DREAM4: Combining Genetic and Dynamic Information to Identify Biological Networks and Dynamical Models

Alex Greenfield[1]◔, Aviv Madar[2]◔, Harry Ostrer[3], Richard Bonneau[1,2,4]*

1 Computational Biology Program, New York University Sackler School of Medicine, New York, New York, United States of America, **2** Department of Biology, Center for Genomics and Systems Biology, New York University, New York, New York, United States of America, **3** Human Genetics Program, Department of Pediatrics, New York University Langone Medical Center, New York, New York, United States of America, **4** Computer Science Department, Courant Institute of Mathematical Sciences, New York University, New York, New York, United States of America

Abstract

Background: Current technologies have lead to the availability of multiple genomic data types in sufficient quantity and quality to serve as a basis for automatic global network inference. Accordingly, there are currently a large variety of network inference methods that learn regulatory networks to varying degrees of detail. These methods have different strengths and weaknesses and thus can be complementary. However, combining different methods in a mutually reinforcing manner remains a challenge.

Methodology: We investigate how three scalable methods can be combined into a useful network inference pipeline. The first is a novel t-test–based method that relies on a comprehensive steady-state knock-out dataset to rank regulatory interactions. The remaining two are previously published mutual information and ordinary differential equation based methods (tlCLR and Inferelator 1.0, respectively) that use both time-series and steady-state data to rank regulatory interactions; the latter has the added advantage of also inferring dynamic models of gene regulation which can be used to predict the system's response to new perturbations.

Conclusion/Significance: Our t-test based method proved powerful at ranking regulatory interactions, tying for first out of 19 methods in the DREAM4 100-gene *in-silico* network inference challenge. We demonstrate complementarity between this method and the two methods that take advantage of time-series data by combining the three into a pipeline whose ability to rank regulatory interactions is markedly improved compared to either method alone. Moreover, the pipeline is able to accurately predict the response of the system to new conditions (in this case new double knock-out genetic perturbations). Our evaluation of the performance of multiple methods for network inference suggests avenues for future methods development and provides simple considerations for genomic experimental design. Our code is publicly available at http://err.bio.nyu.edu/inferelator/.

Citation: Greenfield A, Madar A, Ostrer H, Bonneau R (2010) DREAM4: Combining Genetic and Dynamic Information to Identify Biological Networks and Dynamical Models. PLoS ONE 5(10): e13397. doi:10.1371/journal.pone.0013397

Editor: Mark Isalan, Center for Genomic Regulation, Spain

Received April 2, 2010; **Accepted** September 9, 2010; **Published** October 25, 2010

Funding: The authors were funded by National Science Foundation (NSF) grant DBI-0820757, National Institutes of Health (NIH) PSOC # I U54CA143907-01, the University of Luxembourg/ Luxembourg Centre for Systems Biomedicine, and the National Institutes of Health through the NIH Roadmap for Medical Research (PN2 EY016586). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: Bonneau@nyu.edu

◔ These authors contributed equally to this work.

Introduction

Predicting how a cell will respond, at the molecular level, to environmental and genetic perturbations is a key problem in systems biology. Molecular regulatory systems-level responses are governed by several regulatory mechanisms including the underlying transcriptional regulatory network (RN). Recently, there has been an increase in the number of genome-wide datasets appropriate for large scale network inference, which has driven a large interest in methods for learning regulatory networks from these datasets. In general, the question of inferring a transcriptional RN can be posed in the following way: given a set of regulators (transcription factors - TFs) and a set of targets (genes), what are the regulatory relationships between the elements in these two sets? These relationships can be directed (e.g. gene A regulates gene B) or undirected (e.g. there is a regulatory relationship between gene A and gene B), and can have parameters describing the strength, confidence and/or kinetics of the regulatory interaction (depending on the method used). RN inference techniques use three main types of genome-wide data: 1) steady-state transcriptional profiling of the response to perturbations (e.g. gene knock-out or exposure to a drug,), 2) collections of time series observations following relevant perturbations, and 3) measurements of TF-DNA binding. Different types of RN inference methods produce RNs that vary in detail and comprehension. One critical distinction is the scalability of any given method. Typically, methods that learn less detailed regulatory models scale to larger systems and data sizes than methods that learn more complex models. Another critical difference between methods is whether causal (directed) edges or undirected relationships are learned. Several current methods aim to learn dynamical parameters, such as TF-target activation rates and rates of degradation of gene products. Ideally, a computational biologist should choose the most detailed method that the data will support, as more detailed models can suggest more focused biological hypothesis and be used to model a system's behavior in ways that simple network models cannot. Given this constant need to balance the specific features of any given biological dataset with the capabilities of multiple RN inference algorithms, testing of RN inference methods using a variety of datasets is a critical field-wide activity. Several recent methods aim to do so by generating biologically meaningful datasets with a known underlying topology [1–4].

To this end, the Dialogue for Reverse Engineering Assessments and Methods (DREAM) [5] provides a set of networks which can be used to develop and test RN inference methods. The networks presented by DREAM make some simplifications of the networks found in a cell, and the corresponding datasets are ideal in their completeness. The control of cellular processes occurs on at least four distinct levels including DNA, transcript, protein, and metabolite. Measuring only transcript levels ignores the fact that cellular interactions happen on the level of proteins, and are mediated in many cases by metabolites. Accordingly, an ideal dataset for RN inference would contain time-series measurements of multiple levels of regulation (RNA, protein, protein-modifications, etc.) with the sampling rate on the order of the fastest reaction. Additionally, the cellular response to genetic perturbation (e.g. gene knock-out) would also be available. Although advances are currently being made in the cost and accuracy of genome-wide proteomics, metabolomics, and protein binding (ChIP-chip, ChIP-seq) [6,7] measurements, the most mature and cost efficient technologies remain those that measure genome-wide transcription-level responses. Experimental and financial constraints typically prohibit obtaining these measurements in a finely time-resolved manner. The DREAM challenge removes many of these constraints and presents participants with an idealized expression dataset for which the true topology (gold-standard) is known. This presents a unique opportunity to develop RN inference methods and immediately test their performance by comparison with the gold-standard.

It should be noted that biological systems present several advantages not relevant to the DREAM4 challenge. These advantages (not discussed here) are leveraged by integrative methods for learning modularity prior to inference [8–12], methods that use structured priors derived from compilations of validated biological regulatory interactions [13–16], and approaches to characterize binding sites [17,18]. A thorough review of current network inference methods is beyond the scope of this introduction but can be found in [19–24]. Here we briefly review only the classes of methods that we utilized in our hybrid approach: mutual information (MI) based methods, ordinary differential equation (ODE) based methods, and resampling methods.

Several methods for detecting significant regulatory associations are based on similarity metrics derived from information theory, such as MI. [25]. The MI between two signals (in this case the expression of a TF and its target) is calculated by subtracting the joint entropy of each signal from the sum of their entropies. It is similar to correlation (the higher the magnitude, the stronger the relationship), but is more generally applicable as it does not assume a linear relationship between the two signals, nor does it assume continuity. At their core, methods that rely on MI generally infer undirected interactions, as the MI between two variables is a symmetric quantity [26–29], however modifications can be made that allow for the inference of direction [30,31]. Here, we use an MI-based method, time-lagged Context Likelihood of Relatedness (tlCLR) [31], which is based on Context Likelihood of Relatedness (CLR) [29], to learn initial topology that is further optimized and parametrized by Inferelator 1.0 [32]. tlCLR extends CLR by making use of the temporal information contained in time series observations to estimate the directionality of a significant regulatory interaction. This method is described in [31] and is reviewed in the methods section. tlCLR cannot be used to predict the response of the system to previously unseen perturbations as it does not infer any dynamical parameters. A different approach is needed to calculate these dynamical parameters. In the context of our full RN inference pipeline, which includes fitting of dynamical parameters, tlCLR is used as a feature selection algorithm that identifies a set of likely regulators for each target based on time-lagged, corrected MI.

Ordinary differential equation based methods for RN inference attempt to learn not only the topology of the network (i.e. "who regulates who"), but also the dynamical parameters associated with each regulatory interaction. Regulatory network models resulting from these methods can be used to predict the system-wide response to previously unseen conditions, future time-points, and the effects of removing system components. A drawback of these methods is that they generally require time-series data and more complete datasets than many alternative methods. ODE methods model the rate of change in the expression of a gene as a function of TFs (and other relevant effects) in the system. ODE based methods differ in their underlying functional forms, how the ODE system of equations is solved (coupled or uncoupled solution), and how prior knowledge and sparsity constraints are imposed on the overall inference procedure. For example, several methods have been proposed that use complex functional forms [33], and solve a coupled system [33,34], while other methods [32,35–38] solve a simplified linear system of ODEs. The Inferelator 1.0 [32], is an RN inference method which learns the network as a system of linear ODEs, where the rate of change for each gene is modeled as a function of the known regulators in the system. Inferelator 1.0 uses a finite difference approximation to estimate the change in the response over a given time interval, and uses an efficient implementation of l_1-constrained linear regression, LARS [39], to enforce model sparsity. The Inferelator 1.0 has previously been used to learn a large portion of the *Halobacterium salinarium* transcriptional regulatory network, and was able to predict mRNA levels of 85% of the genes in the genome over new experimental conditions [40]. Additionally, feature selection by tlCLR followed by optimization and parameterization via Inferelator 1.0 was a top performing method for the DREAM3 network challenge [31]. One drawback of the original formulation of these scalable MI and ODE based methods is that they rely on point estimates for many network parameters and thus are not ideal for estimating the error in the inferred parameters [41]. One possible solution is to use a resampling approach [42,43] to generate an ensemble of predicted networks from which the confidence interval for any parameter can be estimated.

Resampling refers to a broad class of statistical methods that are often used to assess confidence bounds on sample statistics by empirically generating distributions [42]. Recently, several groups have used resampling approaches in a biological context. In this setting resampling methods are an attractive means of determining confidence bounds on model parameters (such as the strength and directionality of a putative regulatory interaction) for two main reasons: 1) resampling methods are non parametric and thus applicable in cases where complex or ill-understood regulatory relationships might confound assumptions about the correct error distribution, and 2) resampling methods do not, in our case, decrease algorithm scalability. Resampling methods have been applied in several contexts to estimate error in a variety of genomics data-analysis contexts. Kerr et al. [44] used a resampling approach to assess confidence bounds of clusters from ANOVA models. Resampling of a gaussian process regression model was used by Kirk et al. [45] to show the sensitivity of the inferred network to uncertainty in the underlying data. Friedman et al. [46] used a resampling approach of a Bayesian network reconstruction algorithm to assess the confidence of inferred parameters. Additionally, Marbach et al. [47] showed that a resampling approach applied to a genetic algorithm for network inference was a top performering method in the DREAM2 five-gene network challenge. We show that by using a resampling approach to generate ensembles of networks with our network inference pipeline we can improve the accuracy of our topology predictions.

