Data-efficient Machine Learning in the Life Sciences

Inaugural dissertation

for the attainment of the title of doctor
in the Faculty of Mathematics and Natural Sciences
at the Heinrich Heine University Düsseldorf

presented by

Christopher Blum
from Hilden

Düsseldorf, July 2018

from the Institute of Mathematical Modeling of Biological Systems
at the Heinrich Heine University of Düsseldorf

Published by permission of the
Faculty of Mathematics and Natural Sciences at
Heinrich Heine University Düsseldorf

Correspondents:

1. Prof. Dr. Markus Kollmann

2. Prof. Dr. Gunnar Klau

Date of the oral examination: 2018-12-10

Statement of authorship

I hereby declare that this dissertation is the result of my own work. No other person's work has been used without due acknowledgment. This dissertation has not been submitted in the same or similar form to other institutions. I have not previously failed a doctoral examination procedure.

Christopher Blum

Contents

Manuscripts included in this thesis

publication
C. F. Blum, N. Heramvand, A. S. Khonsari & M. Kollmann. Experimental noise cutoff boosts inferability of transcriptional networks in large-scale gene-deletion studies.
Published in *Nature Communications* **9**(1), 2018.

manuscript I
C. F. Blum & M. Kollmann. Circularly shifted filters enable data-efficient motif inference.
Submitted to *Bioinformatics*.

manuscript II
L. Zhao, N. Abedpour, C. F. Blum, P. Kolkhof, M. Beller, M. Kollmann & E. Capriotti. Predicting gene expression changes in E. coli from mRNA sequence information.

A note to the reader

This thesis was written with the intention to be intelligible to readers from biology or machine learning. To allow faster reading,

comprehensive introductions on Biology are marked with a green line and

general sections on machine learning are marked with a gray line.

Nomenclature

DNA	Deoxyribonucleic acid. Molecular information storage for traits.
RNA	Ribonucleic acid
mRNA	Messenger RNA. Molecular copies of genes that serve as templates for protein production.
ribosome	Molecular machinery that produces proteins based on mRNA templates.
rRNA	Ribosomal RNA, components of ribosomes.
Translation	Protein synthesis. Based on messenger RNA templates.
Transcription	RNA synthesis. Based on DNA templates (genes).
synonymous	Synonymous changes in DNA or mRNA are changes that do affect the corresponding protein's sequence.
Phenotype	Observable trait of an organism, e.g. appearance, metabolism or growth rate.
Genotype	Gene variants responsible for a phenotype.
MFE	Minimum free energy. For example, the energy that is released when an RNA molecule assumes its most favorable structure.
RNAfold	Program of the ViennaRNA Package to predict the RNA MFE structure
NCBI	The National Center for Biotechnology Information (Bethesda, MD, USA) maintains databases and provides bioinformatics services. It is part of the United States National Library of Medicine, which in turn is part of the USA's National Institute of Health.
WT	Wild type. Not altered by genetic modification or breeding.

Summary

Machine learning is the study of algorithms that can learn from data. Recently, the successful application of machine learning to biological problems has demonstrated effective integration of several large biological data sets to yield biological conclusions. However, technical or biological constraints often prevent the production of sufficient relevant data, and out-of-the-box application of machine learning tools can fail to learn relevant data features. Prior knowledge about the underlying data structure can substantially improve the performance of machine learning algorithms.

This thesis describes the effects of biologically informed machine learning techniques on the ability to learn models for four fundamental biological problems from data: gene regulatory networks, sequence motifs, translational efficiency and RNA folding.

Gene regulatory networks enable tightly regulated adaptive gene expression and are comprised of thousands of molecular interactions. In chapter 1, it is shown that a biologically informed noise cutoff as well as a novel gene clustering method substantially improve inference of gene regulatory networks from large-scale data sets. Chapter 2 describes how a novel convolutional neural network architecture that utilizes all circularly shifted variants of the same filter enables robust inference of transcription factor binding sites from very small sample sizes and/or long sequences with short motifs. Messenger RNAs that code for the same protein but are composed of different codons are translated at different rates, a phenomenon termed translational efficiency. The roles of different sequence features are explored on the basis of several large-scale data sets, and the generalization ability of the trained models is examined in chapter 3. RNAs assume specific secondary structures that are determined by their sequences, a process called RNA folding. In chapter 4, it is shown that enumerating intramolecular hybridization states helps to learn secondary structure, however the applied neural network does not scale with sequence length.

Overall, the presented work provides evidence that it can be insufficient to address complex biological problems with large amounts of data and computational power alone. Progress requires data-efficient machine learning algorithms that are tailored to the problem of interest and promote learning of the relevant data features.

Zusammenfassung

Die Untersuchung von Algorithmen, die von Daten lernen können, wird als Maschinelles Lernen bezeichnet. Kürzlich wurde demonstriert, dass Maschinelles Lernen die effektive Integration großer biologischer Datensätze und damit die Analyse komplexer biologische Systeme ermöglicht. Allerdings verhindern sowohl biologische als auch technische Einschränkungen oft die Produktion von genügend relevanten Daten, und die "out-of-the-box" Anwendung von Maschinellem Lernen kann nicht immer relevante Dateneigenschaften erfassen. Allerdings kann Vorwissen über die zugrundeliegenden Datenstrukturen die Leistung von Algorithmen stark verbessern.

Diese Doktorarbeit beschreibt den Effekt von biologischer Vorinformation auf die Fähigkeit von Algorithmen, Modelle für vier fundamentale biologische Systeme zu lernen: Genregulatorische Netzwerke, Sequenz-Muster, translationelle Effizienz und RNA-Faltung.

Genregulatorische Netzwerke ermöglichen die genaue Steuerung und Anpassung von Genexpression und beruhen auf tausenden molekularer Interaktionen. In Kapitel 1 wird gezeigt, wie sowohl eine biologisch inspirierte Obergrenze für Messrauschen, als auch eine neuartige Methode zum Zusammenfassen von Genen die Inferenz von Netzwerken von großen Datensätzen signifikant verbessert. Kapitel 2 beschreibt ein neuartiges künstliches neuronales Netzwerk, welches alle zirkulär verschobenen Varianten des gleichen Filters benutzt. Damit wird die robuste Inferenz von Transkriptionsfaktorbindestellen selbst bei sehr kleinen Datensätzen und/oder langen Sequenzen ermöglicht. Messenger RNAs, die für das gleiche Protein kodieren aber aus verschiedenen Kodons bestehen, werden mit verschiedener Effizienz translatiert. Dieses Phänomen wird als translationelle Effizienz bezeichnet. In Kapitel 3 wird, basierend auf mehreren großen Datensätzen, die Rolle von verschiedenen Sequenz-Eigenschaften bezüglich ihres Effektes auf translationelle Effizienz untersucht. Außerdem wurde die Generalierungsfähigkeit der trainierten Modelle untersucht. RNA-Faltung bezeichnet den Prozess, bei dem sich RNAs in spezifische Sekundärstrukturen zusammenfalten, die durch ihre Sequenzen determiniert sind. In Kapitel 4 wird gezeigt, dass das Auflisten von intramolekularen Hybdridisierungszuständen dabei hilft, die Regeln für RNA-Faltung zu lernen.

Insgesamt zeigt die vorliegende Arbeit Belege dafür auf, dass es oftmals nicht ausreicht, komplexe biologische Systeme ausschließlich mit großen Datenmengen und Rechenkraft zu untersuchen. Vielmehr bedarf es Daten-effizienter Algorithmen, die maßgeschneidert für zugrundeliegende Probleme sind, sodass das Lernen von relevanten Dateneigenschaften ermöglicht wird.

0 Introduction

0.1 General introduction

All living organisms maintain ordered structures by utilizing energy gradients [1,2]. Traits that convey better* use of these gradients are beneficial and likely to be selected by natural selection [3]. For example, the ability to produce the protein and enzyme lactase during adulthood quickly spread in human populations alongside the domestication of cows, as it allowed better digestion of lactose (milk sugar) [4,5]. The information about such traits is encoded in genes, regions of long DNA molecules. DNA consists of four distinct molecular components, nucleotides, whose arrangement in a sequence enables a molecular information storage. Gene expression refers to the decoding of genetic information. This is a two-step process for protein production. In the first step called transcription, molecular copies of genes called messenger RNAs (mRNAs) are produced. In the second step, translation, these mRNAs then serve as templates for the production of proteins†. Translation itself has multiple phases, including the initiation phase, in which the molecular machinery for translation is assembled, and the elongation phase, in which amino acids are polymerized into proteins. Both phases are affected by the 3-dimensional structure of mRNAs [6–8] (chapter 4 focuses on RNA structure). Moreover, the mRNA sequence affects the number of proteins that can be produced from one mRNA template, a phenomenon termed translational efficiency (chapter 3 focuses on translation efficiency).

Adaptiveness has emerged as a strategy for optimal utilization of energy gradients in changing environments [11]. A common bacterium (*E.coli*), for example, can utilize lactose as an energy source by producing the enzyme β-Galactosidase [12]. Production of this enzyme, however, requires energy itself [13]. Consequently, this bacterium increases β-Galactosidase production when lactose becomes available [12]. This regulated production of β-Galactosidase is an example of adaptive gene expression. Adaptive gene expression enables production of RNA and proteins based on environmental or intracellular states, such as the availability of certain nutrients. This requires information processing, which is realized through complex molecular networks [14,15]. Gene regulatory networks are comprised of genes encoding for molecular factors that can affect the expression of specific target genes [12,16] (chapter 1 focuses on revealing gene regulatory network structure). The specificities of these molecular interactions are commonly enabled by unique patterns in both DNA and protein sequences [17] (chapter 2 is on learning sequence motifs).

Mutations are spontaneous changes in the nucleic acid sequence of DNA. Natural selection tends to

*There is no universal definition of what "better" utilization exactly means as it depends on the context. For example, in a fermenter (a well-defined environment) with two strains of bacteria, the strain that accumulates biomass more quickly will outnumber the other strain [9]. In this case, "better" solely means biomass production. On the other hand, parasites that grow or proliferate too rapidly might kill their hosts, resulting in their own demise. In this case, "better" means maintenance of the energy gradient (host), which ultimately increases biomass over the long run because more hosts can be infected [10].

