The mode-of-action by network identification (MNI) algorithm: a network biology approach for molecular target identification

Heming Xing[1,2] & Timothy S Gardner[1]

[1]Department of Biomedical Engineering, Boston University, 44 Cummington Street, Boston, Massachusetts 02215, USA. [2]Cellicon Biotechnologies, Inc., 222 Berkeley Street, Suite 1040, Boston, Massachusetts, 02116, USA. Correspondence should be addressed to T.S.G. (tgardner@bu.edu).

Published online 29 December 2006; doi:10.1038/nprot.2006.300

This protocol details the use of the mode-of-action by network identification (MNI) algorithm to identify the gene targets of a drug treatment based on gene-expression data. Investigators might also use the MNI algorithm to identify the gene mediators of a disease or the physiological state of cells and tissues. The MNI algorithm uses a training data set of hundreds of expression profiles to construct a statistical model of gene-regulatory networks in a cell or tissue. The model describes combinatorial influences of genes on one another. The algorithm then uses the model to filter the expression profile of a particular experimental treatment and thereby distinguish the molecular targets or mediators of the treatment response from hundreds of additional genes that also exhibit expression changes. It takes ~1 h per run, although run time is significantly affected by the size of the genome and data set.

INTRODUCTION

DNA microarray technology has been extensively used to measure the concentration of all mRNAs within an organism. For example, genes responding to drug treatments or disease conditions can be identified based on gene-expression profiles. However, the gene-expression profile alone does not distinguish initial targets or mediators of a treatment response from the many genes that respond secondarily to the initial effects. Different network-inference approaches have been proposed to help address this problem[1]. We have developed a model-based approach, called mode-of-action by network identification (MNI)[2], for target identification.

The MNI algorithm requires three sets of data: a compendium data set consisting of hundreds of expression profiles in the organism of interest, a calibration data set consisting of at least five expression profiles for treatments with known mode of action (MOA), and a test data set of experimental conditions or treatments of interest. Gene-expression data from either single-channel oligonucleotide microarrays (such as the Affymetrix GeneChip) or two-channel spotted microarrays (such as custom cDNA microarrays) can be used for the data sets.

The algorithm operates in two phases (**Fig. 1**). In phase one, the algorithm applies a machine-learning approach based on multiple regression to the training data to construct a statistical model of regulatory influences of genes on one another. In phase two, the algorithm filters the test expression profiles to distinguish the molecular targets or mediators of the treatment response from hundreds of additional genes that also exhibit expression changes. In this step, the algorithm looks for genes with expression changes that are not explained by the regulatory model. The algorithm assumes that such 'outlier' genes are influenced externally by the drug treatment. Alternately, if the experimental profile represents a disease state, then the outlier genes are probably influenced by a mutation in that gene or its regulatory region. The significance of this external effect is quantified with a *Z*-score. The high-score genes are identified as the putative MOA of the experimental condition.

Differential expression analysis, which identifies genes based on the significance of expression change, uses no knowledge of regulatory influences. Therefore, it does not typically distinguish initial mediators from genes responding to signal propagation through the regulatory network. By contrast, the MNI approach performs well at identifying treatment MOA because it makes use of information on the gene-regulatory network underlying expression changes.

The MNI approach requires no prior knowledge of gene-network structure to identify treatment targets. Thus, it is applicable to organisms with limited previous pathway or regulatory network data. The MNI algorithm has been applied successfully to identify drug targets and disease mediators based on gene-expression data from yeast[2], humans (A. Ergun and J.J. Collins, unpublished data), bacteria and other organisms (X.H., unpublished data). In these applications, MNI identifies both well-annotated and unannotated genes as targets of experimental conditions. The ability to identify novel genes as targets of drug treatment or as mediators of disease condition also makes the MNI algorithm a potential tool for gene-function annotation.

MATERIALS

EQUIPMENT

- Hardware: PC running Windows 2000, Windows XP, Mac OS X or Linux. A 1-GHz processor or faster, ≥2 GB RAM and ≥1 GB available disk space ▲ CRITICAL Large data sets and genomes might require significantly more RAM and storage for optimum performance
- Software: the MNI algorithm is available to academic users for free download (http://gardnerlab.bu.edu). The software is available as both an executable and a Matlab script ▲ CRITICAL A Matlab library, distributed with the MNI software, must be installed in order to run the executable; Matlab 7.0 is recommended to run the script version
- Files: raw gene-expression data files in Affymetrix CEL format or gene-expression data files as comma-separated or tab-delimited expression values; an example data set is available on the MNI download page (http://gardnerlab.bu.edu)