Here we focus on which data types (time-series or steady-state), and which methods (ODE-based, MI-based, genetic perturbation based, or combinations thereof) can be expected to perform best at either reconstructing network topology or predicting the response of the system to new perturbations. Our analysis suggests several simple considerations for determining the correct balance between time-series and steady-state data required for large-scale network inference, and how to use these distinct data types in a mutually reinforcing manner.

Methods

The DREAM4 datasets consisted of both time-series and steady-state data, and participants were challenged to predict: 1) the topology of the network (as a ranked list of regulatory interactions), and 2) the response of the network to combinations of genetic perturbations (double knock-outs). We have participated in both challenges. For challenge 1 we used a relatively simple t-test based method, Median Corrected Z-scores (MCZ, pipeline 1, Figure 1), which tied for 1*st* place at predicting the topology of the network. For challenge 2 we used a network inference pipeline (pipeline 3, Figure 1) that combined MCZ with our previously published top-performing method for DREAM3 [31] (tlCLR-Inferelator 1.0, pipeline 2, Figure 1), placing 2*nd* at predicting the response of the network to double knock-outs. Pipeline 3 represents our initial attempt at combining pipeline 1 and pipeline 2 in a mutually reinforcing manner. Although pipeline 3 was complementary to MCZ in that it allowed us to predict the response of the system to double knock-outs, it was not complementary at predicting the topology of the network, placing 8*th* (out of a total of 19 predictions).

After the results for DREAM4 were in, we re-evaluated our methods with the goal of identifying where improvements can be made. We aimed to find an alternate way to combine pipeline 1 and 2 in a mutually reinforcing manner with respect to topology predictions. We show that by combining pipelines 1 and 2 in a resampling approach (pipeline 4, Figure 1), we were able to generate topology predictions that outperformed those of either pipeline. Pipeline 4 also retains the ability to predict the response of the system to double knock-outs. Additionally, we were able to improve upon the ability of pipeline 3 to predict the data (the response of the network to double knock-outs) by reconsidering how we construct the initial conditions. Originally the initial conditions were set to the w.t. expression levels for all genes. We found that alternate initial conditions based on the single gene knock-outs and informed by the MCZ topology predictions were able to achieve an order of magnitude greater data prediction accuracy.

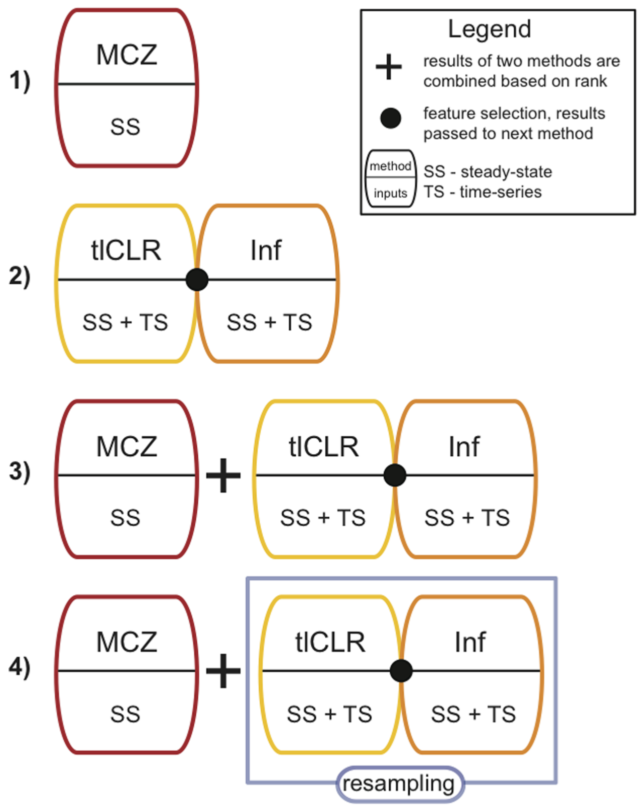

Figure 1. Network inference pipelines tested. We developed and tested four network inference pipelines composed of the methods described in Table 1. Pipeline 1, MCZ, uses median corrected z-scores on the steady state genetic knock-out data (eq. 2). We submitted topology predictions from this method, tying for 1st place. Pipeline 2, tlCLR-Inferelator, (eq. 19) uses tlCLR as a feature selection procedure, followed by further model selection and parameterization by the Inferelator 1.0. This pipeline was previously published and placed 2*nd* for the DREAM3 network inference competition [31]. Pipeline 3, tlCLR-Inferelator+MCZ, (eq. 20, developed to test the complementarity of the topology predictions made by pipelines 1 and 2). Pipeline 3 combines the results form pipelines 1 and 2. Double knock-out predictions from this pipeline were submitted, placing 2*nd* (topology predictions from this pipeline were submitted, placing 8*th*). Pipeline 4, Resampling+MCZ, (eq. 22) presents an alternate way to combine predictions made from complementary methods. In pipeline 4 a resampling approach was applied to pipeline 2 and the results were combined with pipeline 1 as described in the methods sections. Topology predictions from pipeline 4 outperformed those of pipeline 3, and double knock-out predictions were on par with those of pipeline 3.
doi:10.1371/journal.pone.0013397.g001

Problem Set Up

The DREAM4 *in-silico* network reconstruction challenge consists of five synthetic networks of 100 genes used to generate five corresponding datasets. The five networks vary in their topology, chosen to mimic either *Escherichia coli* or *Saccharomyces cerevisiae*, and their dynamical properties, determined by initial conditions and the kinetic parameters chosen for each of the five networks [1]. Stochastic differential equations, followed by the addition of noise proportional to the level of gene expression (as seen in real microarray datasets), were used to generate expression data from each topology.

Denote the expression levels of the genes by $x=(x_1,\ldots,x_N)^T$. We are given four sets of observations: time-series (X^{ts}), wild-type (X^{wt}), knock-out (X^{ko}), and knockdown (X^{kd}). To generate the time-series data a perturbation was introduced into the system for a period of time, and then removed. Measurements were taken at evenly spaced time intervals as the system responded to the perturbation, and as it relaxed. We treated the response of the system to the perturbation and the relaxation of the system once the perturbation was removed as separate time-series experiments. In order to simplify notation and without loss of generality we will assume that X^{ts} is the result of one such time-series experiment with K observations, $t_1,t_2,\ldots,t_K$, (i.e. $x(t_1),x(t_2),\ldots x(t_K)$ are columns in X^{ts}). X^{wt} is composed of the first observation in each time series (of which there are ten), and one provided observation of wild-type expression. To generate the knock-out data the transcription rate of each gene was set to zero (in turn), the network was equilibrated, and the steady-state expression for all genes in the system was measured. Likewise, for the knockdown data the transcription rate of each gene was set to half of its wild-type rate, the network equilibrated, and the steady-state expression levels of all of the genes in the system were measured. For the main challenge participants were presented with this data, but not the underlying network topology or kinetic parameters, and asked to submit a ranked list of regulatory interactions sorted by confidence (highest-confidence interactions at the top of the list). The topology predictions were evaluated by area under the precision recall curve (AUPR) [5]. A perfect prediction would have all true regulatory interactions (i.e. true positives) ranked higher than false regulatory interactions (i.e. true negatives), and an AUPR = 1. A random topology prediction for the DREAM4 networks would have an AUPR close to zero.

In addition to this main challenge participants also had the option of taking part in a bonus-round challenge aimed at assessing a method's ability to predict system-wide behavior in response to new genetic perturbations, the double knock-out challenge. For each network participants were also presented with twenty double knock-out perturbations (in which the transcription rate of a pair of genes was set to zero simultaneously), and asked to predict the steady-state expression of all other genes in response to the perturbation. The accuracy of the prediction was evaluated by calculating the mean square error (MSE) between the actual and predicted expression of the N genes. We now describe the three component RN methods which comprise our network inference pipelines: MCZ, tlCLR, and the Inferelator 1.0.

Core Method 1: Median Corrected Z-scores

The underlying model for the expression data in DREAM4 was generated by stochastic differential equations. Each measurement can be thought of as the observation of only a few cells, as opposed to a population of cells. Accordingly, each measurement of wild-type expression, contained in X^{wt}, is an estimate of the population wild-type expression derived from only a few samples, making it a relatively noisy observation. Thus, any single observation will not accurately describe the population wild-type expression, and methods that rely on population-wide statistics (such as the t-test) will suffer. A natural way to correct for this is to increase the sample size. By taking the mean (or median) of the expression levels for each gene, x_i, over all wild type observations (11 in total) we can improve our estimate of the population mean. We use the median since it is more robust to outliers than the mean.

We further improved our estimate of the population wild-type expression by taking the median of x_i not only with respect to the wild-type observations, X^{wt}, but also with respect to the genetic knock-out data, X^{ko}. We can do so under the assumption that the networks are sparse (i.e. each gene is regulated by relatively few regulators). Thus, in most single knock-out experiments the level of most genes will be an independent measurement of their wild-type expression. Accordingly, we consider the wild-type expression of gene x_i to be the median of its expression in X^{wt} and X^{ko}, and denote this median-corrected estimate of wild-type expression as $x^{wt}=(x_1^{wt},x_2^{wt},\ldots x_N^{wt})$.