†The terms transcription and translation will be used frequently throughout this thesis.

prevent mutations that negatively affect traits from spreading within a population or species. This process results in conserved genes.

High-throughput technologies have enabled the measurement of thousands of biological variables at once, including cellular mRNA or protein concentrations and their sequences. But due to the large number and sequence lengths of these molecules, detecting variants that are responsible for diseases, for example, is difficult and resembles "searching for the needle in the haystack".

Mathematical modeling is the process of formulating abstract descriptions of systems that allow quantitative predictions. One type of model can be based solely on mathematical functions, in which variables and parameters are related to the function value by mathematical operations. For example, the formula $F = mg$ relates the variables mass (m) and force (F) with parameter g, the gravitational acceleration constant. In contrast to variables whose values can change depending on the context, parameters remain fixed once they have been inferred from data. This inference process is referred to as training. The resulting model can then be used to make predictions for the gravitational force F (output) exerted on, for example, an apple of mass m (input) hanging from a tree.

Learning is the ability to infer general principles from data and generalizing them to similarly structured problems [18]. Machine learning is the study of algorithms that enable machines to learn from data [19], a requirement for artificial intelligence. These algorithms work by training mathematical models on data such that accurate predictions on new but similar data sets can be made.

Humans are capable of finding general principles, even abstract ones, from few examples. For example, when toddlers learn to speak, they commonly overgeneralize the past tense of irregular verbs, saying "eated" instead of "ate" [20]. It has been hypothesized that human generalization ability is enabled by a certain predefined neural architecture [21]. This is also supported by the existence of a special neural architecture in the eye's retina that helps to identify relevant features such as object edges in the visual input [22]. In this case, the visual input can be considered as data with a specific structure consisting of data features such as objects.

Specific data structures also occur in machine learning problems. Common face recognition algorithms, for example, exploit that faces are locally correlated features: eyes are always close to the nose and lips. Moreover, they are translation invariant: faces are recognizable as faces regardless of their position in the image. A special class of Machine learning algorithms have been developed that can learn problems with this data structure particularly well, and which have achieved superhuman performance at classifying images based on what they show [23,24]. A number of specialized machine learning algorithms have been designed for variety of problems, but they often perform well only on particular data structures. Although great efforts are undertaken to obtain a universal models that can learn any problem and/or transfer knowledge between problem domains [25–28], it is unclear whether a single algorithm can be found that covers the combination of all possible data structures.

It is thus crucial to notice that there are fundamental differences between biological problems and classical machine learning applications (such as image classification). First, machine learning re-

quires sufficient data. While millions of labeled images have been uploaded to the internet, biological organisms only have thousands of unique mRNAs and proteins. This can make it difficult to learn general principles. Second, the quality of the data is different. Faces in images, for instance, are typically not blurred and different than the background, which makes it easier to distinguish the signal (face) from noise (background). Moreover, faces are generally still recognizable even when some pixels are changed. For biological data, however, even single nucleotide changes can have severe effects. For instance, single nucleotide mutations can directly cause diseases[29,30]. Furthermore, the signal is often difficult to distinguish from the background. The main reason is that that measurements are often based on only few molecules, and molecules are often similar to one another.

0.2 Concepts related to machine learning

Model training is an optimization process in which the parameters of a model are adjusted so that the model performs a mapping on a training data set as well as possible. Most machine learning models are trained on data derived from measurements or observations that are subject to noise[31]. Consequently, probabilistic considerations often underlie the development of these models. Depending on whether the goal is to map input to output data (e.g. map a sequence to a trait or class) or whether only input data is supplied (e.g. cluster sequences based on similarities), either supervised or unsupervised machine learning methods are used, respectively.
The objective of **supervised methods** is to find a parameterized function that maps an input to predictions such that some distance measure called loss (or cost or error function) between predictions and desired outputs (targets) is minimized. Particular choices for loss functions are often rooted in the **maximum likelihood** framework. In this framework, a parameterized distribution model is chosen based on prior assumptions about the underlying data. Then, the parameters of the distribution model are adjusted so that the probability of observing a particular data sample (the training data) is maximized.

In **unsupervised methods**, the goal is to optimize some objective function of the input data alone, that is, neither targets nor labels are supplied or predicted. Autoencoders, for example, learn to compress input data into abstract **representations** (points in a typically high dimensional space), based on which they reconstruct the input as well as possible. In order to work well, autoencoders must find the relevant input data features to produce an optimal representation that can be used for reconstructing the input[32,33]. Representations can sometimes be interpreted intuitively or correlated to abstract data features[34]. Another classical domain of unsupervised methods are clustering methods that aim to find patterns in input data which can be used to form distinct clusters of examples.

In Deep Learning, complex functions are approximated by neural networks. Neural networks are composed functions of variables and can be represented as directed graphs. In such a representation, three general types of nodes can be distinguished. The input layer consists of nodes that represent input variables, nodes of hidden layers correspond to temporary results of composed functions and the nodes of the output layer represent the (possibly multidimensional) value of the composed function (figure 1 a). Depending on how nodes between layers are connected, several types of neural network layers can be classified. Two frequently used types of neural network layers are fully connected and convolutional (figure 1 c and d, respectively). In **fully connected layers**, each node receives weighted input from all nodes of the preceding layer. **Convolutional layers** can be thought of as the result of a filter applied to the preceding layer, that is, each node receives weighted input only from nodes of a certain region in the preceding layer. Convolution is a hard-coded restriction on which information of the preceding layer can be used for a node. When two layers are connected by a convolution, the number of involved weights is determined by the size of the filter and generally much lower than for fully connected layers. The layers of neural networks can be stacked onto one

another, resulting in **deep neural networks***. Moreover, interjacent nonlinear functions such as the rectifying linear unit (ReLU) or the softplus function enable weighting input differently depending on its intensity, thus serving as noise filters (figure 1 b).

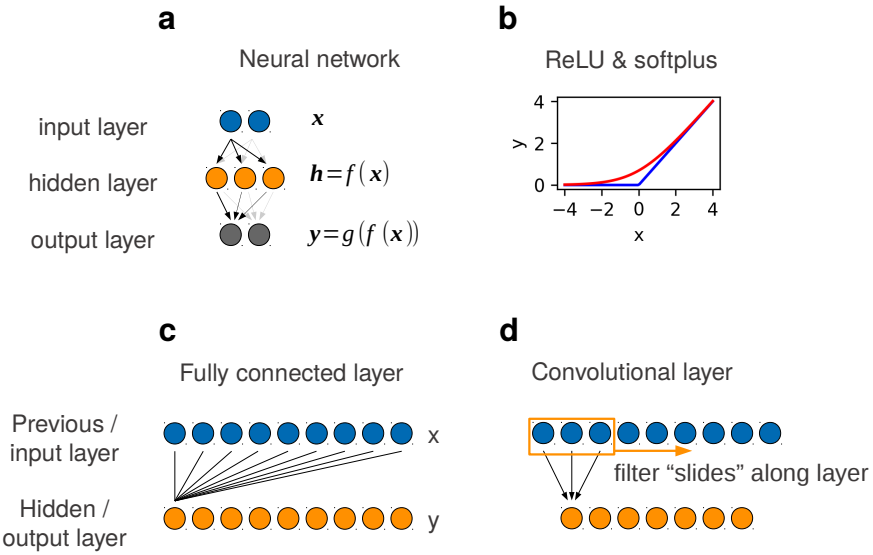


Figure 1: Graphical representation of concepts related to neural networks. **a:** Illustration of a fully connected network and the three basic types of layers: input (x), hidden (h) and output layers (y). The functional relationships are typically vector-valued functions. All nodes of a hidden/output layer receive input from all nodes of the previous/input layer.
b: Blue curve: rectifying linear unit (ReLU), defined by $y = \max(0, x)$. Red curve: softplus function, defined by $y = \log(1 + \exp x)$. Both functions enable that only sufficiently large input values are carried into the subsequent layer.
d: In fully connected layers, each node receives input form all nodes of the preceding layer.
d: A convolutional layer. Each node of the hidden/output layer receives information only from a few adjacent nodes in the previous/input layer.

Machine learning models such as neural networks often contain millions of parameters and non-linear activation functions connecting the layers, making exact and global numerical optimization impossible in practice. A widely used class of local optimization methods for training neural networks is backpropagation, which is based on **gradient descent**[32,39]. These iterative methods require evaluation of the objective function's analytical derivative with respect to the parameters to update parameters according to a small step in the opposite direction of the gradient (if minimization of the objective function is the goal). The step size is also referred to as learning rate. As commonly used functions for connecting layers of neural networks have well-defined derivatives, the chain rule can be used to find the objective function's gradient. Modern Deep Learning software such as TensorFlow[40] enables automatic symbolic differentiation of even highly complex neural networks. Gradients can be computed based on a single example from the training data (stochastic gradient descent), multiple examples from the training data (mini-batch gradient descent) or the entire data set (gradient descent). The number of examples in each batch of mini-batch gradient is referred

*There is no generally accepted convention on the smallest number of layers so that a network can be called “deep”. Some networks that are called “deep” only have 2 or 3 layers[35,36], whereas “very deep” networks have around 20 layers[37], and “extremely deep” architectures have more than 18 layers, with 18 layers being “not overly deep”[38]

to as mini-batch size. For deep models with many convoluted functions (deep neural networks, recurrent networks), the vanishing gradient problem can limit model depth [38,41]. This phenomenon occurs when parameter updates become so small that the model does not improve, even with additional parameters. Furthermore, strongly varying gradients can result in low convergence rates [42]. Several regularization techniques have been developed that solve these problems by small adjustments to the neural network's structures, including batch normalization and residual connections [38,42]. These methods enable sufficiently large gradients at each training step, resulting in better model performance and faster convergence rates.