PROCEDURE

Collection and preparation of the training data set

1| Assemble a training data set. These data include the test experiments (for which you want to determine the MOA), the calibration experiments (for which you know the MOA) and the compendium experiments (training-only experiments for which you will not determine the MOA). The algorithm can use all groups of experiments during the training phase to build the regulatory model. Test experiments are selected out for analysis during the algorithm runtime. Data can be from either in-house or public domain. A good resource for public microarray data is the National Institutes of Health Gene Expression Omnibus (http://www.ncbi.nlm.nih.gov/geo/).
▲ CRITICAL STEP The training set might include multiple types of data, such as drug treatments, RNA interference (RNAi) treatments, mutant profiles or physiological conditions. We recommend that the training data set consists primarily of steady-state samples of cell responses, because the algorithm does not explicitly account for dynamics. Nevertheless, we have found that inclusion of time-series data can improve results. We also recommend the inclusion of baseline data (e.g., wild-type strains or unperturbed conditions). The most important characteristic in a training data set is high diversity (low correlation) between expression profiles of the various conditions. The 'rule of thumb' for the minimum number of training experiments is $M/10$, where M is the number of genes on the microarray. Include $\geq$200 experiments in the training data set. More data are generally helpful although less data might still work. Investigators can determine the adequacy of the data during the optimization process (Steps 10 and 11).

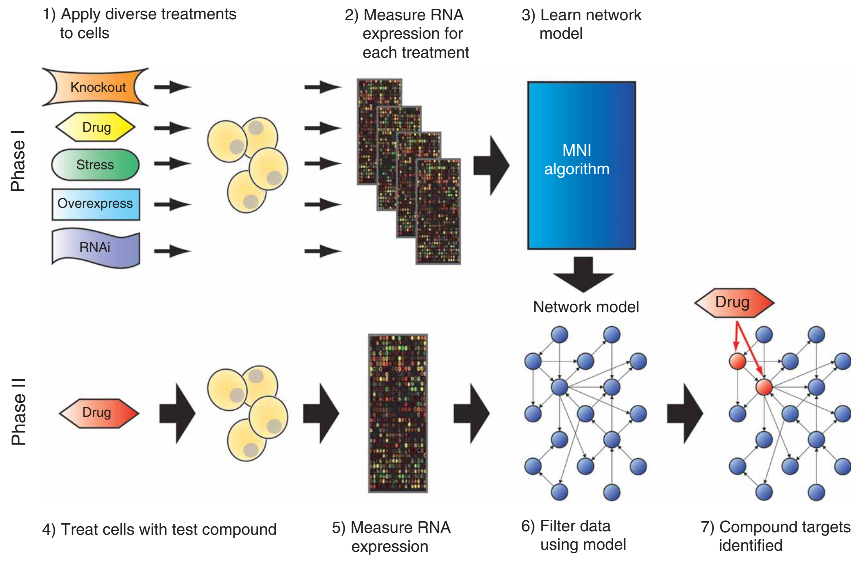
1) Apply diverse treatments to cells
2) Measure RNA expression for each treatment
3) Learn network model
Phase I
Knockout
Drug
Stress
Overexpress
RNAi
MNI algorithm
Network model
Drug
Phase II
Drug
4) Treat cells with test compound
5) Measure RNA expression
6) Filter data using model
7) Compound targets identified

Figure 1 | Schematic overview of the MNI method. In phase 1, a set of treatments, including knockouts, compounds, overexpressions and/or RNAi, is applied to an organism. Microarray data from the samples are processed by the MNI algorithm to infer a model of the regulatory influences between genes in the organism (blue-filled circles indicate genes and arrows indicate regulatory influences). In phase 2, microarray data from the test treatment are filtered using the reconstructed network model to distinguish the molecular targets of the test treatment (red-filled circles) from secondary responders. Reproduced from *Nature Biotechnology*.

2| Normalize the gene-expression data using appropriate methods. We recommend using robust multichip analysis (RMA)[3] for normalizing data from Affymetrix oligonucleotide microarrays and the LOWESS algorithm[4] for normalizing data from two-color spotted microarrays. The algorithm can process expression data as ratios (for two-color array data), or as expression intensities (for Affymetrix GeneChip data). The use of ratios or raw intensities is governed by the capabilities of the microarray technology, not by the MNI algorithm.
▲ CRITICAL STEP We recommend that training data sets include only profiles from a single experimental platform. Mixing of data from various platforms could lead to poor results due to quantitative biases among the technologies.

3| Apply log transformation to the gene-expression data. Either log2 or log10 transformation can be applied.
▲ CRITICAL STEP Note that the output of the RMA normalization program is already log2 transformed and this step should be skipped.