Previously, we have observed that the genetic knock-out data, X^{ko}, is very informative in regards to the topology of the network [31]. Yip et al. [48] showed that a simple global noise model to filter out non-significant interactions using genetic knock-out data alone was able to produce regulatory interaction ranks of high quality, resulting in the top-performing method for the DREAM3 *in-silico* network challenge. However, for DREAM4 the noise for each gene was a function of the gene's expression (higher noise for higher expression), more accurately simulating the noise found in real microarray measurements. Thus, we used a method that takes into account a more biologically relevant gene-specific noise model to rank regulatory interactions. A natural way of identifying if a gene, x_i, is a target of a TF, x_j, is by comparing the expression level of x_i when x_j is knocked out to the corrected wild-type expression of x_i, x_i^{wt}. We do so using a median corrected Z-score (MCZ):

$$z(x_i|x_j(-/-))=\frac{x_{i,j}^{ko}-x_i^{wt}}{\sigma_i} \tag{1}$$

where the notation $(-/-)$ indicates a knock-out experiment (i.e. $z(x_i|x_j(-/-))$ denotes to the MCZ score of target gene x_i given that x_j is knocked out), $x_{i,j}^{ko}$ is the expression of gene x_i when x_j is knocked out, and σ_i is the standard deviation of gene x_i over all wild-type and single gene knock-out observations. We use $z(x_i|x_j(-/-))$ as a measure of confidence for each regulatory interaction $x_j \rightarrow x_i$, which we store in:

$$Z^1=\begin{pmatrix} z^1_{1,1(-/-)} & z^1_{1,2(-/-)} & \cdots & z^1_{1,N(-/-)} \\ z^1_{2,1(-/-)} & z^1_{2,2(-/-)} & \cdots & z^1_{2,N(-/-)} \\ \vdots & \vdots & \ddots & \vdots \\ z^1_{N,1(-/-)} & z^1_{N,2(-/-)} & \cdots & z^1_{N,N(-/-)} \end{pmatrix} \tag{2}$$

MCZ performed very well in reconstructing the topology of the network (i.e. ranking regulatory interactions based on confidence), however it cannot be used to learn dynamical models of regulation (and hence cannot be used to make predictions of the system's response to double knock-outs). Additionally, it requires a very complete dataset (knock-out of each gene, in turn) to rank all possible regulatory interactions. Moreover, if a regulator is not highly expressed in the wild type condition then the prediction of its targets using MCZ is not very reliable (in this dataset expression and activity seem to be correlated).

Core Method 2: Time Lagged Context Likelihood of Relatedness

tlCLR is a MI based method that extends the original CLR algorithm to take advantage of time-series data [29]. tlCLR is more general than MCZ in that it can explicitly use both steady state and time series data to make a prediction of network topology. tlCLR has three main steps: 1) model the temporal changes in expression as an ODE, 2) calculate the MI between every pair of genes, 3) apply a background correction (filtering) step to remove least likely interactions. tlCLR treats all of the steady state data in the same manner. Thus, we combined the three steady state datasets (X^{wt},X^{ko},X^{kd}) into one $N\times 2N+11$ matrix, X^{ss} (N knock-out experiments, N knock-down experiments, and 11 wild type observations).

Mutual information is a metric of dependency between two random variables X and Y, which can be defined as [25]

$$I(X;Y)=\sum_{x\in X}\sum_{y\in Y}p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{3}$$

where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p(y)$ are the marginal probabilities that $X=x$ and $Y=y$, respectively. Note that MI is a symmetric measure. Faith et al. [29] have previously shown that Context Likelihood of Relatedness (CLR), a MI based method, performed well at identifying a large portion of the known *E.coli* regulatory associations as well as predicting novel interactions. However, CLR can only predict undirected regulatory interactions, and must rely on additional data to determine directionality (e.g. by knowing that one gene encodes for a TF and the other for an enzyme, directionality can be resolved). By taking advantage of the temporal information available from time-series observations, we have shown that CLR can be extended (in a method we call tlCLR), allowing us to infer directed regulatory interactions, and improving overall performance [31]. At the core of tlCLR's ability to resolve directionality is its reliance on dynamic-MI instead of static-MI. The computation of static and dynamic -MI is described below.

As previously suggested [26–29], MI can be used as a measure of similarity between the expression levels of gene-pairs, $I(x_i,x_j)$, where gene-pairs that show a significantly higher MI scores (compared to other gene-pairs) are more likely to have a regulatory interaction between them. Since $I(x_i,x_j)=I(x_j,x_i)$ both regulatory edges ($x_j\rightarrow x_i$ and $x_i\rightarrow x_j$) are equally likely. We refer to the MI calculated from $I(x_i,x_j)$ as static-MI, because it does not use the temporal information available from time-series data (treating time-series and steady-state data identically).

Step 1: Applying an ODE model to the time-series data

We now describe dynamic-MI, which is motivated by our previous work on the Inferelator 1.0 [40], an ODE-based method. We assume that the temporal changes in expression of each gene, x_i, can be approximated by the linear ODE:

$$\frac{dx_i(t)}{dt}=-\alpha_i x_i+\sum_{j=1}^{N}\beta_{i,j}x_j(t), \qquad i=1,\ldots,N \tag{4}$$

where $\alpha_i>0$ is the first-order degradation rate of x_i and the $\beta_{i,j}$'s are a set of dynamical parameters to be estimated. The value of $\beta_{i,j}$ describes the extent and sign of the regulation of target gene x_i by regulator x_j. We store the dynamical parameters in an $N\times N$ matrix, $\boldsymbol{\beta}$. Note that $\boldsymbol{\beta}$ is typically sparse, i.e. most entries are 0 (reflecting the sparsity of transcriptional regulatory networks). Later, we describe how to calculate the values $\beta_{i,j}$ by the Inferelator 1.0. Now we focus on how to use the time-series data in the context of improving the calculation of MI values between a gene x_i and its potential regulator x_j. Using a finite difference approximation, we can write (4) for time-series experiments as

$$\tau_i\frac{x_i(t_{k+m})-x_i(t_k)}{t_{k+m}-t_k}+x_i(t_k)=\tau_i\sum_{j=1}^{N}\beta_{i,j}x_j(t_k), \qquad i=1,\ldots,N \qquad k=1,\ldots,K-m \tag{5}$$

where $\tau_i=\frac{1}{\alpha_i}$ is related to the half-life, $t_{1/2}$, of x_i by $t_{1/2}=\tau_i\log(2)$, and m defines the time intervals we consider (e.g. $m=1$ corresponds to time intervals composed of consecutive time-points). We have set τ_i to 50 (for all i), which is the time-interval between measurements, assuming that the sampling frequency was on the time order of most regulatory reactions. For DREAM4 we consider two time intervals: those of length 50 ($m=1$) and those of length 100 ($m=2$). We chose to stop at a time interval of 100 because using longer time intervals did not improve the dynamic predictive performance (as estimated by the Inferelator 1.0 cross validation step which will be described below). Note that the time in the DREAM4 datasets was measured in arbitrary units (i.e. it does not correspond to any of the typical time units: seconds, minutes, hours, etc.).

The purpose of the next two steps is to separate the terms in (5) that involve the putative regulators (the explanatory variables) from the terms in (5) that involve the target (the response variable). We do so first for time-series data and then for steady-state data.

For every gene pair (x_i,x_j), we define a time-series response variable,

$$y_i(t_{k+m})=\tau_i\frac{x_i(t_{k+m})-x_i(t_k)}{t_{k+m}-t_k}+x_i(t_k), \qquad m=1,2 \qquad k=1,\ldots,K-m \tag{6}$$

We pair this response variable with a corresponding explanatory variable, $x_j(t_k)$. Both variables were derived from the left and right hand sides of (5), respectively.

For steady state experiments, the derivative, $\frac{dx_i(t)}{dt}$, in (4) equals zero, and we can write (4) as

$$x_i(l)=\tau_i\sum_{j=1}^{N}\beta_{i,j}x_j(l), \qquad i=1,\ldots,N \qquad l=1,\ldots,2N+11 \tag{7}$$

Thus, we define a steady-state response variable,

$$y_i(l)=x_i(l), \tag{8}$$

and a corresponding explanatory variable $x_j(l)$, again both derived from (7). Taking the time-series and steady-state response variables together, we get the final response vector:

$$y_i=(y_i(t_2),\ldots,y_i(t_K),y_i(t_3),\ldots,y_i(t_K),y_i(1),\ldots,y_i(2N+11)) \tag{9}$$

and the final explanatory variables vector:

$$x_j=(x_j(t_1),\ldots,x_j(t_{K-1}),x_j(t_1),\ldots,x_j(t_{K-2}),x_j(1),\ldots,x_j(2N+11)) \tag{10}$$

Note that for time-series data the explanatory variables are time-lagged with respect to the response, and that for time intervals much larger than τ_i (5) limits to steady state behavior.

To simplify notations, we will define R to be the total number of elements in y_i and x_j, and let r iterate over these entries, $y_i=(y_i(1),\ldots,y_i(r),\ldots,y_i(R))$ and $x_j=(x_j(1),\ldots,x_j(r),\ldots,x_j(R))$, i.e. r iterates over corresponding response and explanatory variables. We now explain how we use these response (y_i) and explanatory (x_j) variables to calculate the MI between every pair of genes.

Step 2: Calculating the dynamic-MI between genes

As a measure of confidence for a directed regulatory interaction between a pair of genes ($x_j \rightarrow x_i$) we use, $I(y_i,x_j)$, where a pair that shows a high MI score (relative to other pairs) is more likely to represent a true regulatory interaction. Note that $I(y_i,x_j) \neq I(y_j,x_i)$, making one regulatory direction more likely than the other. We refer to the MI calculated from $I(y_i,x_j)$ as dynamic-MI, as it takes advantage of the temporal information available from time-series data (distinguishing time-series data from steady-state data).

As described above, we calculate $I(x_i,x_j)$ and $I(y_i,x_j)$ for every pair of genes and store the values in the form of two $N \times N$ matrices M^{stat} and M^{dyn}, respectively. Note that M^{stat} is symmetric while M^{dyn} is not. We now briefly describe how tlCLR integrates both static- and dynamic-MI to produce a final confidence score for each regulatory interaction. For a more detailed explanation we refer the reader to [31].

Step 3: Background correction

For each regulatory interaction $x_j \rightarrow x_i$ we compute two positive Z-scores (by setting all negative Z-scores to zero): one for the regulation of x_i by x_j based on dynamic-MI (i.e. based on M^{dyn}),

$$z_1(x_i,x_j) = \max\left(0, \frac{M_{i,j}^{\text{dyn}} - \frac{\sum_{j'} M_{i,j'}^{\text{dyn}}}{N}}{\sigma_i}\right), \tag{11}$$

where σ_i is the standard deviation of the entries in the i'th row of M^{dyn}. And one for the regulation of x_i by x_j based on both static and dynamic-MI,

$$z_2(x_i,x_j) = \max\left(0, \frac{M_{i,j}^{\text{dyn}} - \frac{\sum_{i'} M_{i',j}^{\text{stat}}}{N}}{\sigma_j}\right), \tag{12}$$

where σ_j is the standard deviation of the entries in the j'th column of M^{stat}. We combine the two scores into a final tlCLR score, $z_{i,j}^{tlCLR} = \sqrt{(z_1^2 + z_2^2)}$. Note, that some entries in Z^{tlCLR} are zero, i.e. Z^{tlCLR} is somewhat sparse. For a more detailed description of tlCLR we refer the reader to [31].