The main task of machine learning is to find models that generalize well. That is, when trained to solve a certain problem on one data set, models should readily be able to solve the same problem on other but similar data sets. However, especially when there are more parameters than training examples, models tend to just "memorize" the training data and not learn the relevant data features to solve the problem as intended. This is referred to as **overfitting**. Monitoring of overfitting commonly involves splitting up a data set into disjoint training and test (or evaluation) data sets: the elements or examples in both sets come from the same distribution or data generating process, but the sets do not share any elements (and the elements in the test set cannot be too similar to elements in the training data set). If the model's performance on both data sets is not significantly different, it is not overfitting. When there are more unique training examples than parameters, the model does not have the capacity to overfit. However, this does not mean that the model is automatically better able to infer the general principle underlying the data. If more training data is not available, overfitting can be reduced by **regularization**, the incorporation of prior knowledge into the model. Several regularization techniques have been developed that aim to remove unnecessary parameters from the model [43], reducing the effective number of parameters by randomly "dropping out" parameters [44] or letting parameter values shrink towards zero. The latter can be rooted in the Bayesian framework. In this framework, the prior knowledge about model parameters is updated by based on observed data (likelihood). This yields the posterior, the knowledge about the model parameters after including the data. In mathematical terms, the posterior $f(\theta \mid x)$ is the distribution over a set of parameters θ given some data x, which is equal to the θ-parameterized likelihood function $\mathscr{L}(x \mid \theta)$ multiplied by the prior distribution over the parameters $\pi(\theta)$, divided by a special term to guarantee that the result is a properly normalized distribution (equation 1).

$$f(\theta \mid x) = \frac{\mathscr{L}(x \mid \theta)\pi(\theta)}{\int_{\Theta} \mathscr{L}(x \mid \theta')\pi(\theta')d\theta'} \propto \mathscr{L}(x \mid \theta)\pi(\theta) \tag{1}$$

The set of parameters that solves the resulting optimization problem is referred to as **maximum *a posteriori*** parameter estimate as it maximizes the posterior function defined by equation 1. At this solution, the statistical support for parameters is balanced against the prior knowledge about the parameters. Shrinkage priors are distributions over model parameters that can let parameters shrink towards zero depending on their statistical support by the data [31]. The Gaussian distribution ($\mathscr{N}$) is a frequently used shrinkage prior as it allows easy implementation in numerical optimization methods. This and its role in the central limit theorem are often the only reason for choosing it as a

prior distribution because the actual parameter distributions are often unknown. For a appropriate choices of both likelihood and prior distribution, Bayesian methods can strongly reduce the risk of overfitting [45].

Bayesian inference enables, in addition to the calculation of maximum *a posteriori* parameter estimates, also the calculation of confidence intervals over predictions and parameters. As opposed to an optimization process, this requires normalization of the posterior, which in turn requires calculation or approximation of the partition function [31], the integral in the denominator of equation 1. As this function is typically a multidimensional integral of a function of the whole data set, common numerical integration methods are computationally too expensive and Monte Carlo based methods are used instead [31]. Monte Carlo methods draw random parameter samples and then evaluate a function at these samples only. By summing up over all function evaluations, integrals can be approximated. Stochastic gradient Langevin dynamics [45] (SGLD, equation 2) has been proposed as a general-purpose method for approximate Bayesian inference as it allows to transition from optimization (learning) to parameter sampling (approximating confidence intervals) by adjusting the learning rate. That is, the model parameters θ are updated at time step t according to the gradient of both log-prior $\log p(\theta_t)$ and likelihood of the data, $\log p(x_i|\theta_t)$, plus Gaussian noise η_t whose variance depends on ϵ_t, the learning rate at time step t (here, m is the mini-batch size and n is the total number of examples):

$$\begin{aligned} \theta_{t+1} &= \theta_t + \frac{\epsilon_t}{2}\left(\nabla \log p(\theta_t) + \frac{m}{n}\sum_{i=1}^{m} \nabla \log p(x_i|\theta_t)\right) + \eta_t \\ \eta_t &\sim \mathcal{N}(0, \epsilon_t) \end{aligned} \tag{2}$$

Furthermore, the computational cost is only minimally increased compared to stochastic gradient descent. SGLD can be used to approximate maximum *a posteriori* parameter confidence intervals at a specific learning rate and batch size.

Determining by hand whether model parameters are significantly supported by the data (that is, whether their existence or, equivalently, values other than 0 are justified) requires statistical hypothesis testing. This involves modeling the distribution of the parameter under the null hypothesis. When calculation of this statistic (the model parameter) includes evaluating complicated functions, it is often troublesome to derive the distribution or even moments such as the variance analytically. Alternatively, the distribution can be approximated via bootstrapping. **Bootstrapping** is a resampling technique that draws samples (with replacement) from the original data set and then calculates the statistic of interest (s.a. model parameter). When this is done a sufficient number of times, the statistics calculated from the samples follow a distribution that can be used to approximate the actual distribution of the statistic. Then, statistical testing can be either based on the bootstrapped statistics directly, or, with some justified assumptions about the actual underlying distribution, a distribution function can be fitted to the bootstrapped statistics to allow (better) approximation of distribution tails.

Besides the model parameters that can be trained, some so called hyperparameters such as the gradient descent step size (learning rate) or the balancing weight between likelihood and prior functions (the so called regularization coefficient) cannot be trained but must be adjusted otherwise [31]. This also includes the choice of the model likelihood, prior and model functions. **Cross-validation** is a commonly used procedure in which the entire available data set is randomly split into disjoint training and validation data sets. Models are then trained on the training data for different hyperparameters, and the hyperparameter combination yielding the best loss function value on the validation set is taken as the optimal choice.

A variety of complex problems can be solved with neural networks, and a key reason for this is their ability to produce abstract representations of the data [46]. Neural networks can learn to approximate logical operations; for example the exclusive OR (XOR) gate can be modeled with 2 layers [32,47]. Sequential problems such as language translation have been enabled by recurrent neural networks that map sequential inputs to outputs [48]. The sequence information is compressed into a high-dimensional sequence representation, and predictions are made in a consecutive manner rather than in one shot. That is, the elements of the sequence to be predicted are produced one after another, and future predictions depend on already produced predictions. Long Short-term Memory Networks (LSTM) are improved recurrent networks that have enabled mapping between long sequences. These were further improved by a mechanism that enables learning which data features to pay attention to [49]. The "paying attention to" part is realized by assigning context-dependent weights (between 0 and 1) to the layer nodes, and context is determined by the agreement (scalar product) of high-dimensional vectors that represent data features or mappings thereof. This method has enabled the development of neural networks that can solve complex combinatorial tasks, such as sorting numbers [50].

The elements within sequences such as DNA are discrete entities. Neural networks, however, are functions of real numbers. Consequently, sequences must be converted into a compatible format so that neural networks can operate on them (figure 2). This is commonly achieved by one-hot coding (also known as 1-of-K coding [31]), which assigns a binary vector to each possible sequence element: a sequence $\Sigma = \Sigma_1, \ldots, \Sigma_{L_S}$ of length L_S with elements Σ_j coming from an ordered set with limited finite cardinality $\Sigma_i \in D, |D| = N$ can be represented as a $N \times L_S$ one-hot coding matrix S with elements S_{ij} that are equal to 1 if Σ_j is the i-th element in D and 0 otherwise:

Figure 2: Illustration: a nucleic acid sequence represented as one-hot coding.

In classification tasks, performance is commonly represented in the form of an accuracy, that is, the number of correctly classified examples or elements in a sequence. Accuracies, however, can only be interpreted when the number of actual class members is given or classes have equal sizes (this issue is illustrated in figure 3). The area under the receiver operator characteristic curve (AUROC) is a performance metric that enables performance comparisons for binary classification tasks (when there are only 2 classes) when classes have different sizes. An AUROC value of 0.5 indicates random guessing whereas a value of 1.0 indicates perfect prediction performance; however, as with other metrics, the expressiveness of AUROC decreases with the number of underlying classification

examples.

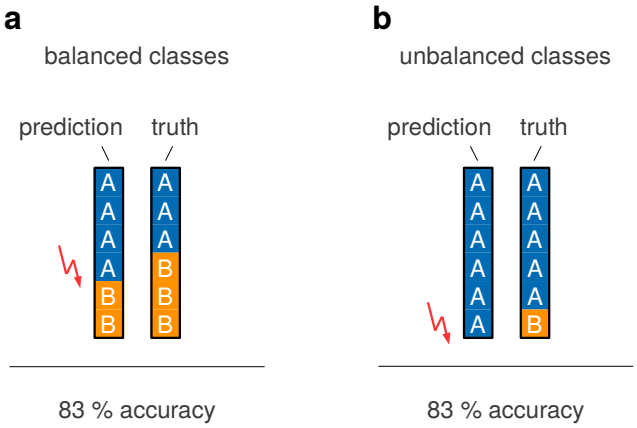


Figure 3: Illustration of why using accuracies as performance metric can be meaningless when no class sizes are provided.
a: in 5 out of 6 cases, the correct classes *A* and *B* were predicted, resulting in an accuracy of 83 %. The true classes were balanced, that is, both classes had equal number of members, making it easy to compare the obtained accuracy to random guessing (which would correspond to an accuracy of 50 %)
b: The true classes have different sizes so that a high accuracy of 83 % was achieved by simply assigning all examples to one class. In this case, it is unclear whether the algorithm has learnt to classify the examples.

0.3 Gene regulatory networks

Gene regulatory networks adjust the abundance of gene products in response to environmental or intracellular states. They are comprised of regulators, either proteins or RNAs, that influence gene expression by affecting transcription or translation. The activity of some of these regulators can change, commonly as a result of structural changes or chemical modifications. These can be induced by environmental changes, binding of certain molecules or binding of other regulators. This essentially makes them molecular sensors or information transmitters. A well-studied class of regulators are transcription factors, which are proteins that affect transcription.
GRNs can be represented as graphs, with nodes representing regulatory components and links denoting interactions.

The relationship between the concentrations of interacting regulatory components and their probability of binding (binding kinetics) is non-linear. Furthermore, multiple factors can influence the expression of a gene, and one regulator can influence multiple genes. These properties enable collections of genes and regulators to execute approximate logical operations or noise-filtering of signals, which are the basis for more complex information processing functions of gene regulatory networks [15].