4| Compute the average and standard deviation for each gene based on experimental replicates. For experiments lacking replicates and associated standard deviations, compute the standard deviation of each gene across all experimental profiles in the data set. Alternatively, if some data include replicates and others do not, a K-nearest neighbor approach can be applied to compute the variance (using the expression value as the predictor of variance). We recommend the K-nearest neighbor approach only if a small fraction of data points lacks variance information.

Formatting the data

5| Save the expression values, standard deviations, probe identifiers and experimental names as separate text files. The expression data file contains one row for each probe on the array, and one column for each experiment. All groups of experiments (compendium, calibration and test) should be included in one file. The entries are either the log expression intensities or the log ratios computed in Step 4. The standard deviation file will have the same format as the expression data file, but the entries are the corresponding standard deviation values. The probe identifier file contains a list of gene names or

other probe identifiers, with each name on a new line. The experimental name file contains a list of experiment names, with each name on a new line.
▲ CRITICAL STEP Name the expression data file as "logR.txt," the standard deviation file as "slogR.txt," the probe identifier file as "gid.txt" and the experimental name file as "expts.txt" in the current directory. The MNI loader accepts these files as input files.

TABLE 1 | Example of an MNI parameter file.

#Perturbation	*thP*	*Kfrac*	*NROUNDS*	*Q*	*TopN*
⟨*Experiment Name1*⟩	0.25	0.333333	3	117	100
⟨*Experiment Name2*⟩	0.25	0.333333	3	117	100

This is an example of configuring an MNI program to run two times. Values are separated by tabs and any row starting with "#" is ignored.

6| To invoke the MNI loader script, type in "⟨*path to MNI*⟩/MNI_loader" at the Matlab command window. Alternatively, at the operating system command line, type "⟨*path to MNI*⟩\MNI_loader.exe" to invoke the executable version of the loader. The MNI loader script will create a file named "data.mat" in the current directory, which is the input of the MNI program.
▲ CRITICAL STEP Using double precision on numerical data will improve the performance of the MNI algorithm.

Running the MNI program with default parameters

7| Create a parameter file named "parameter.txt" in the current directory. Use this file to set up parameters for the execution of the MNI program. Within the parameter file, each row is composed of tab-delimited values that define all six parameters for each execution of the program: the experiment to be analyzed (*Perturbation*), the threshold for significant external influence (*thP*), the fraction of genes to be kept (*Kfrac*), the number of rounds (*NROUNDS*), the number of singular values (*Q*) and the number of genes to be ranked in the MNI output (*TopN*). See Steps 10 and 11 for more details on each parameter. Append additional rows to run the MNI program multiple times, analyzing a different experiment or using different parameters each time. **Table 1** shows an example of the parameter file.

8| Run the MNI program by typing "⟨*path to MNI*⟩\MNI.exe" at the operating system command line. Otherwise, run MNI from the Matlab command window by typing in "⟨*path to MNI*⟩/MNI". MNI outputs a list of the top ranked genes into a text file, which has the same name as the experiment tested (**Table 2**). The file resides in the current working directory.
▲ CRITICAL STEP MNI removes the expression data of the experiment to be analyzed during the training phase and builds a new network model dynamically in each run. This step is performed to avoid biasing the network model by the analyzed experiment.

Optimizing MNI parameters

9| Because MNI parameters are data-dependent, investigators must tune MNI for each new training data set using the calibration experiments. The calibration data set should include at least five experiments with known MOA and should represent diverse conditions (i.e., the data should exhibit low correlations with each other). This is important to avoid over-training of the model to a specific class of profiles. The data set might include any type of experiment, such as drug treatments, RNAi treatments, mutant profiles or physiological conditions. Run the MNI program with each calibration experiment and choose a set of parameters that allows MNI to identify gene targets for the calibration experiments with the least overall error. To do this, we recommend the use of a cross-validation procedure. Initially, select 80% of the calibration experiments (at random) and use MNI to predict the targets. Then compute the rank error between the predicted and known targets for the last 20% of the experiments. Repeat this procedure five times, each time randomly selecting a different set of 80% of the calibration experiments. Choose the parameter set with the least average error. These are the optimum parameters for the training data set.
▲ CRITICAL STEP When computing the rank error, if the known target appears below a rank of 100, then set its rank to 100. Investigators might achieve further enrichment of the predicted targets using pathway analysis, as described in Step 13.

10| Set the first four parameters. Choose the *NROUNDS*, *Kfrac* during each tournament, *thP* and *TopN*. A tournament approach is used to improve the specificity of the algorithm. The top *Kfrac* is selected and the algorithm is applied to the selected genes for ranking until *NROUNDS* is reached. The selection of above parameters should ensure that >200 genes are kept in the last round of tournament. The *thP* is used to determine the significant external influences and decreased threshold leads to more false-positive predictions.

TABLE 2 | Example of an output file.

