
ns	Number of samples.
nodiag	When A and B are the same, log likelihood ratio between alternative and null hypotheses gives infinity. To avoid its contamination in the conversion from log likelihood ratios into probabilities, users need to arrange data accordingly, when $\{A\}$ and $\{B\}$ are the same or when $\{A\}$ is a subset of $\{B\}$. The top submatrix of B's expression data must be identical with A, and nodiag must be set to 1, True, or TRUE. Otherwise, in the default configuration, $\{A\}$ and $\{B\}$ should not have any intersection and nodiag = 0, False, or FALSE.
fp ans['p'] ans$p	Output matrix of inferred probability of correlation $A \cdots B$ versus no correlation $A \quad B$, in file path, numpy.ndarray, or matrix format. Element [i,j] is the probability of gene i being correlated with gene j. The matrix has dimension (nt,nt2).

- Inference of regulation $E(A) \rightarrow A \rightarrow B$ probability with expression data of A, B and best eQTL data for A as $E(A)$. Conversion from log likelihood ratios to probabilities is performed for each A. Methodology can be found in [1].
  pijs_gassist_a fg ft ft2 nt nt2 ns fp1 fp2b fp2c fp3 na nodiag
  ans=l.pijs_gassist_a(dg,dt,dt2,na=None,nodiag=False)
  ans=findr.pijs_gassist_a(dg,dt,dt2,na=NULL,nodiag=FALSE)

  Inference of regulation $E(A) \rightarrow A \rightarrow B$ probability with expression data of A, B and best eQTL data for A as $E(A)$. Conversion from log likelihood ratios to probabilities is performed altogether. Methodology can be found in [1].
  pijs_gassist_tot fg ft ft2 nt nt2 ns fp1 fp2b fp2c fp3 na nodiag
  ans=l.pijs_gassist_tot(dg,dt,dt2,na=None,nodiag=False)
  ans=findr.pijs_gassist_tot(dg,dt,dt2,na=NULL,nodiag=FALSE)

fg dg dg	Input matrix of best eQTL genotype data $E(A)$, each row of which is the best eQTL of the corresponding row of ft,dt. Data is in file path, numpy.ndarray, or matrix format. Element [i,j] is the genotype value of the best eQTL of gene i of sample j, and should be among values $0, 1, \ldots, na$. The matrix has dimension (nt,ns).
ft dt dt	Input matrix of expression levels of A, in file path, numpy.ndarray, or matrix format. Element [i,j] is the expression level of gene i of sample j. The matrix has dimension (nt,ns).
ft2 dt2 dt2	Input matrix of expression levels of B, in file path, numpy.ndarray, or matrix format. Element [i,j] is the expression level of gene i of sample j. The matrix has dimension (nt2,ns).
nt	Number of genes for A.
nt2	Number of genes for B.
ns	Number of samples.
na	Number of alleles for the species considered. This constrains every genotype data to be among $0, 1, \ldots, na$. If unspecified (0, None, or NULL), na is automatically determined as the maximum value of fg,dg + 1.
nodiag	When A and B are the same, log likelihood ratio between alternative and null hypotheses can give infinity. To avoid its contamination in the conversion from log likelihood ratios into probabilities, users need to arrange data accordingly, when $\{A\}$ and $\{B\}$ are the same or when $\{A\}$ is a subset of $\{B\}$. The top submatrix of B's expression data must be identical with A, and nodiag must be set to 1, True, or TRUE. Otherwise, in the default configuration, $\{A\}$ and $\{B\}$ should not have any intersection and nodiag = 0, False, or FALSE.
fp1 ans['p1'] ans$p1	Output vector of inferred probability of test 1, $E(A) \rightarrow A$ (alternative) versus $E(A) \quad A$ (null), in file path, numpy.ndarray, or array format. Element [i] is the probability of best eQTL of gene i regulates gene i. The vector has dimension (nt). Since the library expects significant eQTLs only, all elements of this variable are set to one artificially.
fp2b ans['p2b'] ans$p2b	Output matrix of inferred probability of test 2 bold, $E(A) \rightarrow A \cdots\cdots B$ with $E(A) \rightarrow B$ (alternative) versus $E(A) \rightarrow A \quad B$ (null), in file path, numpy.ndarray, or matrix format. Element [i,j] is the probability of alternative hypothesis for A = gene i and B = gene j. The matrix has dimension (nt,nt2).
fp2c ans['p2c'] ans$p2c	Output matrix of inferred probability of test 2 conservative, $E(A) \rightarrow A \cdots\cdots B$ with $E(A) \rightarrow B$ (alternative) versus $E(A) \rightarrow A \leftarrow B$ (null), in file path, numpy.ndarray, or matrix format. Element [i,j] is the probability of alternative hypothesis for A = gene i and B = gene j. The matrix has dimension (nt,nt2).
fp3 ans['p3'] ans$p3	Output matrix of inferred probability of test 3, $E(A) \rightarrow A \rightarrow B$ (null) versus $E(A) \rightarrow A \cdots\cdots B$ with $E(A) \rightarrow B$ (alternative), in file path, numpy.ndarray, or matrix format. Element [i,j] is the probability of null hypothesis for A = gene i and B = gene j. The matrix has dimension (nt,nt2).

5 Examples

Part of GEUVADIS Consortium dataset is provided within the binary and python interfaces, as well as the R package. Usage of examples is provided in every distribution as below:

- **Binary interface**: see file EXAMPLES.
- **Python interface**: see module findr.examples.
- **R package**: see documentation of every function or Findr package.

6 Frequently asked questions

1. I have GCC on my Mac but when compiling Findr still asks me to download GCC.

   On Mac, Apple installs its own C compiler which tries to pretend to be GCC, although many functionalities of GCC are lacked in Apple's compiler. Findr needs some of these functionalities (such as OpenMP). You can download the source code of GCC from https://gcc.gnu.org. Because of the complications in building GCC, some Mac users prefer to download unofficial binary copies of GCC from third parties, such as Homebrew. The unofficial binary GCC may be installed under a name other than 'gcc'. Under such circumstances, consult the question 'How do I change C compiler name?'

2. How do I change C compiler name?

   For Findr library or binary interface, open Makefile with any text editor. Find the lines 'CC=gcc' and 'LD=gcc', and change gcc into your C compiler name. After that, run 'make distclean' before jumping back at the Build phase of installation in Section 2.1. For Findr R package, go to src/lib and perform the same operation. For python interface, no C code is included so no change is needed.

References

[1] Lingfei Wang and Tom Michoel. Improved microRNA target predictions from genetics of gene expression data with Findr (submitted). 2016.

[2] Lin S. Chen, Dipen P. Sangurdekar, and John D. Storey. *trigger: Transcriptional Regulatory Inference from Genetics of Gene ExpRession*, 2007. R package version 1.16.0.