Mutations or inhibitors of regulators can strongly affect GRN function. In humans, for example, mutations in a certain regulator gene were found to be obligatory for small cell lung cancer [51]. The mutated regulator can lead to altered network behavior, which can result in uninhibited and abnormal cell proliferation, accumulation of further mutations and eventually cancer[52]. Knowledge about the gene regulatory interactions that underlie complex diseases has resulted in several targeted therapy approaches, including monoclonal antibody therapies that elicit specific cellular or immune responses[53] and therapeutic modulation of gene expression via gene therapy[54].

The advent of high-throughput technologies has enabled the simultaneous measurement of thousands of biological molecules, and numerous methods have been developed that aim at integrating such data to produce comprehensive models of the molecular interactions underlying gene regulatory networks. From a practical perspective, modeling approaches fundamentally differ in the regulatory relationships that they represent, the network size and type of data they are compatible with, and their robustness towards noise. Specifically, models differ in whether they can:

i distinguish causal relationships from correlations,

ii distinguish direct from indirect interactions,

iii model quantitative network behavior,

iv model dynamic behavior such as oscillations,

v quantify uncertainty.

Inference of causal interactions (i) is more informative than merely inferring correlations because it

allows prediction of effects. Distinguishing direct from indirect interactions (ii) enables identification of cause/effect relationships between individual network components. The ability of different approaches at this task has been studied before [55]. Although qualitative interaction data such as ChIP-seq* data can help to distinguish direct from indirect interactions that affect transcription, unspecific binding events are common in practice. Quantitative data such as concentrations of regulatory components is a valuable alternative. Modeling of quantitative network behavior (iii) such as link strengths enables quantitative predictions. However, this also requires quantitative data such as concentrations of network components. Models based on systems of differential equations or Gaussian processes can be used to model complex dynamic network behavior such as oscillations (iv), in contrast to models based on linear approximations. Training of such models however requires time-series data and computationally expensive optimization techniques, making them suitable only for small networks. Finally, measuring uncertainty about model parameters (v) can help researchers to estimate the probability of successful interventions or feasibility of further studies.

The nodes in a network are associated with node activities such as mRNA or protein concentrations. Perturbations are events that affect node activities (figure 4), such as gene deletions. The effect of perturbations on node activities is commonly quantified relative to a well-defined reference state, such as a non-affected (e.g. wild type) cell. There are specific and unspecific perturbations. Specific perturbations in gene regulatory networks can be induced by gene deletions so that no more mRNA and protein can be produced, causing the corresponding molecule concentrations to drop to zero. For specific perturbations, the node that originally receives information (a.k.a. perturbation target), is known. Unspecific perturbations can be induced by environmental changes such as heat or lack of nutrients, but the perturbation targets are unknown. Once a node has been affected by a perturbation (either specific or unspecific), the perturbation effects propagate through the network depending on which regulatory targets subsequently affected nodes have. These effects can be assessed by either measuring the set of all mRNAs within a cell, the so called transcriptome, or by measuring the set of all proteins in a cell, referred to as proteome.

Inference of the causal relationships between regulatory components (or equivalently, links between nodes) requires measuring their response towards selective perturbation of their cellular concentrations (node activities). In order to infer all possible regulatory interactions in a network, all network components need to be perturbed independently.

In practice, there are two problems related to the induction of perturbations and measurement of their effects. First, selective perturbations induced by gene deletions is not possible for all genes as some genetic alterations are lethal [56], and creating such constructs or constructs for inducible gene deletion [57] are expensive. This leads to the situation in which the number of independent perturbation experiments is smaller than the number of nodes or model parameters [56]. Second, genomes contain thousands of genes†, which must all be regarded as potential regulatory components.

The data produced with high-throughput technologies such as microarrays and RNA sequencing is

*See section 0.4 for a brief description of the ChIP-seq method.

†*Mycoplasma genitalium*, a bacterial pathogen: $\sim$ 500 protein coding genes [58], *Pinus taeda*, a conifer: estimated at $\sim 50,000$ protein coding genes [59], human: $20,338$ protein-coding and $19,201$ RNA-coding genes [60]

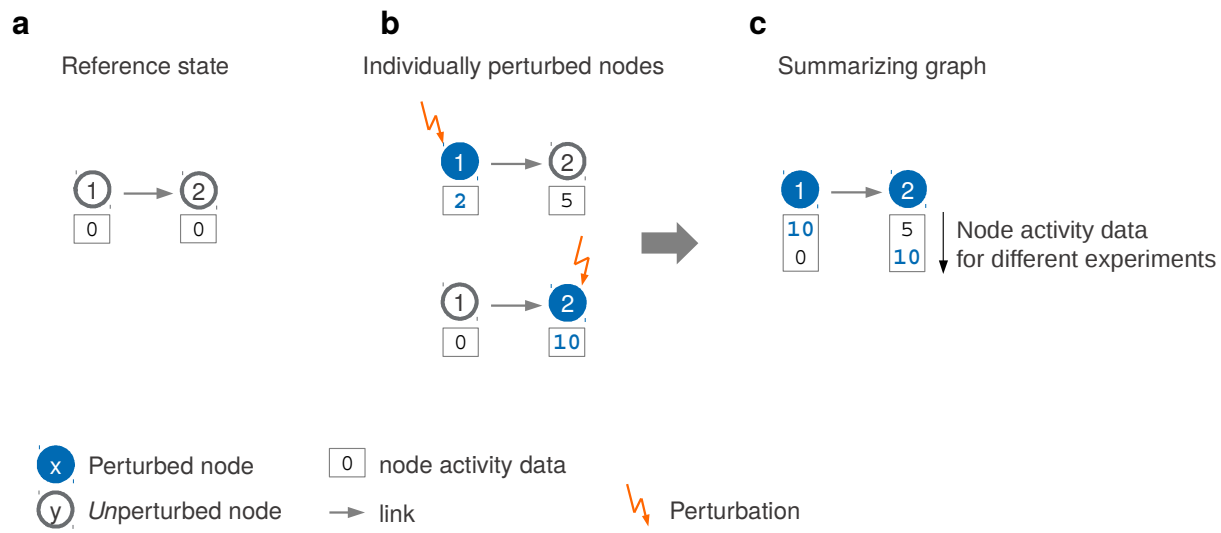


Figure 4: Illustration how node perturbations can be represented graphically.
a: Both nodes in this graph are unperturbed, and the corresponding node activities are zero (per definition). This state is used as a reference state.
b: When node 1 is perturbed, it's node activity changes to an exemplary value of 10. Since it is causally linked to node 2, the node activity of node 2 also changes, here to a value of 5. When Node 2 is perturbed, however, node 1 is not affected because there is no link pointing from node 2 to node 1.
c: This graph represents a comprehensive summary of **b**, that is, it contains both perturbation experiments shown in **b**. Both nodes in this graph are perturbed independently so that there are multiple independent experiments associated with each node.

subject to substantial biological and/or technical noise [61,62]. Discriminating signal from noise is a profound problem of network inference. The reason is that, in a network model with P nodes, the number of possible links is on the order of P^2 (curse of dimensionality), but gene regulatory networks are typically sparse, meaning that there are much less than P^2 interactions between the regulatory components. Thus the chance of inferring false positive interactions increases quadratically with network size. Moreover, increasing the number of experimental replicates to average out noise is often not affordable. A general solution to both problems is to increase the N/P ratio by reducing the number of model parameters (N is the overall number of experiments). Regularization techniques reduce the effective number of parameters. Lasso regularization, for example, subjects models to a sparsity constraint that relates the existence of links to their statistical support [43]. Another alternative is to treat mRNAs and proteins as one entity: since mRNA levels are correlated with protein levels [63], they can be used as a proxy for protein levels. This can substantially reduce the number of model parameters, however the resulting model must be interpreted accordingly.

Based on benchmarks, it was shown that a combination of several network inference methods outperformed individual methods [55]. This implies that the individual methods were biased and designed to work well in particular settings only. For novel or understudied organisms, however, there is no specific prior knowledge available that could be used to improve network inference. Network inference in novel organisms thus requires a method that works well for any organism.

On a molecular level, the association between transcription factors and DNA is subject to stochasticity[64,65] (the same phenomenon is the cause for Brownian motion). The extent of this stochasticity is especially pronounced when the components occur at low concentrations and association strengths are weak. This results in noisy gene expression, and some genes are noisier than others[66]. It might be tempting to assume that noisier genes have less crucial functions, but many extremely noisy genes are highly conserved[66]. In principle, averaging over cells should average out such stochastic effects, however batch effects can remain and require averaging over biological replicates or experiments carried out on different days or by different scientific staff[67].

Due to the large number of genes that organisms have, generating features that summarize certain similarities or differences between genes is necessary for a comprehensive understanding of gene function. A number of methods have been developed that can be used to group genes based on certain characteristics. k-means clustering[68], for instance, aims at categorizing examples such that k clusters of examples are formed, with cluster affiliation being determined by the cluster to which an example is closest[31]. This unsupervised method only takes the data and the number of clusters, k, as inputs, and returns cluster affiliation and cluster centers.

0.4 Biological sequence motifs

Cellular processes are highly organized and rely on specific molecular interactions. For example, gene regulatory networks only function because transcription factors affect specific genes; eukaryotic mRNAs are only spliced* at specific sites; and only certain mRNAs are designated for long-distance transport in cells. The specificities between interacting molecules are commonly determined by the compatibility of their 3-dimensional structures, and, associated with that, van der Waals forces and electrostatic fields that govern the strengths of polar interactions. Since biological macromolecules such as DNA, RNA and proteins are polymers of simpler molecules (nucleotides and amino acids), the three-dimensional structure also often (but not always [69,70]) manifests as motifs in their sequence. Motifs can thus be regarded as local, translation-invariant patterns in sequences. Biological molecules such as transcription factors frequently affect multiple targets. Furthermore, highly similar 3-dimensional structures can be obtained with different sequence motifs, meaning that structure is generally more conserved than sequence [71]. As a consequence, there is variation among biological sequence motifs. A probabilistic interpretation of this is that biologically realized motifs are samples drawn from a motif distribution.

Mutations in motifs can affect cell function and cause diseases, some of which have substantial economic effects. Spinal muscular atrophy [30], for example, is a disease that contributes to infant mortality and is caused by incorrect mRNA splicing due to a mutation [30]. A mutation in the tumor suppressor gene TP53 prevents the protein from binding to its target motifs, promoting accumulation of further mutations that may lead to cancer [72]. In maize plants, the virulence of an important pathogen is altered by motif mutations leading to impaired mRNA binding [73].