Core method 3: Inferelator 1.0

We use Inferelator 1.0 to learn a sparse dynamical model of regulation for each gene x_i. As potential regulators of x_i we consider only the P highest confidence (non-zero) regulators. Such a set of P potential regulators can come from any method that ranks regulatory interactions. For example rankings from MCZ, correlation, mutual information, CLR, or tlCLR can all be used to calculate a set of P highest confidence regulators of x_i. Note that we cannot guarantee that every x_i will have P regulators meeting this criteria, thus we denote by $P^i (P^i \leq P)$ the number of regulators that do. Accordingly, for each gene, x_i, we denote this subset of potential regulators as x^i. We then learn a sparse dynamical model of regulation for each x_i as a function of x^i's (using Inferelator 1.0). We assume that the time evolution in the x_i's is governed by

$$\frac{dx_i(t)}{dt} = -\alpha_i x_i + \sum_{j=1}^{P^i} \beta_{i,j} x_j^i(t), \qquad i = 1, \ldots, N \tag{13}$$

which is exactly (4) with our constraint on the number of regulators. Least Angle Regression (LARS) [39] is used to efficiently implement an l_1 constraint on $\boldsymbol{\beta}$, which minimizes the following objective function, amounting to a least-square estimate based on the ODE (13):

$$\mathcal{E}(\boldsymbol{\beta}) = \sum_{i=1}^{N} \mathcal{E}_i(\boldsymbol{\beta}) \tag{14}$$

where

$$\mathcal{E}_i(\boldsymbol{\beta}) = \sum_{r=1}^{R} \left| y_i(r) - \sum_{j=1}^{P^i} \beta_{i,j} x_j^i(r) \right|^2 \tag{15}$$

under an l_1-norm penalty on regression coefficients,

$$\sum_{j=1}^{P^i} |\beta_{i,j}| \leq s_i \sum_{j=1}^{P^i} |\beta_{i,j}^{\text{ols}}| \tag{16}$$

where $y_i(r)$ and $x_j^i(r)$ are corresponding elements in the response (9) and design vectors (10), $\boldsymbol{\beta}^{\text{ols}}$ is the over-fit ordinary least-squares estimate (i.e. the minimizer of (15) with no penalty), and s_i is a number between 0 and 1 referred to as the shrinkage parameter; setting $s_i = 1$ corresponds to ordinary least-square regression. To avoid over fitting, we chose the shrinkage parameter s_i by ten fold cross-validation at one standard deviation away from the minimum error (as described in [32]). Each resultant model (row of $\boldsymbol{\beta}$) is a parameterization of an ODE describing the temporal evolution of x_i. The l_1 constraint ensures that Inferelator 1.0 results in a sparse matrix, $\boldsymbol{\beta}$, with a small number of entries $|\beta_{i,j}| > 0$. These entries are dynamical parameters that can be used to predict the response of the system to new conditions, such as the removal of genes or future time-points (given initial time points in a time series).

The three methods just described (MCZ, tlCLR, and the Inferelator 1.0) comprise the core network inference methods on which our inference pipelines were built. We now present our four inference pipelines and how they were used to generate topology predictions for the DREAM4 competition. For pipeline 1 (MCZ), a ranked list of regulatory interactions is trivially obtained by using the values in Z^1 (2). For pipilines 2,3,4 the process of combining multiple methods and generating topology predictions is described below.

Pipeline 2: combining results from tlCLR and the Inferelator 1.0

The predictions made by pipeline 2 placed 2^{nd} for the DREAM3 competition, but were not submitted for the DREAM4 competition. The reason we did not use pipeline 2 for DREAM4 is that it was outperformed by pipeline 3 on the DREAM3 data. We present pipeline 2 here to simplify our explanation of pipeline 3 (which is a combination of pipelines 1 and 2). When developing pipeline 2 we suspected that predictions made from two different methods (tlCLR, and the Inferelator 1.0) can be complementary. We have previously shown this to be the case [31]. We combined tlCLR and Inferelator 1.0 in two ways: 1) we use the ranked list of regulatory interactions from tlCLR as a feature selection step for the Inferelator l_1 shrinkage approach, and 2) we combined the ranked list generated by tlCLR with the ranked list generated by the Inferelator 1.0.

Using the tlCLR ranking as a feature selection step for the Inferelator 1.0

The entries of each row, i, of Z^{tlCLR} correspond to a ranking of the potential regulators for x_i. As possible regulators of a gene x_i in the Inferelator 1.0 l_1 model selection step (15) we used the P^i most likely regulators from row i of Z^1. In this way we used the ranking of regulatory interactions predicted by tlCLR as a feature selection step for the Inferelator 1.0. We then combined the ranked list of regulatory interactions generated by each of the two methods. Z^{tlCLR} can be used to rank regulatory interactions based tlCLR. We note that prior to combining the results of tlCLR with those from the Inferelator 1.0 we employed a simple filtration step where we removed all regulatory interactions that had MCZ scores in the lower 50% of all MCZ scores. We now describe how a ranked list of regulatory interactions was calculated by the Inferelator 1.0.

Calculating a ranked list of regulatory interactions by the Inferelator 1.0

The dynamical parameters stored in $\boldsymbol{\beta}$ (the result of minimizing (15) subject to (16)) describe the regulation of each target gene as a function of its regulators (TF's) in the system, with $|\beta_{i,j}|$ corresponding to the strength of the regulation, and the sign of $\beta_{i,j}$ indicating repression or activation. For the DREAM3 *in-silico* challenge we ranked regulatory interactions using $|\beta_{i,j}|$ as the measure of confidence for a regulatory interaction ($x_j \rightarrow x_i$) [31]. However, this ranking does not take into account the explanatory power of each predictor x_j in the ODE model for a target x_i (e.g. $|\beta_{i,j}|$ may be large even though the model for the regulation of x_i is not a good one). Here, we propose a confidence measure that incorporates the explanatory power of predictors (i.e. the quality of the model for x_i).

Denote the predicted expression of $y_i(r)$ as $\hat{y}_i(r)$, calculated as $\hat{y}_i(r) = \tau_i \sum_{j=1}^{P^i} x_j^i(r)\boldsymbol{\beta}_{i,j}$. The error in this approximation of y_i was measured as sum-of-squares, $\sum_{r=1}^{R} (y_i(r) - \hat{y}_i(r))^2$, where R is the number of elements in the response vector, y_i. We estimated the predictive error of our model for y_i using mean error obtained from ten fold cross-validation. In order to place all model errors on the same scale, we normalized the absolute sum-of-squares error to derive a measure of relative error, $\frac{\sum_{r=1}^{R} (y_i(r) - \hat{y}_i(r))^2}{\sum_{r=1}^{R} y_i(r)^2}$. Given this relative error, we defined the explanatory power of the model for y_i to be given by 1 minus relative error:

$$\gamma_i = 1 - \frac{\sum_{r=1}^{R} (y_i(r) - \hat{y}_i(r))^2}{\sum_{r=1}^{R} y_i(r)^2} \tag{17}$$

where γ_i represents the merit of the model for y_i (i.e. how good of an estimate is $\hat{y}_i$). We can now calculate the contribution of each predictor $\beta_{i,j}$ to the explanatory power of the model for y_i, (i.e. the explanatory power of each regulatory interaction) as a weighted average

$$z_{i,j}^{\gamma} = \frac{\beta_{i,j}}{\beta_{i,0} + \sum_{j=1}^{N} \beta_{i,j}} \gamma_i \tag{18}$$

where $\beta_{i,0}$ is the bias term for the regulatory model of y_i. Note that here we use the fact that all the observations of the regulators x_j's, are on the same scale, as they were normalized to have zero mean and standard deviation of 1 before model selection by Inferlator 1.0 (a common step in a regression framework). We stored these values in the form of an $N \times N$ matrix, Z^{γ}, which can be used regulatory interactions based on Inferelator 1.0 alone. However, based on our previous results that the predictions made by different methods may be complementary, we combined the predictions made by tlCLR (Z^{tlCLR}) with those made by Inferelator 1.0 (Z^{γ}).

Combining topology predictions made by tlCLR with those made by the Inferelator 1.0

The main challenge in combining these confidence scores is that they are not guaranteed to be on the same scale. Thus, we developed a single rank based heuristic (described previously in [31]) to combine two separate sets of confidence scores. Our approach is best explained by an example: Let Z^2 denote the resultant matrix from combining the confidence scores contained in Z^{tlCLR} and Z^{γ}, i.e. the results our tlCLR-Inferelator pipeline (pipeline 2, Figure 1). We first replace the value of the highest-ranking entry in Z^{γ} with the value of the highest-ranking entry from Z^{tlCLR}. We then replace the value of the second highest-ranking entry from Z^{γ} with the value of the second highest-ranking entry from Z^{tlCLR}. We continue in such a way until all non-zero entries in Z^{γ} have been replaced by equally ranked entries in Z^{tlCLR}. This produces two ranked lists of regulatory interactions that are on the same scale. Once this assignment is done we can combine the two matrices as follows:

$$z_{i,j}^2 = \sqrt{(z_{i,j}^{\gamma})^2 + (z_{i,j}^{tlCLR})^2}. \tag{19}$$

Note that here Z^{γ} refers to the matrix after the assignment of values from Z^{tlCLR}. Z^2 constitutes the results of applying pipeline 2. Pipeline 3 is very similar to pipeline 2. In order to assess how complementary tlCLR-Inferelator and MCZ were we combined the confidence scores stored in Z^{γ} with those in Z^{tlCLR} (replacing scores of equal ranks from Z^1 into Z^2, as above):

$$z_{i,j}^3 = \sqrt{(z_{i,j}^2)^2 + (z_{i,j}^1)^2}. \tag{20}$$

The confidence scores contained in Z^3 were generated by a combination of our three methods, and we will refer to this integrated method as tlCLR-Inferelator+MCZ (pipeline 3, Fig 1). Z^2 and Z^3 can be used as ranked lists to rank regulator interactions for pipelines 2 and 3, respectively.

