M I D E R

USER'S GUIDE

Alejandro F. Villaverde (afvillaverde@iim.csic.es)

Julio R. Banga (julio@iim.csic.es)

With the collaboration of John Ross (Stanford) and Federico Morán (UCM)

January 17, 2014

Contents

1 Introduction

MIDER (Mutual Information Distance and Entropy Reduction) is a general purpose software tool for inferring network structures. It calculates distances among variables using an entropic measure based on mutual information, which takes into account time delays. For this purpose the user can choose between several definitions and normalizations of mutual information. After obtaining the distance map, conditional entropies calculated from joint entropies of multiple variables are used to distinguish between direct and indirect interactions and to assign directionality. A detailed description can be found in Villaverde et al. (2014); a diagram is shown in Fig. 1.

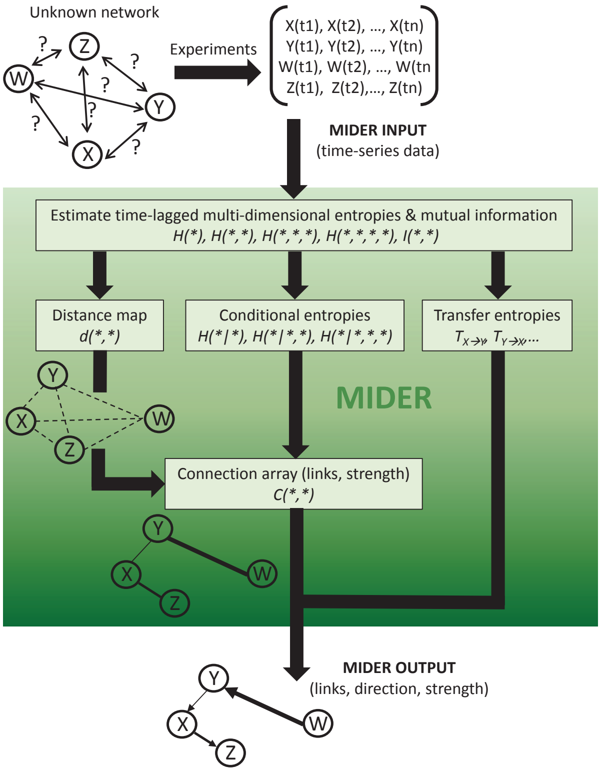


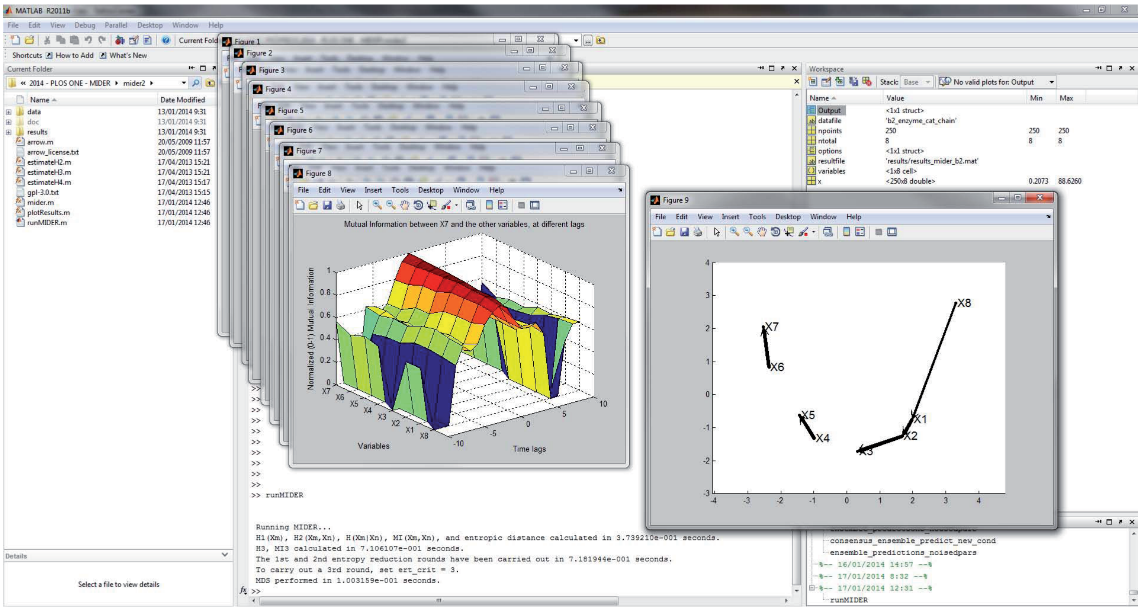
Figure 1: Workflow of the MIDER algorithm