A variety of methods for inferring motifs from high-throughput data have been developed [74] (details in chapter 2). Representations of sequence motifs are commonly based on the spatial relative nucleotide frequencies within the motif (figure 5), such as the frequently-used position weight matrix [75]. Motif inference methods are subject to a trade-off between performance and motif interpretability. On the one hand, the inferred models should be as easy to interpret as position weight matrices. On the other hand, models should allow accurate predictions, for which a sufficiently complex model of the motif distribution is required, preventing easy interpretation. For example, mutually exclusive nucleotides cannot be represented with one nucleotide frequency matrix (figure 5).

The Encyclopedia of DNA Elements (ENCODE) consortium aims at revealing the function of all human genes [76]. A part of this endeavor includes characterization of transcription factor binding sites, for which numerous transcription factor ChIP-seq experiments have been conducted and submitted to a database. In short, the ENCODE ChIP-seq protocol includes chemical crosslinking of DNA with currently bound proteins, followed by antibody-based extraction of specific transcription factors including the DNA fragments they are linked to, and then sequencing of these DNA fragments. The number of reads† that align to a specific region in a reference genome can then can be used as a proxy for the interaction strength between a transcription factors and the respective DNA binding

*Splicing refers to the regulated excision of certain mRNA regions in eukaryotic organisms including yeast and humans, but not bacteria.

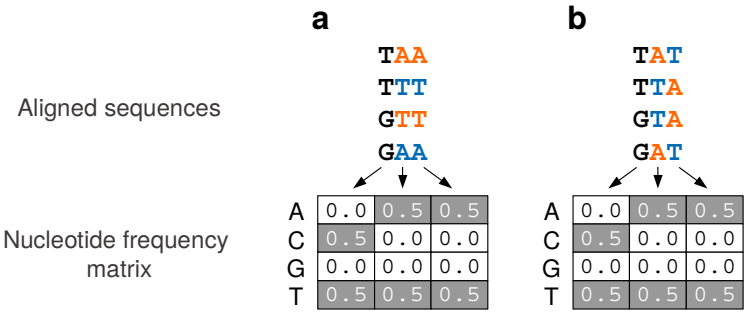


Figure 5: Illustration of nucleotide frequency matrices. By counting the number of times that a specific nucleotide occurs at each position in the sequence alignment, relative frequencies can be calculated and represented as a nucleotide frequency matrix. However, different sequences can result in the same frequency matrices: while the A's and T's at the second and third position in **a** are correlated, they are inversely correlated in **b**. This illustrates the limitations of nucleotide frequency matrices.

sites.

The previous state-of-the-art method for motif inference, DeepBind, was based on a convolutional neural network[35]. This network outperformed established methods for motif inference at learning and subsequently predicting whether sequences from ENCODE ChIP-seq experiments had high or low read counts. DeepBind employed multiple convolutional filters that were typically longer than the motif. However, it was reported that DeepBind did not yield robust predictions[77]. This study also revealed that deeper architectures did not improve prediction accuracies.

[†]Reads are fragments of nucleic acids (either DNA or RNA) that are measured during sequencing.

0.5 RNA secondary structure

RNA molecules are polymers of alternating sugar (ribose) and phosphate molecules, with each sugar attached to either one of four nucleobases [16]. Each nucleobase-sugar-phosphate subunit is referred to as a nucleotide. The nucleotides can bind to other nucleotides within the same RNA molecule, but the interactions between the four different nucleotides vary in the strengths of the hydrogen bonds that are formed between them. The most strongly interacting nucleotides are basepairs between Adenine and Uracil as well as Guanine and Cytosine, but other interactions such as between Guanine and Uracil are possible. In combination with the negatively charged phosphate groups that repel each other and partially hydrophobic nucleobases that tend to stack and displace water [78], only certain binding configurations are energetically favorable. In aqueous solution under biologically relevant conditions, RNA molecules thus tend to assume 3-dimensional structures that are determined by their nucleotide sequences, a process referred to as RNA folding. This property both enables and affects a wide range of biological processes. Gene expression, for example, can be regulated by RNAs that act as direct or indirect molecular sensors of other molecules or temperature [79–81]. The structure of RNAs can also confer catalytic properties: a certain RNA was shown to cleave itself, and it has been shown that even the catalytic activity of ribosomes is due to the involved ribosomal RNAs [82,83]. Furthermore, a variety of highly relevant viruses are RNA-based and possess genome-scale ordered RNA structure [84]. These findings suggest that proper understanding of RNA biology requires understanding of RNA folding.

Although a variety of molecular interactions within RNA molecules are possible [78], the 3-dimensional RNA structure is commonly described in a simplifying way. In this description, nucleotides interact with only one other nucleotide via mutually exclusive interactions called base pairs [85]. The set of all these interactions forms a specific but not unique secondary structure for each RNA molecule. These structures can be represented graphically, and several recurring structural patterns can be classified, most notably hairpins (figure 6 a). A commonly used one-dimensional representation of secondary structure is the dot-bracket notation [86]. The standard dot-bracket notation however cannot represent all secondary structures such as pseudoknots [87] (figure 6 b).

Since RNA molecules are subject to thermal fluctuations, the microscopic interactions between nucleotides are affected as well, and there is a chance that the hydrogen bonds between nucleotides break and other bonds between other nucleotides are formed. Thus, usually not one but several structural conformations are assumed within a certain time span: structures that are energetically more favorable (have lower Gibbs free energy) than others are assumed more often [88,89]. The minimum free energy (MFE) structure is the conformation that is assumed most often.

De novo RNA secondary structure prediction is a combinatorial problem that aims at finding the MFE structure for a certain RNA sequence. Under the simplifying assumption that no pseudo-knotted structures occur, the RNA folding problem can be solved recursively. This involves finding optimal structures of subsequences first and then selecting and combining substructures such that the overall number of energetically favorable base pairs is maximized (or equivalently, the folding energy is

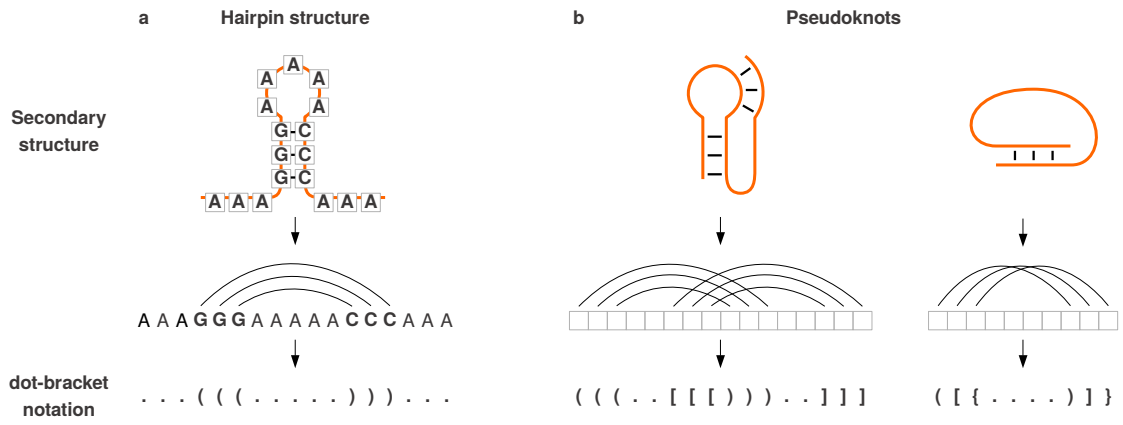


Figure 6: Illustration of graphical representations of RNA folding.
a: The standard dot-bracket notation consists of three symbols (dot, open bracket, closed bracket) and can represent non-overlapping structural elements such as hairpins.
b: An advanced dot-bracket structure with multiple bracket symbols is required to represent pseudo-knotted structures in which multiple, overlapping structural elements occur. The secondary structure is represented here without nucleotides for a better overview.

minimized) [90,91]. This global optimization algorithm belongs to the class of dynamic programming algorithms. These methods, however, rely on models of the interactions between nucleotides with empirically found parameters [85]. Interactions between nucleotides are also affected by the directly adjacent nucleotides in a sequence. This results in energy models that rely on nearest neighbor interaction or "dinucleotide" parameters [85]. Despite these efforts, the RNA secondary structure predictions made by a commonly used program that relies on dinucleotide parameters are far from perfect [92].

Recently, neural networks have been successfully employed to solve complex tasks by learning from data instead of relying on predefined models of the underlying rules governing a problem. For example, neural networks have outperformed conventional algorithms for language translation that were based on statistics [93]. Furthermore, neural networks were recently shown to solve combinatorial tasks such as the traveling salesman problem *, which are classically solved by dynamic programming [94,95].

*In this task, the goal is to visit a number of cities in such a way that the overall traveling distance is minimized, and the traveler returns to the starting city.

0.6 Translational efficiency

Translation is the step of gene expression in which proteins are produced based on the information stored in messenger RNAs (mRNAs). This process involves ribosomes - molecular complexes that move along mRNAs and catalyze the ordered polymerization* of specific amino acids into proteins. Ribosomes consist of a small and a large subunit, both of which contain unique ribosomal RNAs (rRNAs) and proteins. Specific and sequential amino acid polymerization is enabled by transfer RNAs (tRNAs) that assign each amino acid to specific, consecutive nucleotide triplets on the mRNA called codons. As there are four different nucleotides, there are $4^3 = 64$ different codons. Organisms typically have only 20 different amino acids, which results in a redundancy of the genetic code: some of the 64 codons code for the same amino acids; they are synonymous. This means that different mRNA sequences can code for the same protein. mRNA molecules have two poles: the 5' and 3' ends. Ribosomes move along mRNAs from the 5' to the 3' end. Bacterial mRNAs typically consist of three distinct regions: an untranslated region at the 5' end which contains a conserved binding site for the ribosome; a region that contains either one (monocistronic) or multiple (polycistronic) coding sequences which is/are fully translated into protein; and another untranslated region at the 3' end that typically contains an RNA hairpin structure. Each coding sequence starts with a start codon and ends with a stop codon. In polycistronic mRNAs, the ribosome binding site can be located in the preceding coding region. It was found in the 1980s that proteins with core cellular function such as bacterial membrane proteins or ribosomal proteins are the most highly expressed [96].