Combining genetic and dynamic information in a resampling approach

We generated an ensemble of networks as follows. Denote by Y the $N \times R$ response matrix, with each row set to $y_i = (y_i(1), \ldots, y_i(R))$. Similarly, denote by X the $N \times R$ design matrix, with each row set to $x_j = (x_j(1), \ldots, x_j(R))$. Let $c = \{1, \ldots, R\}$ be the vector of column indices for both Y and X. We sample with replacement R times from c, storing the selected indices in $c^* = i_1, i_2, \ldots, i_j, \ldots, i_R$, $i_j \in \{1, \ldots, R\}$. We now consider the permuted data matrices, Y^* and X^*, comprised of the c^* columns of Y and X respectively. We generate $\boldsymbol{\beta}$, Z^2, and Z^3, as described before, with the only difference being that we use the response and explanatory vectors from Y^* and X^*, respectively, instead of Y and X. We repeat this procedure B times, with $B = 200$ for the DREAM4 networks, each time generating $\boldsymbol{\beta}$, Z^2, and Z^3. We store this ensemble of regulatory network predictions in:

$$E = \{[\boldsymbol{\beta}(1), Z^3(1)], [\boldsymbol{\beta}(2), Z^3(2)], \ldots, [\boldsymbol{\beta}(B), Z^3(B)]\} \tag{21}$$

where $[\boldsymbol{\beta}(b), Z^3(b)]$ corresponds to the dynamical parameters and rankings based on tlCLR-Inferelator+MCZ (pipeline 3, Figure 1) at resample $b=1,2,\ldots,B$. Note that throughout this resampling procedure the ranks generated by MCZ remain constant. We used this ensemble of network predictions by selecting, for each regulatory interaction $x_j \rightarrow x_i$ (corresponding to entries $\beta_{i,j}$ and $z^3_{i,j}$), the median dynamical parameters in $\{\boldsymbol{\beta}(1), \boldsymbol{\beta}(2), \ldots, \boldsymbol{\beta}(B)\}$ and the median tlCLR-Inferelator+MCZ rank in $\{Z^3(1), Z^3(2), \ldots Z^3(B)\}$. We store these median values in $\boldsymbol{\beta}^{\text{median}}$ and Z^4, respectively. These matrices have entries:

$$z^4_{i,j} = \text{median}(z^3_{i,j}(1), z^3_{i,j}(2), \ldots, z^3_{i,j}(B)) \tag{22}$$

$$\beta^{\text{median}}_{i,j} = \text{median}(\beta_{i,j}(1), \beta_{i,j}(2), \ldots, \beta_{i,j}(B)). \tag{23}$$

We used Z^4 to rank the regulatory interactions, and we refer to this resampling approach as Resampling+MCZ (pipeline 4, Figure 1). $\boldsymbol{\beta}^{\text{median}}$ is a set of dynamical parameters which can be used to predict the response of the system to new perturbations (such as the simultaneous knock-out of two regulators).

Bonus round: Generating the double knock-out predictions

The challenge of predicting the response of the system to double knock-outs (double-KO) can be phrased as: given a simultaneous knock-out of two genes (i.e. $x_i, x_j = 0$ for some i and j), predict the steady-state expression of all other genes. In order to predict steady-state expression levels for each gene we used the steady-state limit of the core Inferelator 1.0 model [32] (7), which we rewrite here (in matrix notation) for the case of predicting the steady state data:

$$x^{ij(-/-)} = \tau_i \boldsymbol{\beta} x^0 \tag{24}$$

where $x^{ij(-/-)}$ is the level of all genes for the double-KO of genes x_i and x_j, and $x^0 = (x^0_1, \ldots, x^0_N)^T$ is some vector of initial conditions (satisfying $x^0_i, x^0_j = 0$). Note, that for DREAM4 we made a simplification, setting $\tau_i = 50$ for all i, i.e. we assume that all mRNA have the same half-life. The only unknown left to determine (in order to make a prediction) is the vector of initial conditions, x^0. The rest of this section deals with computing a good initial condition vector.

A simple way to pick this vector would be to set $x^0 = x^{wt}$, with the exception that $x^0_i, x^0_j = 0$. The results that we submitted for the DREAM4 bonus-round challenge were calculated using this initial condition. Note, however, that the system's response to the double-KO of genes x_i, x_j individually was already given to us in the single-gene knock-out dataset, X^{ko}. Upon revisiting our initial results, after submission of the predictions, we reasoned that using the single gene knock-out (single-KO) information to predict double-KO expression would most likely yield better results, as it reflects a system state that is closer to the state we are trying to predict. Indeed, using the single-KO data to determine initial conditions markedly improved the accuracy of our double knock-out predictions.

One simple approach to construct initial conditions from the single gene knock-outs of x_i and x_j is to simply take their mean. However, we chose to use a more informed approach by taking advantage of our previous knowledge regarding likely regulatory interactions (i.e. the confidence scores from MCZ (stored in Z^1). We do so by computing the following weighted average:

$$x^0_l = \frac{z^{ko}_{l,i} x^{ko}_{l,i} + z^{ko}_{l,j} x^{ko}_{l,j}}{z^{ko}_{l,i} + z^{ko}_{l,j}} \qquad l = 1, \ldots, N \tag{25}$$

where x^0_l is our estimate for the initial expression level of gene x_l, $x^{ko}_{l,i}$ and $x^{ko}_{l,j}$ are the observed levels of x_l when genes x_i and x_j were knocked out, respectively, and $z^{ko}_{l,i}$ or $z^{ko}_{l,j}$ are the confidence scores (calculated by MCZ) for each regulatory interaction $x_i \rightarrow x_l$ and $x_j \rightarrow x_l$, respectively. In this manner we computed an initial condition vector, x^0, for every double-KO we were asked to predict. We then used these initial conditions to calculate a prediction of the expression of all genes in the presence of a double-KO of x_i, x_j via (24). We denote this prediction as $\tilde{x}^{ij(-/-)}$.

Note that some models had more predictive merit than others, as measured by the explanatory power of each model (17). Thus we weighted the prediction of double knock-outs by the predictive merit of each model. We computed the final double-KO predictions as follows:

$$x^{ij(-/-)} = \tilde{x}^{ij(-/-)} \gamma + x^0 (\mathbf{1} - \gamma) \tag{26}$$

where $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_N)^T$. Note that in (26) the final prediction $x^{ij(-/-)}$ is weighted by our estimate of the predictive performance of the models, γ calculated in (17), and constrained, using the initial conditions, by our estimate of the model errors $(1-\gamma)$.

Results

Performance of tested methods: ranking putative regulatory interactions

The main challenge in the DREAM4 100 gene *in-silico* regulatory network competition was to predict the topology of five networks. Predictions were made in the form of a list of regulatory interactions ranked in decreasing order by confidence. We evaluated the performance of four pipelines for learning regulatory networks, namely: MCZ (pipeline 1, eq. 2), tlCLR-Inferelator (pipeline 2, eq. 19), tlCLR-Inferelator+MCZ (pipeline 3, eq. 20), and Resampling+MCZ (pipeline 4, eq. 22). We developed these pipelines with a focus on combining results from multiple methods in a mutually reinforcing manner. In all four cases we evaluated the quality of the rankings of all possible regulatory interactions using the area under precision recall curve (AUPR), as this was the basis for the evaluation of performance in DREAM3 and DREAM4.

We submitted the results of MCZ as our ranked list of regulatory interactions for the DREAM4 challenge. This method tied for first place (out of 19 teams). In Figure 2 we see that pipeline 2 exhibits lower performance for most of the networks. In pipeline 3 we combined the predictions made by pipeline 1 with those made by pipeline 2. As expected for methods that are not complementary, the performance of pipeline 3 is better than that of pipeline 2 but worse than that of pipeline 1. However, by using a resampling approach, pipeline 4 (eq. 22), to generate an ensemble of likely networks we see a marked improvement over the performance of any other method (Figure 2, purple bars). This improvement is most evident in networks 3–5, which appear to be more difficult to predict for all of the methods we tested.

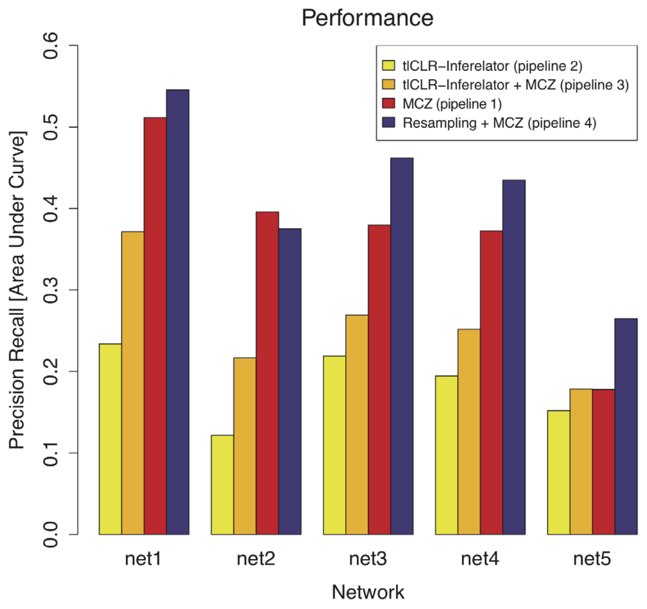

Figure 2. Area under precision recall curve for each ranking scheme. For each pipeline we evaluated the performance in predicting topology using area under the precision recall curve (AUPR). We see that pipeline 4 generally outperforms all other methods, followed by MCZ, pipeline 3, and pipeline 2.
doi:10.1371/journal.pone.0013397.g002

Performance of methods based on genetic knock-out data decreases with decreasing expression of the regulators

For the DREAM3 *in-silico* challenge all methods, including several similar to the ones we test herein, were found to perform significantly worse for networks with very high in-degree (targets regulated by many TFs) and to be relatively insensitive, performance-wise, to the out-degree of TFs [31,49]. We did not find this trend in the current challenge (Figure 3A,B). However, we did find that performance varies considerably across the five 100 gene networks for all tested methods; performance was best for the first network and dramatically worse for the fifth network (Figure 2). We investigated possible reasons for this, finding that performance is correlated with the median expression of the regulators. Given a regulatory interaction, $x_j \rightarrow x_i$, our chance of correctly predicting that regulatory interaction (based on MCZ) tends to be higher if the median expression of x_j over all conditions in the knock-out data-set, X^{ko}, is high. Conversely, the smaller the median expression of x_j, the worse our performance. Figure 3C shows that our predictions for the regulatory interactions in network 1 have relatively low error (black box plot), and the corresponding median expression of the regulators in this network is relatively high (gray box plot). For network 5 we see a relatively high error, and the corresponding median expression of the regulators in this network is the lowest of the five networks. In Figure 3D we see a high correlation, $R^2 = .95$, between the median expression of the regulators and the performance of MCZ in terms of AUPR. By combining ranks from MCZ with our resampled network inference pipeline, pipeline 4, we significantly improve performance on networks 3–5 (Figure 2), and lower the correlation between performance and median TF expression over all five networks to $R^2 = .81$ (Figure 3D).