	CHOSE MODIFIED *Z*-SCORE	
	100 predicted mediators for itraconazole	
	NORMAL *Z*-SCORE	**MODIF *Z*-SCORE**
Rank	**Gene ID**	**Gene ID**
1	YDR367W	YDR367W
2	YHR007C	YHR007C
3	YNL280C	YPL272C
4	YPL272C	YIL117C
5	YDR454C	YER011W
6	YGR060W	YBR089W
7	YGL186C	YGR151C
8	YGR175C	YMR015C
9	YJL113W	YJR149W
10	YDR487C	YJR150C

This is an example of MNI output files, only the 10 highest ranked genes from normal and modified *Z*-scores are shown.

▲ CRITICAL STEP We recommend using the following default values: 1/3 for *Kfrac*, 3 for *NROUNDS*, 0.25 for *thP* and 100 for *TopN*. Algorithm tuning is primarily achieved by setting the complexity parameter *Q*.

11| Set the model complexity parameter *Q*, which represents the number of singular values retained during the data-compression step. Choosing a *Q* that is too low will result in high prediction error. Choosing a *Q* that is too high will result in overtraining and poor generalization performance. Calculate the initial *Q* value as *n*/5, where *n* is the number of experiments in the training data set. Increase or decrease the *Q* value and evaluate the performance of MNI via the cross-validation procedure of Step 9.
▲ CRITICAL STEP The *Q* value is the most important parameter governing MNI performance.

Prediction of molecular targets by MNI
12| Create a parameter file using a text program. Use the optimum parameter set chosen from the calibration process and specify the test experiments to analyze. Run the MNI program. The top-ranked genes in each output file are the putative molecular targets of the specified experiment. The higher the rank, the more likely the gene is to be the target.

Enrichment of MNI results (optional)
13| We recommend that investigators enrich the MNI results by performing pathway analysis on the ranked gene list. Using structured gene annotations, such as Gene Ontology (GO) terms[5], calculate the associated *P* value for pathway enrichment by Fisher's exact test or hypergeometric probability[6].
▲ CRITICAL STEP Investigators might use pathway data from the GO database, the Kyoto Encyclopedia of Genes and Genomes (KEGG) database or any other database of pathway-structured gene annotations.

● TIMING
Depending on the number of genes on the microarray, the number of experiments in the training data set and the computer hardware, it typically takes from 10 min to 1 h to finish each run of the MNI program. Longer run times are possible.

? TROUBLESHOOTING
The MNI algorithm creates a log file that provides information on system errors and intermediate results. Additional error messages are displayed in the command window. Refer to **Table 3** for troubleshooting advice.

TABLE 3 | Troubleshooting table.

Problem	Possible reason	Solution
Error message: out of memory	MNI algorithm needs more memory to process the data set	Free up more memory by quitting other applications or by restarting computer; if problem still occurs, add more RAM memory to computer or switch to a 64-bit platform if >4 GB RAM is needed
Error message: bad conditioning using last estimate of A	The training data are not diverse enough	Add more data to the training data set
Control probes on the microarray are listed as top ranked genes	Data from control probes are included in the training data	Remove the data corresponding to control probes

ANTICIPATED RESULTS
The output file contains the putative mediators of experimental condition tested as shown in **Table 2**. The first row indicates whether the investigator should choose the normal *Z*-score or modified *Z*-score for ranking. The second row describes the experimental condition. The top-ranked genes from both the normal *Z*-score and modified *Z*-score are listed after the second row. Pathway-enrichment analysis is optional and is not included in the MNI output. Investigators might apply such analyses separately.

COMPETING INTERESTS STATEMENTS The authors declare competing financial interests (see the HTML version of this article for details).

Published online at http://www.natureprotocols.com
Reprints and permissions information is available online at http://npg.nature.com/reprintsandpermissions

1. Ambesi-Impiombato, A. & di Bernado, D. Computational biology and drug discovery: from single-target to network-target. *Curr. Bioinform.* **1**, 3–13 (2006).
2. di Bernardo, D. *et al.* Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. *Nat. Biotech.* **23**, 377–383 (2005).
3. Bolstad, B.M., Irizarry, R.A., Astrand, M. & Speed, T.P. A comparison of normalization methods for high density oligonucleotide array data based on bias and variance. *Bioinformatics* **19**, 185–193 (2003).
4. Yang, Y.H. *et al.* Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.* **30**, e15 (2002).
5. Ashburner, M. *et al.* Gene Ontology: tool for the unification of biology. *Nat. Genet.* **25**, 25–29 (2000).
6. Curtis, R.K., Oresic, M. & Vidal-Puig, A. Pathways to the analysis of microarray data. *Trends Biotech.* **23**, 429–435 (2005).