Different mRNAs that code for the same protein result in different amounts of protein: the mRNAs have different translational efficiencies. Several cellular and mRNA sequence features were found to be correlated with translational efficiency, and a variety of mechanisms have been proposed to explain these correlations [97]. In a large-scale study in which more than $14,000$ artificial constructs were screened, it was found that mRNA folding energy was a main predictor of translational efficiency [98]. It was hypothesized that sequences with high† folding energy are less likely to be in a folded state and hence more accessible to ribosomes, promoting translation initiation. This was supported by the observation that codons in which the last nucleotide (wobble base) is C or G appear less frequently at the beginning of coding sequences (intragenic spatial codon usage bias) and by the fact that base pairing between C and G is stronger than between other nucleotides. That is, strongly binding nucleotides appear less frequently around the translation start site.

Codon usage also varies between genes within a genome (the set of all genes of an organism). It was found that mRNAs whose codons corresponded to the more abundant tRNAs were more highly expressed in *E. coli* [96], resulting in several measures such as the codon adaptation index to describe these effects [99,100]. In another study, evidence was found which suggested that codons are chosen such that ribosome translation speed is regulated, which may help to increase the number of ribosomes that can simultaneously translate a messenger RNA without stalling [101]. However other studies in *E. coli* did not find evidence supporting this hypothesis [98,102].

*attachment

†Folding energies are reported as negative values per convention, with lower (more negative) values corresponding to stronger folding.

Optimizing the translational efficiency of genes is of biotechnological relevance as more protein can be produced with less RNA. Consequently, a number of tools for translational optimization of mRNAs have been developed [103–108]. But despite these studies and efforts, it remains unclear whether all explanatory factors have been elucidated or if a precise functional relationship between nucleotide sequence and translational efficiency can be found given the currently available data. In particular, the above-mentioned sequence features only explained 54 % of the variation in a recent large-scale data set [98].

Aims

The precise prediction of phenotypes from genotype data is the key goal of the post-genomic era, but biological or technical constraints often limit acquisition of relevant data for certain biological problems. Predicting phenotypes from genotype data thus requires methods that can learn to extract the relevant information from few training examples and still yield accurate predictions. However, little is known about the data requirements models and vice versa.

The aim of this thesis was to improve the understanding of the relationship between the data structure of biological problems and the ability of machine learning methods to efficiently learn to extract relevant data features. This was pursued by investigating four fundamental biological problems by:

i analyzing requirements and limits of gene regulatory network inference from large-scale gene deletion studies,

ii investigating the reasons for previously unexplained low performance of convolutional neural networks in motif inference problems when small filters are used,

iii exploring which sequence features of mRNAs enable accurate prediction of their translational efficiencies and which method combines these features optimally, and by

iv determining if learning the secondary structure of biopolymers, with RNA folding as an example, improves with hard-wired prior knowledge about underlying biophysical mechanisms.

1 Inference of gene regulatory networks

1.1 Summary of the obtained results

Network inference aims at finding a model that describes observed node activity data based on causal interactions between nodes. A major problem is that infererence of links is hindered by the curse of dimensionality: the number of possible links between P nodes scales with P^2, meaning that there is much less statistical support for the existence of some links as network size increases. Methods that investigate this statistical support for link existence[43] are computationally expensive. Moreover, the results of a study[55] in which a consensus inference method outperformed individual methods suggested that common inference methods are biased. This study also found that most methods cannot correctly infer simple network motifs.

It was found that common inference method fail to correctly infer simple network motifs, and a bias toward measurement noise was identified as the reason. Investigation of the inference of simple motifs identified a bias towards measurement noise as the main reason why common inference methods cannot infer simple network motifs. Based on this finding, a novel inference method called Partial Response Coefficients was developed. It is unbiased to measurement noise and allowes correct inference of simple network motifs and thus improved network inference in general (figure 7 a).

Gene expression is noisy on a single cell level[64,65]. On average, however, a species must make the right decision in a given situation, meaning that noise should not be interpreted as signal. This means that, when averaging over a large collection of cells, node activity data that is indistinguishable from noise is unlikely to contain information about causal gene regulatory interactions. Based on this rationale, a novel method was developed which only infers links within subnetworks defined by the following condition: nodes within the subnetwork have activities that are significantly above or below a certain noise threshold (figure 7 b). With this method, inference of large networks from synthetic data improved substantially to the level of the state-of-the-art method, but at an in practice lower computational cost. When nodes with similar node activity data were grouped into clusters and treated as single nodes, network inference improved further since the number of indistinguishable solutions was reduced (figure 7 c).

A combination of the noise-unbiased, subnetwork and clustering methods enabled correct inference of the yeast galactose utilization network from a large-scale gene deletion transcriptome data set. Furthermore, this method scored at least second in a network inference benchmark.

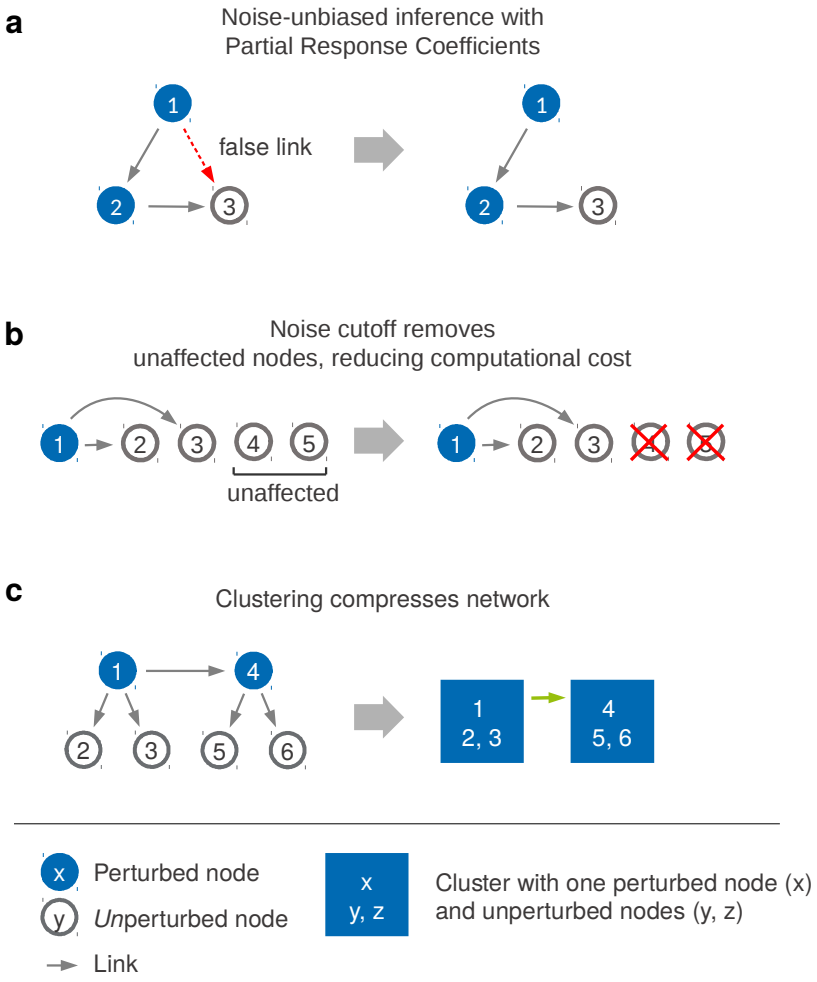


Figure 7: Network inference is improved by several algorithmic steps.
a: Common inference methods detect false links in simple network motifs, which is not the case for Partial Response Coefficients, a noise-unbiased inference method described in the following publication.
b: Nodes that are not significantly affected by a perturbation, either directly or indirectly, do not contain information about regulatory interactions, and their removal improves network inference by reducing the computational cost. The remaining nodes constitute a subnetwork. Each perturbation can result in a unique subnetwork, depending on which nodes the perturbation target is linked to (not shown here).
c: Node clustering improves network inference because equivalent routes of information flow are omitted; this is described in more detail in the appendix to the following publication (section 1.3).

1.2 Publication & supplementary note: Experimental noise cutoff boosts inferability of transcriptional networks in large-scale gene-deletion studies

Authors: C. F. Blum, N. Heramvand, A. S. Khonsari & M. Kollmann
This article was first published in *Nature Communications* **9**(1), 2018.

Contribution of Christopher Blum:

- developed (with M. K.) and assessed the clustering and subnetwork methods
- analyzed the transcriptome data
- analyzed the DREAM challenge data
- wrote Supplementary Note 3
- designed figures 3 and 4

Supplementary material is accessible via the publisher's website:
`https://www.nature.com/articles/s41467-017-02489-x`

nature COMMUNICATIONS

ARTICLE

DOI: 10.1038/s41467-017-02489-x OPEN

Experimental noise cutoff boosts inferability of transcriptional networks in large-scale gene-deletion studies

C.F. Blum[1,2], N. Heramvand[1,2], A.S. Khonsari[1] & M. Kollmann[1]

Generating a comprehensive map of molecular interactions in living cells is difficult and great efforts are undertaken to infer molecular interactions from large-scale perturbation experiments. Here, we develop the analytical and numerical tools to quantify the fundamental limits for inferring transcriptional networks from gene knockout screens and introduce a network inference method that is unbiased with respect to measurement noise and scalable to large network sizes. We show that network asymmetry, knockout coverage and measurement noise are central determinants that limit prediction accuracy, whereas the knowledge about gene-specific variability among biological replicates can be used to eliminate noise-sensitive nodes and thereby boost the performance of network inference algorithms.