Regulatory interaction rankings derived from genetic knock-out data and rankings derived from resampling pipeline 2 are complementary

In the above section we focused on differences between the performance of each method for each of the five networks. In this section we focus on the performance of each method in a gene-by-gene manner, in an effort to better understand how to best utilize heterogeneous data collections. Specifically, we investigated the performance of each network inference pipeline as a function of the median expression of the regulators in the network. We bin regulators based on their median expression, and compare the error made in predicting their respective targets.

In Figure 4 we see that the performance of MCZ is better for regulators with a higher median expression (shown in red). This trend is more apparent in this gene-by-gene view than in our network-centric analysis. Looking at each bin, shown from low to high median expression, we see that predictions made by pipelines that incorporate rankings made by tlCLR-Inferelator perform better than the predictions made by MCZ for regulators whose median expression is low (bins 0.1, 0.2). The error distributions of the predictions made by pipeline 4 (purple bars) are lower than those of MCZ (red bars) for regulators with a median expression upto 0.4, and on par with the predictions made by MCZ for regulators with a median expression of up to 0.6. The predictions made by pipeline 4 are better than those made by pipelines 2 and 3 for all bins.

Predicting response of the system to double knock-out

For each 100-gene network we were asked to predict the cell's steady-state mRNA levels given that a pair of genes is knocked out. There are twenty such pairs of genes (x_i,$x_j = 0$) for each network. We make these predictions using the parameterization, $\boldsymbol{\beta}$, of the system obtained from pipeline 3 (tlCLR-Inferelator+MCZ). We also make these predictions using the parameters obtained by taking the median weight from the ensemble, $\boldsymbol{\beta}^{\text{median}}$ (eq. 23), generated by pipeline 4 (Resampling+MCZ). The measure of performance for the DREAM4 double knock-out predictions was mean squared error (MSE). As a baseline, we compare the error of our prediction to the error we would make if we used the initial conditions as the prediction.

In Figure 5 we bin regulators based on their median expression and show the corresponding error distributions for our predictions. We compare our error to the error made if we used the initial conditions as a prediction of the response of the system. In Figure 5A we use the wild type expression, x^{wt}, as the set of initial conditions. We see that predictions made using either pipeline 3 (gray boxplots) or pipeline 4 (red boxplots) outperform the initial conditions (green boxplots). In Figure 5B we construct our initial conditions from the given single gene knock-out values and our MCZ confidence scores (eq. 25). We see that our predictions (black and red boxplots) outperform the initial conditions (green boxplots). Furthermore, by comparing the green boxplots in Figure 5B to those in Figure 5A, we see that predictions based on initial conditions derived from the single knock-out data have much lower error than predictions based on initial conditions derived from the wild type. Regardless of which initial conditions are chosen, predictions using parameters derived from pipeline 4 show almost identical performance as those made by using the parameterization derived from pipeline 3.

Discussion

We participated in the DREAM4 100-gene *in-silico* network inference competition. The method that we submitted, and that

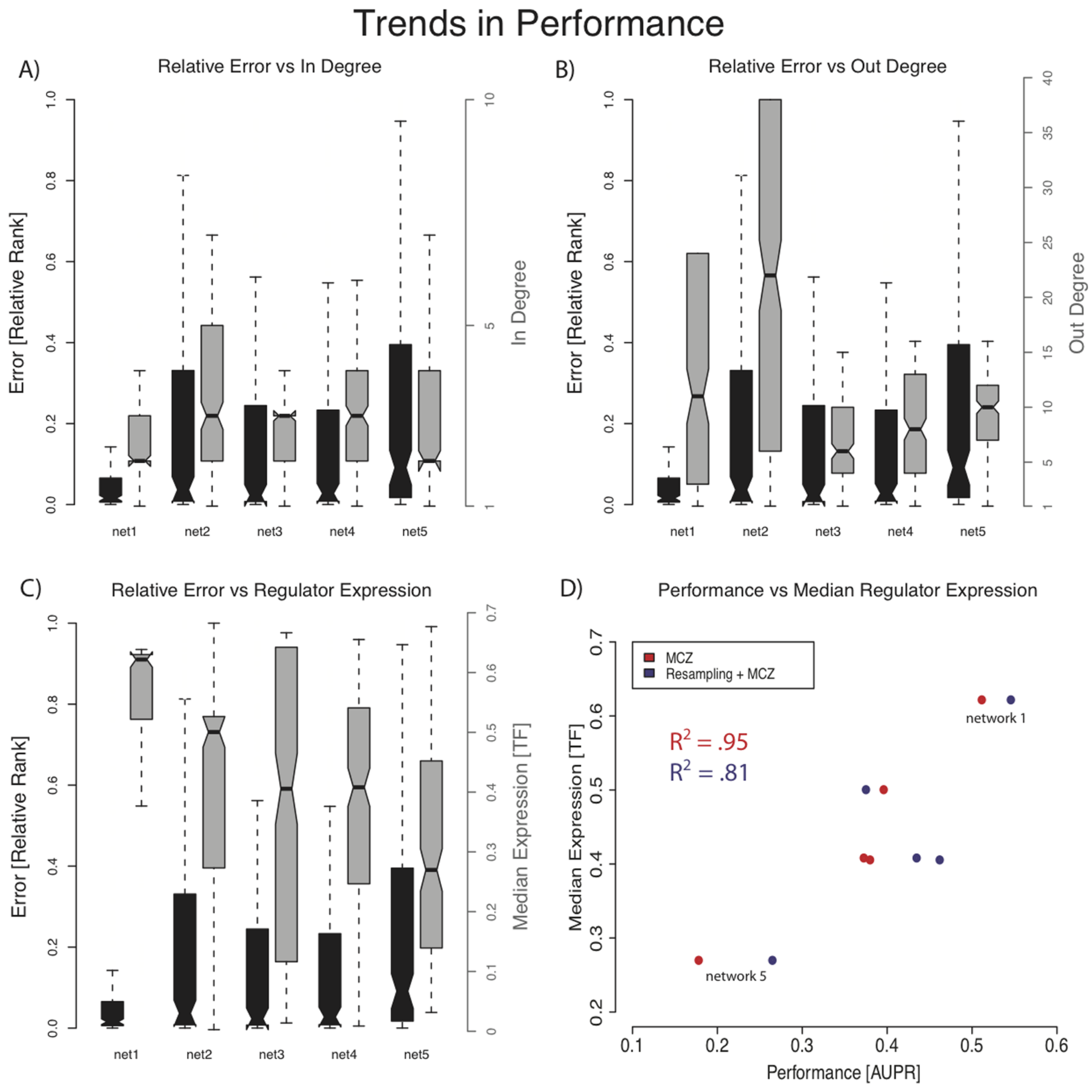

Figure 3. Trends in performance over the five networks. For panels A,B,C we consider only the performance of MCZ, and use relative rank as an estimate of error. We compute relative rank in the following way. Denote by L the total number of possible regulatory interactions, and by l the rank that was given to each regulatory interaction, $x_j \to x_i$. The relative rank of $x_j \to x_i$ is defined to be $\frac{l}{L}$. Error distributions of the predictions for the five networks are shown as black boxplots in panels A,B,C. Distributions of in-degree of the regulators, out-degree of the regulators, and median expression of the regulators are shown as gray boxplots in panels A,B,C, respectively. **A**) There is no apparent relationship between relative rank (Error) and the in-degree of the regulators. **B**) There is no apparent relationship between relative rank (Error) and the out-degree of the regulators. **C**) Relative rank (Error) in network prediction increases as the median expression of the regulators decreases. **D**) we show the relationship between median expression of the regulators and the performance in ranking regulatory interactions, in terms of AUPR, across all five networks. For MCZ a correlation of ($R^2 = .95$) exists between the TFs median expression and AUPR (shown in red), while for Resampling+MCZ there is a smaller correlation of ($R^2 = .81$) between the TFs median expression and AUPR (shown in purple).
doi:10.1371/journal.pone.0013397.g003

was the co-best performer on the 100-gene *in-silico* challenge, was the rankings derived from the median corrected Z-scores of the genetic knock-out data, MCZ. The power of the genetic knock-out data, as also shown by Yip et al. in DREAM3 [48], is an important point to consider for experimental design. However, it does have limitations for which we compensated by integrating other data-types, particularly time-series data. We observed that as the median expression of the regulators in a network decreases, error in predicting regulatory interactions using MCZ increases (Figure 3). A plausible explanation for why a low median expression of regulators leads to poor performance is that if a regulator that is more likely to be active (i.e. a regulator whose wild-type expression is high) is removed, then the corresponding effect on its targets will be relatively large. Conversely, if a regulator that is less likely to be active (i.e. its wild-type expression is low) is removed, then the effect on its targets will be marginal. Perhaps, the targets of such regulators will be most apparent in over-expression experiments. If over-expression experiments are not available, the poor performance in predicting the targets of these regulators can be mitigated by combining the predictions made by MCZ with predictions made by a method that takes advantage of time-series data.