The MIDER package is implemented in Matlab. The only requisite for its use is that some version of Matlab is installed in the system. MIDER is self-contained: it has no dependencies with external software. The MIDER code consists of a main script, `runMIDER.m`, and five auxiliary functions, `mider.m`, `plotResults.m`, `estimateH2.m`, `estimateH3.m`, and `estimateH4.m`. Additionally, a sixth function, `arrow.m`, which was developed by Dr. Erik A. Johnson (johnsone@usc.edu), is used for visualization purposes and is also distributed with the MIDER package. The functions `estimateH2.m`, `estimateH3.m`, and `estimateH4.m` perform adaptive estimation of mutual information and multi-dimensional joint entropies (of 2, 3, and 4 variables). They are based on the algorithm presented in Cellucci et al. (2005), which was kindly provided by Dr. Alfonso Albano (aalbano@brynmawr.edu).

2 Quick start: How to infer a network with MIDER

The MIDER package can be downloaded from `http://www.iim.csic.es/~gingproc/mider.html`. After downloading it, unzip it and save it in your computer. To use MIDER you only need to follow these three steps:

1. Open a Matlab session and go to the MIDER root directory (“mider”).
2. Define the problem and options by editing the script `runMIDER.m` (for details, see section 3). EXAMPLE (DEMO): If you are running MIDER for the first time and/or just want to see how it works, you can skip this step and leave `runMIDER.m` unedited. This will solve the benchmark problem B2 with the default options.
3. Run `runMIDER.m` (to do this you can either type “runMIDER” in the Matlab command window, or right-click runMIDER.m in the “Current Directory” tab and select “run”).


MATLAB R2011b
File Edit View Debug Parallel Desktop Window Help
Current Folder
Shortcuts How to Add What's New
Figure 1
Figure 2
Figure 3
Figure 4
Figure 5
Figure 6
Figure 7
Figure 8
Figure 9
File Edit View Insert Tools Desktop Window Help
Mutual Information between X7 and the other variables, at different lags
Normalized (0-1) Mutual Information
0
0.2
0.4
0.6
0.8
1
X7
X6
X5
X4
X3
X2
X1
X8
Variables
-10
-5
0
5
10
Time lags
X8
X7
X6
X5
X4
X1
X2
X3
4
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3
4
Workspace
Stack: Base
No valid plots for: Output
Details
Select a file to view details

« 2014 - PLOS ONE - MIDER ▸ mider2 ▸

Name	Date Modified
data	13/01/2014 9:31
doc	13/01/2014 9:31
results	13/01/2014 9:31
arrow.m	20/05/2009 11:57
arrow_license.txt	20/05/2009 11:57
estimateH2.m	17/04/2013 15:21
estimateH3.m	17/04/2013 15:21
estimateH4.m	17/04/2013 15:17
gpl-3.0.txt	17/04/2013 15:15
mider.m	17/01/2014 12:46
plotResults.m	17/01/2014 12:46
runMIDER.m	17/01/2014 12:46

Name	Value	Min	Max
Output	<1x1 struct>		
datafile	'b2_enzyme_cat_chain'		
npoints	250	250	250
ntotal	8	8	8
options	<1x1 struct>		
resultfile	'results/results_mider_b2.mat'		
variables	<1x8 cell>		
x	<250x8 double>	0.2073	88.6260

```
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
>> runMIDER

Running MIDER...
H1(Xm), H2(Xm,Xn), H(Xm|Xn), MI(Xm,Xn), and entropic distance calculated in 3.739210e-001 seconds.
H3, MI3 calculated in 7.106107e-001 seconds.
The 1st and 2nd entropy reduction rounds have been carried out in 7.181944e-001 seconds.
To carry out a 3rd round, set ert_crit = 3.
MDS performed in 1.003159e-001 seconds.
>>
```

```
consensus_ensemble_predict_new_cond
ensemble_predictions_noisedpars
%-- 16/01/2014 14:57 --%
%-- 17/01/2014 8:32 --%
%-- 17/01/2014 12:31 --%
runMIDER
```

Figure 2: Screenshot of an execution of MIDER

Done! Results should be obtained in a few seconds. A screenshot is shown in Figure 1. MIDER outputs two types of figures: (1) a 2D map of the distances among variables and the predicted links ("Figure 9" in the screenshot), and (2) for every variable, a plot of the mutual information between that variable and the rest, for all the time lags considered ("Figure 8" in the screenshot). Additionally, the results of the calculations are stored in the workspace and saved in a MAT-file. MIDER outputs are described in more detail in section 3.2. Further details about the use of MIDER are given in the next section, starting with the problem definition in 3.1.