[1] Institute for Mathematical Modeling of Biological Systems, Heinrich-Heine University of Düsseldorf, Universitätsstraße 1, 40225 Düsseldorf, Germany. [2] Max Planck Institute for Plant Breeding Research, Carl-von-Linné-Weg 10, 50829 Köln, Germany. C.F. Blum and N. Heramvand contributed equally to this work. Correspondence and requests for materials should be addressed to M.K. (email: markus.kollmann@hhu.de)

The functionality of a living cell is determined by the interplay of multiple molecular components that interact with each other. Generating a global map of these molecular interactions is an essential step to advance our understanding of the molecular mechanisms behind disease, development and the reprogramming of organisms for biotechnological applications[1]. The current advances in gene-editing methods[2] have scaled up the size of genome-wide single and double knockout libraries, ranging from microbes[3, 4] to higher eukaryotes[5] and open up a much more informative data source than inferring gene-regulatory networks from unspecific perturbations, such as stress or changes in growth conditions[6]. However, the detection of direct interactions between two genes from association measures–for example, the covariance between transcript levels–remains a highly non-trivial task, given the significant variation among biological replicates, the frequent case where the number of parameters exceeds the number of independent data points, and the high dimensionality of the inference problem. In addition, direct interactions inferred from transcriptome data typically oversimplify the molecular complexity behind gene regulation, which frequently involves protein–protein interactions and modifications on protein or DNA level. Consequently, gene interaction networks inferred from transcriptome studies should in general not be interpreted as or compared with gene-regulatory networks. In this work we first investigate the causes that affect network inferability by introducing a simple network inferability measure and subsequently use the gained insight to design an unbiased, scalable network inference algorithm.

Results

Network inferability. The existence of a direct interaction between gene A as source of regulation (source node) and gene B as target of regulation (target node) can be detected if a significant part of the transcriptional activity of B can be explained by the transcriptional activity of A but not by the transcriptonal activities of the remaining genes in the network. Thus, a necessary condition for identifiability or inferability of links is the knowledge about the information that can be transmitted by alternative routes in the network, which can be obtained by targeted, external perturbations of node activities[7]. As most gene perturbation screens are incomplete—for example, owing to the fact that essential genes cannot be knocked out—we have in general the situation that a significant amount of interactions within an N-gene network are non-inferable, regardless of the amount of experimental replicates and the strength of perturbations.

Limits of network inferability. To estimate the upper bound of links that can be inferred from knockout screens, we consider a directed but not necessarily acyclic network of N nodes, with node activities as observables and a predefined subset of nodes that are perturbed independently by external forces. The perturbed nodes are randomly distributed within the network and we denote by q the fraction of nodes that are perturbed. We assume that an arbitrarily large set of perturbation experiments can be generated, with the freedom to tune the perturbation strength for each node independently. We further assume that other perturbative sources and measurement noise are absent. Calculation of the expected fraction of inferable links, $F(q)$, can be carried out by a simple counting procedure (Figs 1a and 2a), assuming that links can be represented by noiseless, linear functions with non-zero slope. Under these conditions, a directed link between source and target node is inferable–or equivalently its link strength is identifiable–if it is not possible to fully reconstruct the activity state of the source node from the node activities of the remaining network. Consequently, a link is inferable if a part of the variation of the target node can be only explained by the source node, given that a link between them exists, and implies non-zero partial correlation between source and target node. To allow detection of arbitrarily small partial correlations, we make sure that there exists a finite fraction of experiments for each target node, where the target node is not perturbed (Online Methods and Supplementary Note 1). If, for example, only one node in the network is perturbed that targets multiple other nodes, its node activity can be fully reconstructed by any of its targets, resulting in zero partial correlation coefficients, which implies that none of the directed links can be inferred (Fig. 1a, right network). In contrast, if two out of three nodes are perturbed, all links targeting the unperturbed node are inferable (Fig. 1a, left network). In addition, nodes that have been identified as targets of the current target node can be removed prior to inference. This is because an existing link from the actual target node excludes them from

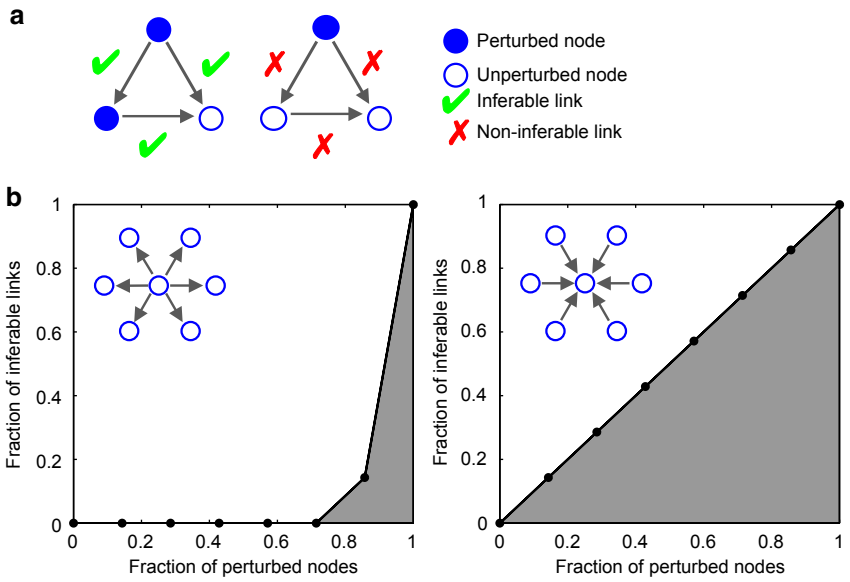


Fig. 1 Illustrative example of network inferability. **a** Left network: fully inferable network; Right network: non-inferable network. **b** Fraction of inferable links versus fraction of perturbed nodes in the network. Left panel: hub of outgoing nodes. Right panel: hub of incoming nodes

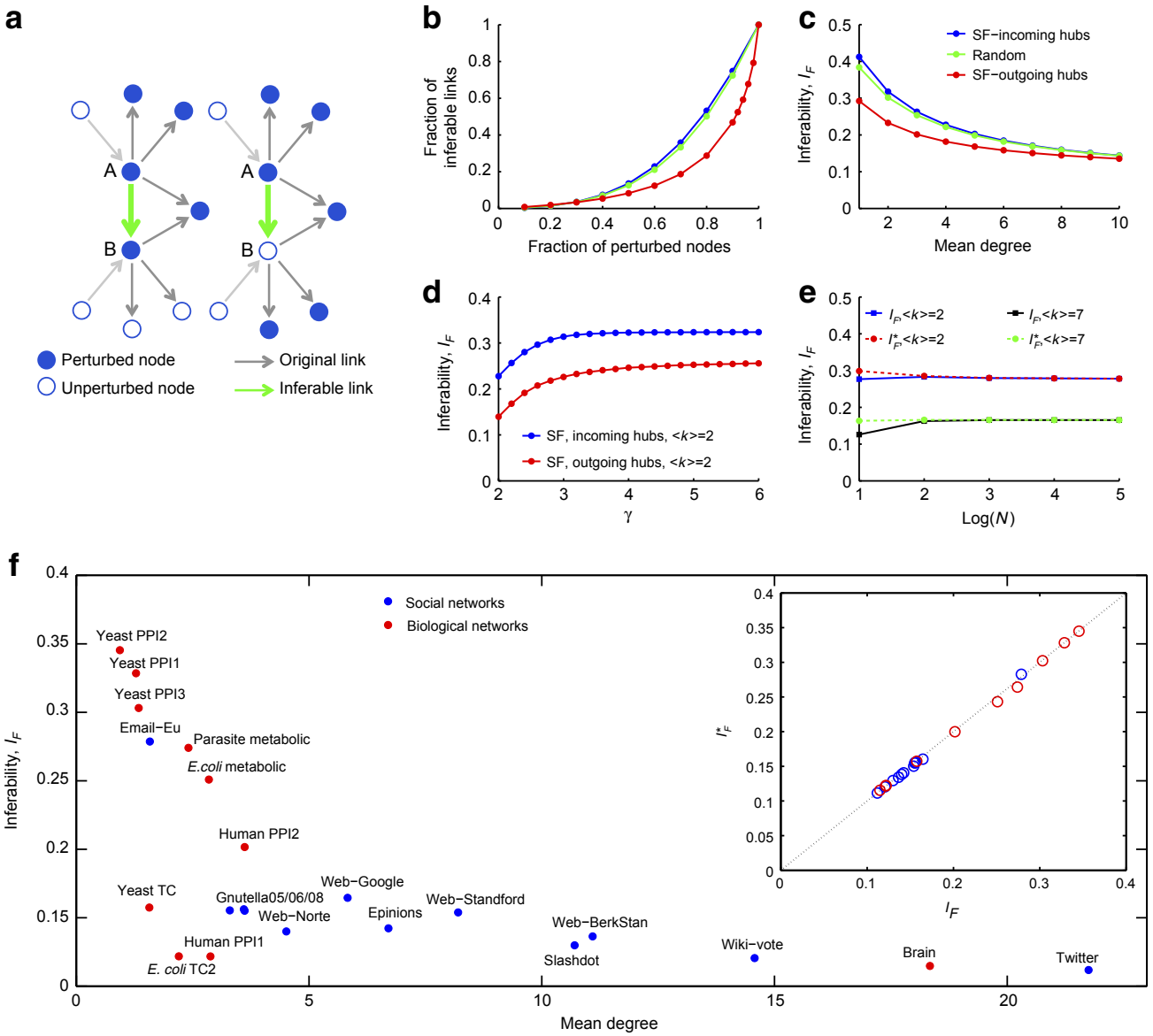


Fig. 2 Inferability as a function of network parameters. **a** Directed links are inferable if either all outgoing links of the source node point to perturbed nodes including the target node (left panel) or if all outgoing links of source node and target node point to perturbed nodes, with the target node not perturbed (right panel). **b** Fraction of inferable links against the fraction of perturbed nodes using three network types: (i) scale-free network with exponent $\gamma = 2.5$ and mean degree $\langle k \rangle = 3$, where nodes of higher degree target nodes of lower degree (outgoing hubs), (ii) same network as (i) but with all link directions inverted (incoming hubs) or a network generated by random insertion of links with the same mean degree as scale-free networks (random network). Colour coding as in **c**. **c** Network inferability versus mean degree, using networks of **b**. **d** Network inferability versus scaling exponent for two types of scale-free networks. **e** Asymptotic invariance of the two inferability measures introduced in the main text with respect to network size. **f** Network inferability as a function of mean degree for social and biological networks. Correlation between the two inferability measures introduced in the main text (inset)

transmitting information back to it, as we exclude bidirectional links from our analysis. This makes the network (Fig. 1a, left network) fully inferable, as the link between the remaining two perturbed nodes can be inferred by collecting experiments for which the target node is unperturbed. We emphasise that our approach to network inferability does not account for a priori known restrictions on the network topology, as in the case of directed acyclic graphs. Such constraints can strongly increase the inferability of directed links[8].