We used pipeline 3 (tlCLR-Inferelator + MCZ), which takes advantage of the time-series data, to predict topology and

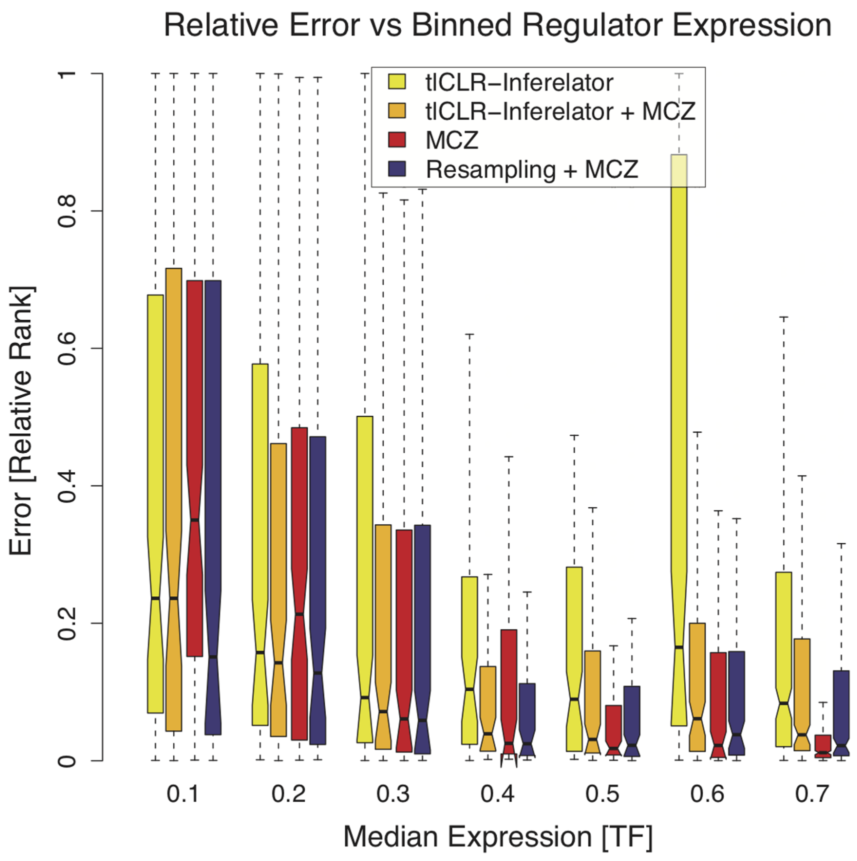

Figure 4. Error as a function of binned median expression for all regulatory interactions. We further investigate the relationship between the median expression of the regulators and each pipeline's performance in predicting topology. We use relative rank as an estimate of error (as in Figure 3). We bin the regulators for all five networks based on their median expression (each of the seven bins has a roughly equal number of regulators). We show the distribution of relative ranks (Error) for each pipeline in each bin of regulator expression. We see that all of the pipelines that incorporate the predictions of tlCLR-Inferelator (pipelines 2,3, and 4) outperform MCZ for regulators with low median expression (bins 0.1, 0.2).
doi:10.1371/journal.pone.0013397.g004

dynamical parameters for each network in a way that was more robust to the median expression of the regulators than methods that use solely genetic knock-out data. We submitted topology predictions and bonus-round (double knock-out) predictions generated by pipeline 3. The topology predictions ranked 8th out of 19 teams. Pipeline 3 is an improvement upon pipeline 2 in terms of AUPR (Figure 2). Pipeline 2 ranked *2nd* out of 22 on the DREAM3 100-gene network inference challenge [31]. However, pipeline 3 ranked *8th* on the DREAM4 challenge. This discordance between a worse performance relative to other teams, but improved ability to recapitulate network topology is probably due to a more concentrated use of the knock-out data by participants of DREAM4.

Upon receiving the gold-standard networks we analyzed our ability to rank regulatory interactions using the different pipelines. Dissecting our performance, in a gene-by-gene manner, we saw the that there are instances when predictions made by pipeline 3 are more accurate than those made by MCZ. Given the performance of each of the methods, as evaluated by AUPR (Figure 2), this is a surprising and promising result, implying that methods that use only genetic knock-out data and those that take advantage of time-series data produce complementary topology predictions. Further demonstrating this point, we showed that applying a resampling approach to pipeline 3 and combining the results with MCZ, by aggregating the ranks derived from each method, produces a final prediction that is better than the predictions generated by either method alone. The improvements from resampling (pipeline 4) are most evident on networks 3–5 (Figure 2), which have the lowest median expression of the regulators (Figure 3A), and are hence hardest to predict using the genetic knock-out data alone. We note that alternate ways of combining predictions from multiple methods may further improve upon our results. We also note that in pipeline 4 the predictions of MCZ remain constant for each network in the ensemble. This implies that although a single network generated by pipeline 3 may perform poorly, our resampling approach generates sufficient alternate topologies such that picking a network based on the ensemble-median produces a much more accurate topology prediction. This resampling approach also infers an ensemble of dynamical parameters, retaining the ability to predict the response of the network to new conditions.

We submitted predictions of system-wide expression in the presence of double knock-outs for the DREAM4 bonus-round challenge. The predictions we submitted were based on the initial conditions derived from wild-type expression levels (x^{wt}). The quality of our double knock-out predictions was very sensitive to the initial conditions (Figure 5). We found that using the single gene knock-out data together with MCZ confidence scores as the basis of our initial conditions dramatically improves our predictive performance (compared to using initial conditions based on the wild-type). This is due to the fact that the single-gene knock-outs present a closer network state to the state we are trying to predict

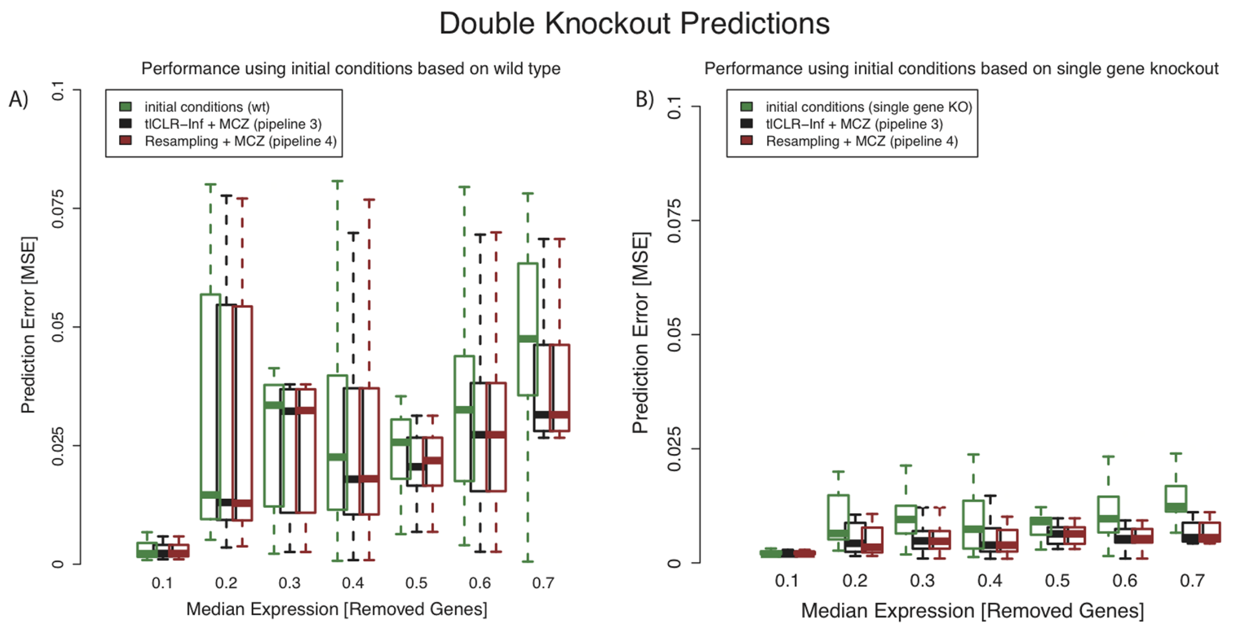

Figure 5. Performance on double knock-out prediction. We assess the accuracy of predicting the system's response to the simultaneous removal (knock-out) of two genes x_i,x_j. In total, there were one-hundred pairs of genes that were knocked out. We bin these pairs of genes based on the average of their respective median expression in the single-gene knock-out data. We made two predictions, which differ only in the choice of initial conditions. We compare the error (as evaluated by the mean squared error) of our prediction to the error made by using the respective initial condition as a prediction. **A**) We use the wild-type expression, x^{wt}, as the set of initial conditions (green boxplots). We see that our predictions (black and red boxplots) are more accurate than if we used the initial conditions as a prediction (this is more apparent for TFs with a larger median expression). **B**) We use a combination of the single-gene knock-outs to compute our initial conditions (eq. 25). We do this because the single-gene knock-out data represents a system state that is closer to the state we are trying to predict than wild-type (as can be observed by comparing the green boxplots in panel A to those in panel B). We show the error distributions using parameters calculated by either pipeline 3 (tlCLR-Inferelator+MCZ) or pipeline 4 (Resampling+MCZ), gray and red boxplots, respectively, are smaller than the error distributions if we used the initial conditions as a prediction. Regardless of the choice of initial conditions, the error distributions using parameters calculated by pipeline 4 (red boxplots) are similar to the error distribution obtained by pipeline 3.
doi:10.1371/journal.pone.0013397.g005

(network response to double knock-outs) than does the wild-type. When using initial conditions based on the single gene knock-out data we saw that our improvement over these initial conditions was larger than when using initial conditions based on wild type (Figure 5). This is an interesting observation since one might expect that it would be harder to improve upon initial conditions that are already close to the true answer. We accurately predicted the response of the network to double knock-outs using dynamical

Table 1. Salient characteristics of the three core methods.

	MCZ	*tlCLR*	*Inf*
input data			
optimal for comprehensive knock-out data	√		
optimal for ts data		√	√
output			
topology (directed network)	√	√	√
kinetic parameters (can predict system's response)			√
statistical approach			
t-statistic	√		
mutual information		√	
regression			√

Here we present the characteristics of the three core network inference methods, combinations of which constituted network inference pipelines. Median corrected z-score (MCZ) uses the t-statistic on solely the steady state genetic knock-out data, and can predict the topology of the network. Time-lagged Context Liklihood of Relatedness (tlCLR) is a mutual information based method that uses both time-series and steady state data to predict the topology of the network. The Inferelator 1.0 (Inf) is an ordinary differential equation (ODE) based method that uses both time-series and steady-state and can predict not only the topology of the network but also the kinetic parameters of regulation, allowing for the prediction of the response of the system to new conditions.
doi:10.1371/journal.pone.0013397.t001

parameters calculated by Pipeline 3 (whose topology prediction was poor relative to those of other methods). Thus, we show that our ability to predict data can tolerate a remarkable amount of error in the predicted topology and still make accurate predictions of the system's response to new perturbations. This is perhaps not surprising, as the Inferelator 1.0 [32] was designed to minimize data prediction error. We have also shown that a parameterization picked from the median of an ensemble of networks (generated by resampling pipeline 3) retains, but does not significantly improve, our ability to predict data in the double knock-out challenge (in spite of the fact that this method produces more accurate topology predictions). Perhaps an alternative way of picking parameters from the ensemble of networks can improve upon the ability to predict new data.