3 MIDER usage

3.1 Problem definition

The script `runMIDER.m` is the main file, and the only one that the user has to modify. It defines a Matlab structure called **options** containing the following fields:

- **options.q**: value of the entropic parameter. Choose $q = 1$ for the classic Shannon entropy (also known as Boltzmann-Gibbs), or $q > 1$ for the generalized Tsallis entropy. If you want to try Tsallis entropy, typical values are $1.5 < q < 3.5$.
  Default: options.q = 1 (Shannon entropy).

- **options.MItype**: selects the type of normalization of mutual information used to create the distance map. Choose 'MI' for the classic, not normalized value; 'MImichaels' for the normalization presented in Michaels et al. (1998); 'MIlinfoot' for the one in Linfoot (1957); or 'MIstudholme' for the one in Studholme et al. (1999). Note that MIDER always calculates (and outputs) all the normalizations; this option is to select the one used in the distance map.
  Default: options.MItype = 'MI'.

- **options.fraction**: this parameter is involved in the adaptive estimation of mutual information of a pair of variables (X,Y). It is the minimum fraction of occupied bins in the (X,Y) space with at least 5 points. It should be between 0.01 and 0.5.
  Default: options.fraction = 0.1*(log10(npoints)-1), where npoints = number of data points.

- **options.taumax**: the maximum time lag between two variables X and Y considered in the calculation of mutual information.
  Default: options.taumax = 10.
- **options.ert_crit**: number of entropy reduction rounds to carry out (0, 1, 2, or 3).
  Default: options.ert_crit = 2.
- **options.threshold**: entropy reduction threshold. Enter a number (typically between 0.0 and 0.2) to fix it manually, or choose 'adapt' to use a value obtained from the data.
  Default: options.threshold = 'adapt'.
- **options.plotMI**: plot mutual information arrays (=1) or not (=0).
  Default: options.plotMI = 1.

Additionally, a MAT-file containing the input data must exist in the "data" folder. The MIDER distribution includes 7 datafiles, one for each of the benchmark problems studied in the publication Villaverde et al. (2014). The input MAT-file is specified in the first lines of the `runMIDER.m` script; by default the B2 benchmark data is chosen (`datafile = 'b2_enzyme_cat_chain';`). The MAT-file must contain at least two variables:

- A m*n data array named 'x', with *m* data points (rows) and *n* variables (columns).
- A vector named 'variables', containing the names of the variables as strings.

Finally, it is recommended to save the results in another MAT-file. To do this, specify a name of the results file (default: `resultfile = 'results/results_mider_b2.mat';`).

3.2 Output

MIDER produces a structure called **Output** containing the following fields:

- **Output.H1** = n-vector of entropies.
- **Output.MI** = n*n*(nlags+1) array, mutual information (several lags).
- **Output.MIl** = mutual information normalized as in Linfoot (1957).
- **Output.MIm** = mutual information normalized as in Michaels et al. (1998).
- **Output.MIs** = mutual information normalized as in Studholme et al. (1999).
- **Output.H2** = n*n*(nlags+1) array, joint entropy of 2 variables.
- **Output.H3** = n*n*n array of joint entropy of 3 variables.
- **Output.H4** = n*n*n*n array of joint entropy of 4 variables.
- **Output.MI3** = n*n*n array of three-way mutual information.
- **Output.dist** = n*n*(nlags+1) array of distance between variables.
- **Output.taumin** = n*n array of the time lags that minimize the entropic distance.
- **Output.cond_entr2** = n*n array of conditional entropies of 2 variables.
- **Output.cond_entr3** = n*n*n array of conditional entropies of 3 variables.
- **Output.cond_entr4** = n*n*n*n array of conditional entropies of 4 variables.
- **Output.con_array** = n*n array of connections between variables

- **Output.adaptThres** = adaptive threshold value
- **Output.T** = n*n array of transfer entropies
- **Output.Y** = coordinates of the points from multidimensional scaling

References

Cellucci, C., Albano, A., and Rapp, P. (2005). Statistical validation of mutual information calculations: Comparison of alternative numerical algorithms. *Phys. Rev. E*, 71(6):066208.

Linfoot, E. (1957). An informational measure of correlation. *Inf. Control*, 1:85–89.

Michaels, G., Carr, D., Askenazi, M., Fuhrman, S., Wen, X., and Somogyi, R. (1998). Cluster analysis and data visualization of large scale gene expression data. In *Pac. Symp. Biocomp.*, volume 3, pages 42–53.

Studholme, C., Hill, D., and Hawkes, D. (1999). An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recogn.*, 32:71–86.

Villaverde, A., Ross, J., Morán, F., and Banga, J. (2014). Mider: network inference with mutual information distance and entropy reduction. *PLOS ONE*, submitted.