We have shown the complementarity between predictions made using genetic knock-out data and those made using time series data. We have shown that using solely genetic knock-out data can result in accurate topology predictions, which can be further improved upon by correctly incorporating predictions made using time-series data. To this end, we have developed a relatively simple method for combining the predictions made from genetic knock-out and time-series datasets, showing an improved ability to infer network topology while maintaining the ability to predict the response of the system to new conditions. We suggest that investigating alternate means of combining genetic and dynamic experimental designs (leveraging the complementarity between these two data-types), as well as methods that incorporate direct binding data, will continue to be fruitful avenues of future investigation.

Acknowledgments

The authors would like to thank Dennis Shasha, David Reiss, Eric Vanden-Eijnden, Yongzhao Shao, Thadeous Kacmarczyk, Peter Waltman, and Francis Kirigin, for their helpful comments and discussions. We would like to thank the DREAM4 organizers for their tireless efforts.

Author Contributions

Conceived and designed the experiments: AG AM HO RB. Performed the experiments: AG AM. Analyzed the data: AG AM. Wrote the paper: AG AM RB.

References

1. Marbach D, Schaffter T, Mattiussi C, Floreano D (2009) Generating Realistic in silico Gene Networks for Performance Assessment of Reverse Engineering Methods. Journal of Computational Biology 16: 229–239.
2. Haynes BC, Brent MR (2009) Benchmarking regulatory network reoncstruction with GRENDEL. Bioinformatics 25: 801–807.
3. Mendes P, Sha W, Ye K (2003) Artificial gene networks for objective comparison of analysis algorithms. Bioinformatics 19: 122–129.
4. Cantone I, Marucci L, Iorio F, Ricci MA, Belcastro V, et al. (2009) A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches. Cell 137: 172–81.
5. Prill RJ, Marbach D, Saez-Rodriguez J, Sorger PK, Alexopoulos LG, et al. (2010) Towards a rigorous assessment of systems biology models: the DREAM3 challenges. PloS one 5: e9202.
6. Horak CE, Snyder M (2002) ChIP-chip: a genomic approach for identifying transcription factor binding sites. Methods in enzymology 350: 469–83.
7. Johnson DS, Mortazavi A, Myers RM, Wold B (2007) Genome-wide mapping of in vivo protein-DNA interactions. Science 316: 1497–502.
8. Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences of the United States of America 95: 14863–8.
9. Tanay A, Sharan R, Shamir R (2002) Discovering statistically significant biclusters in gene expression data. Bioinformatics 18: 136–144.
10. Ihmels J, Friedlander G, Bergmann S, Sarig O, Ziv Y, et al. (2002) Revealing modular organization in the yeast transcriptional network. Nat Genet 31: 370–377.
11. Bergmann S, Ihmels J, Barkai N (2003) Iterative signature algorithm for the analysis of large-scale gene expression data. Physical review E 67: 31902.
12. Reiss DJ, Baliga NS, Bonneau R (2006) Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. BMC bioinformatics 7.
13. Mardis ER (2007) ChIP-seq: welcome to the new frontier. Nature methods 4: 613–4.
14. Schmid CD, Bucher P (2007) ChIP-Seq data reveal nucleosome architecture of human promoters. Cell 131: 831–2; author reply 832–3.
15. Facciotti MT, Reiss DJ, Pan M, Kaur A, Vuthoori M, et al. (2007) General transcription factor specified global gene regulation in archaea. Proceedings of the National Academy of Sciences of the United States of America 104: 4630–5.
16. Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, et al. (2002) Transcriptional regulatory networks in Saccharomyces cerevisiae. Science 298: 799–804.
17. Gama-Castro S, Jiménez-Jacinto V, Peralta-Gil M, Santos-Zavaleta A, Peñaloza Spinola MI, et al. (2008) RegulonDB (version 6.0): gene regulation model of Escherichia coli K-12 beyond transcription, active (experimental) annotated promoters and Textpresso navigation. Nucleic acids research 36: 120–4.
18. Matys V, Wingender E (2003) TRANSFAC(R): transcriptional regulation, from patterns to profiles. Nucleic Acids Research 31: 374–378.
19. D'haeseleer P, Shoudan L, Somogyi R (2000) Genetic network inference: from co-expression clustering to reverse engineering. Bioinformatics 16: 707–726.
20. Smith VA, Jarvis ED, Hartemink AJ (2003) Influence of network topology and data collection on network inference. Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing. pp 164–75.
21. Bonneau R (2008) Learning biological networks: from modules to dynamics. Nature Chemical Biology 4: 658–664.
22. He F, Balling R, Zeng Ap (2009) Reverse engineering and verfication of gene networks: principles, assumptions and limiations of present methods and future perspectives. Journal of Biotechnology 144: 190–203.
23. Hecker M, Lambeck S, Toepfer S, Someren EV, Guthke R (2009) Gene regulatory network inference: Data integration in dynamic models—A review. BioSystems 96: 86–103.
24. Bansal M, Belcastro V, Ambesi-Impiombato A, di Bernardo D (2007) How to infer gene networks from expression profiles. Molecular systems biology 3: 78.
25. Shannon C (1948) A Mathematical Theory of Communication. The Bell System Technical Journal 27: 379–423.
26. Basso K, Margolin AA, Stolovitzky G, Klein U, Dalla-Favera R, et al. (2005) Reverse engineering of regulatory networks in human B cells. Nature genetics 37: 382–90.
27. Margolin AA, Nemenman I, Basso K, Wiggins C, Stolovitzky G, et al. (2006) ARACNE: An Algorithm for the Reoncstruction of Gene Regulatory Networks in a Mammalian Cellular Context. BMC Bioinformatics 15: 1–15.
28. Butte AJ, Kohane IS (2000) Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. Proceedings of the 5th Pacific Symposium on Biocomputing. pp 418–429.
29. Faith JJ, Hayete B, Thaden JT, Mogno I, Wierzbowski J, et al. (2007) Large-Scale Mapping and Validation of Escherichia coli Transcriptional Regulation from a Compendium of Expression Profiles. PLoS Biology 5: 54–66.
30. Liang KC, Wang X (2008) Gene regulatory network reconstruction using conditional mutual information. EURASIP Journal on Bioinformatics & Systems Biology 2008.
31. Madar A, Greenfield A, Vanden-Eijnden E, Bonneau R (2010) Dream3: Network inference using dynamic context likelihood of relatedness and the inferelator. PLoS ONE 5: e9803.
32. Bonneau R, Reiss DJ, Shannon P, Facciotti MT, Hood L, et al. (2006) The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets. Genome Biology 7: doi: 10.1186/gb-2006-7-5-r36.
33. Mazur J, Ritter D, Reinelt G, Kaderali L (2009) Reconstructing nonlinear dynamic models of gene regulation using stochastic sampling. BMC Bioinformatics 10: 448.
34. Madar A, Greenfield A, Ostrer H, Vanden-Eijnden E, Bonneau R (2009) The inferelator 2.0: A scalable framework for reconstruction of dynamic regulatory network models. Conference proceedings : Annual International Conference of the IEEE Engineering in Medicine and Biology Society IEEE Engineering in Medicine and Biology Society Conference 1: 5448–51.
35. Gardner TS, Collins JJ, di Bernardo D, Lorenz D (2003) Inferring Genetic Networks and Identifying Compound Mode of Action via Expression Profiling. Science. pp 102–105.
36. van Someren EP, Vaes BLT, Steegenga WT, Sijbers AM, Dechering KJ, et al. (2006) Least absolute regression network analysis of the murine osteoblast differentiation network. Bioinformatics 22: 477–84.
37. Bansal M, Gatta GD, di Bernardo D (2006) Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. Bioinformatics 22: 815–22.
38. di Bernardo D, Thompson MJ, Gardner TS, Chobot SE, Eastwood EL, et al. (2005) Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. Nature Biotechnology 23: 377–83.
39. Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Annals of statistics data. pp 407–451.

40. Bonneau R, Facciotti MT, Reiss DJ, Schmid AK, Pan M, et al. (2007) A Predictive Model for Transcriptional Control of Physiology in a Free Living Cell. Cell 131: 1354–1365.
41. Alvarez M, Luengo D, Lawrence N (2009) Latent Force Models. Proceedings of the Twelfth International Workshop on Artificial Intelligence and Statistics 5: 9–16.
42. Efron B, Tibshirani R (1993) An Introduction to the Bootstrap. Boca Raton: FL: Chapman & Hall.
43. Shasha D, Manda W (2008) Statistics is Easy! Morgan & Claypool. doi: 10.2200/S00142ED1V01Y200807MAS001.
44. Kerr MK, Churchill GA (2001) Bootstrapping cluster analysis: assessing the reliability of conclusions from microarray experiments. Proceedings of the National Academy of Sciences of the United States of America 98: 8961–5.
45. Kirk P, Stumpf M (2009) Gaussian process regression bootstrapping: exploring the effects of uncertainty in time course data. Bioinformatics 25: 1300.
46. Friedman N, Goldszmidt M, Wyner A (1999) Data analysis with Bayesian networks: A bootstrap approach. In: UAI'99 Citeseer. pp 206–215.
47. Marbach D, Mattiussi C, Floreano D (2009) Combining multiple results of a reverse-engineering algorithm: application to the DREAM five-gene network challenge. Annals of the New York Academy of Sciences 1158: 102–13.
48. Yip KY, Alexander RP, Yan KK, Gerstein M (2010) Improved Reconstruction of In Silico Gene Regulatory Networks by Integrating Knockout and Perturbation Data. PLoS ONE 5: e8121.
49. Marbach D, Prill RJ, Schaffter T, Mattiussi C, Floreano D, et al. (2010) Revealing strengths and weaknesses of methods for gene network inference. Proceedings of the National Academy of Sciences of the United States of America 107: 6286–91.