As $F(q)$ is an upper bound for the expected number of directed links that can be inferred from stationary node activities in the absence of noise and other constraints on the network structure, we now ask how this bound is related to the structural properties of the network. To compare different network architectures, it is useful to define the network inferability, I_F, as the area under the $F(q)$-curve, $I_F := \int_0^1 F(q)\mathrm{d}q$. Comparison of I_F between two general classes of network structures with node degrees either power law or Poisson distributed shows that networks that are enriched with nodes of high outdegree are the most difficult ones to infer (Figs 1b and 2b). The reason is that whenever hubs with high outdegree are perturbed there is a high chance that they target more than one of the unperturbed nodes and without additional perturbations it is impossible to detect which of the targets are affected directly and which indirectly. Differences in inferability due to network structure are most predominant for networks with low mean degree and become less predominant with high mean degree (Fig. 2c). As our measure of inferablity, I_F, is essentially determined by the outdegree distribution, the curve

starts saturating for scale-free exponents $\gamma > 3$, as in this regime the variance of the number of links per node is essentially constant for increasing γ and fixed mean degree[9] (Fig. 2d). The network inferability, I_F, is asymptotically independent of network size (Fig. 2e). We further investigated the inferability of causal interactions in biological and social networks as a function of the mean degree (Fig. 2f). The decreasing trend can be explained by the higher number of alternative routes that come with a more strongly connected network. The low inferability of gene-regulatory networks can be attributed to master regulators that regulate a large fraction of the genome (hubs with high outdegree), whereas the comparatively high inferability of protein interaction networks is a consequence of the low number of different binding domains per protein and that only a fraction of the existing interactions have been identified due to limitations of experimental methods[10]. If we assume that the conditional probability $P(k, l, m|k \to l)$ of finding two connected nodes in the directed network, where the source node has $k \geq 1$ outgoing links, the target node has $l \geq 0$ outgoing links, and both share m nodes as common targets of their outgoing links, can be factorised, the resulting inferability measure, I_F^*, is simply a function of the outdegree distributions, $P(k)$ and $P(l)$. We observed that $I_F^* \approx I_F$ for all networks investigated in this work (Fig. 2f, inset). This result shows that for a large variety of networks structures the outdegree is the dominating factor that determines network inferability. Consequently, if the perturbed nodes are not selected at random but are biased towards higher outdegree, inferability is further reduced.

Network inference concepts. From our analyses of network inferability we gained the insight that the number of potential alternative routes how a source node can affect a target node correlates positively with the outdegree of the source node and inversely with the expected inferability of the directed link between source and target, given that perturbed nodes are uniformly distributed in the network. Consequently, network inference algorithms should strongly benefit from an a priori reduction in the number of alternative routes. In the following we present an unbiased network inference algorithm that eliminates alternative routes with low signal-to-noise ratio as a preprocessing step. Inference of transcriptional networks on genome scale is best realised by methods that are (i) asymptotically unbiased, (ii) scalable to large network sizes, (iii) sensitive to feed-forward loops[11] and (iv) can handle data sets with and without knowledge about which nodes are targeted by experimentally induced perturbations[7, 12–16] (Supplementary Note 2). Inference methods for directed networks typically require individual perturbation of all nodes[7] or many perturbations of different strengths to compute conditional association measures[6, 17] or conditional probabilities[18]. Generation of time course data seems to be the most natural way to infer directed networks by simply analysing the temporal ordering of the transcriptional activities[19, 20]. However, this approach precludes the use of knockout experiments and requires fast acting perturbations in combination with monitoring node activities over time, which is experimentally demanding, especially if nodes respond on very different time scales[21].

Experimental variability and technical noise. Inference is further complicated by the fact that transcriptome data contain a significant amount of stochastic variation between biological replicates despite pooling over millions of cells (Fig. 3a). It is interesting to see that the variation across biological replicates for baker's yeast[3] is close to a normal distribution and follows almost exactly a t-distribution with 11 degrees of freedom over five standard deviations (Fig. 3a, inset). The same data set also shows

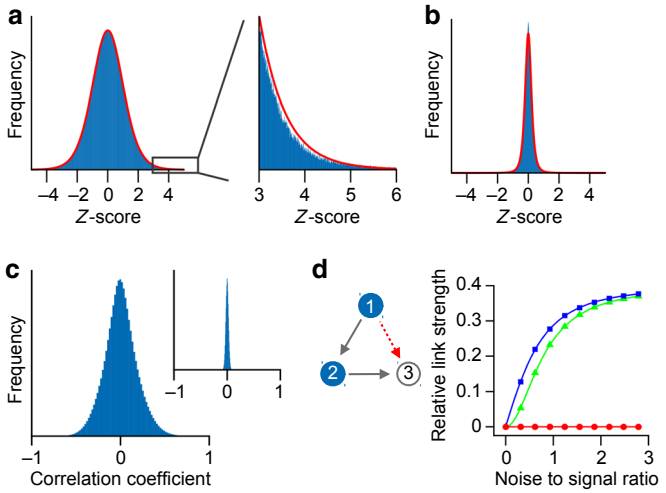


Fig. 3 Distribution of wild-type expression levels for *S. cerevisiae* from 748 biological replicates. **a** Distribution of the relative expression, $\log_2(r_i)$, with $r_i := x_i/x_i^{pool}$ and x_i the expression of gene i relative to x_i^{pool} followed by standardisation of the $\log_2$ fold changes (z-score). The values x_i^{pool} have been obtained by first pooling the 748 biological replicates before measuring gene expression. The distribution is well described within five standard deviations by a t-distribution with 11 degrees of freedom (red line). **b** Distribution as in **a** but now for differences among technical replicates. **c** Correlation between gene expression levels is significantly higher than expected by chance (inset) **d** Illustration of a noise induced false positive link (red arrow) as described in the main text. Data for a three-node network with two links was generated by applying independent perturbations on node 1 and node 2. The link strength of the non-existing link from node 1 to node 3 relative to the existing link from node 2 to node 3 was inferred using three different methods (i) partial correlations (blue squares), (ii) conditional mutual information (green triangles) and PRC (red circles)

that variability among biological replicates is much larger than technical noise (Fig. 3b) for this experimental setup. As variability among biological replicates may arise from subtle differences in growth conditions that induce changes in gene regulation, we expected to see significant cross-correlations among genes (Fig. 3c), whose magnitude is much larger than expected by chance (Fig. 3c, inset). These cross-correlations can be exploited for inferring the structure of undirected networks[12], if the driving noise is independent and identically distributed for all nodes (Supplementary Note 2). In contrast, technical noise not only reduces the statistical significance for detecting true interactions but can also induce a significant fraction of false positive interactions, especially if the interaction network under investigation is sparse. Such noise induced misclassification of links can be illustrated by a simple linear network $A \to B \to C$ for which standard inference methods—such as partial correlations–interpret the information that A has about C erroneously as a direct link between A and C if the state of B is corrupted by measurement noise (Fig. 3d). The reason is that a part of the correlation between A and C cannot be explained by B.

Algorithm for large-scale and unbiased network inference. To make use of the rapidly growing amount of single-gene knockout screens for which transcriptome data are[3] or may become available[22, 23], we developed a method to infer directed networks on a genome scale, where the number of genetic perturbations is typically below the number of nodes or genes in the network (Online Methods). In brief, our method uses the concept of

probabilistic principle component analysis[24] to compute partial response coefficients (PRC) that are asymptotically unbiased with respect to Gaussian measurement noise. In addition, the algorithm provides a feature to identify non-inferable links, which are removed before statistical analysis. In the absence of noise, our numerical method correctly predicts the fraction of links that are inferable, $F(q)$ (Supplementary Note 1), for a network with links represented by linear functions of slope one. To evaluate the performance of our method we generated two synthetic knockout data sets that closely resemble the gene-regulatory network structure of baker's yeast, using the GeneNetWeaver software[25] that uses a hierarchical network structure and our own generative model that uses a scale-free network structure (Supplementary Note 3). We added Gaussian measurement noise to the synthetic data with a standard deviation of 10% the $\log_2$ fold-change in expression level for each perturbation for each gene. Residual bootstrapping among replicates was used to quantify the statistical significance of the inferred link strengths. In comparison with standard inference methods, such as partial correlations[12–14, 26, 27], our method shows a significantly higher performance in the absence of any penalties that enforce sparse network structures (Fig. 4b, left panel). The improved performance of our approach can be assigned to the fact that the method is unbiased with respect to measurement noise (Online Methods).

To further improve the predictive power of our method we included the prior knowledge that transcriptional networks are highly sparse. Sparsity constraints are typically realised by penalising either the existence of links or the link strengths by adding appropriate cost functions, such as L^1-norm regularised regression (Lasso)[28]. Adding a cost function to the main objective comes with the problem to trade-off the log-likelihood against the number of links in the network whose strength is allowed to be non-zero. In the absence of experimentally verified interactions there is no obvious way how to determine a suitable regularisation parameter that weights the likelihood against the cost function, which is one of the great weaknesses of such methods.

In our approach we reduce network complexity by assuming that functionally relevant information in molecular networks can only pass through nodes whose response to perturbations is significantly above the base line that is given by the variability among biological replicates. The individual noise levels can be estimated from natural variations among wild-type experimental replicates (Fig. 3a). The significance level that removes nodes from the network with low signal-to-noise ratio can be set to a desired false discovery rate. It can be shown that removal of noisy nodes imposes a sparsity constraint on the inference problem (Online Methods). The different steps required to arrive at a list of significant links are illustrated in Fig. 4a. In the first step, genes are grouped in clusters that are co-expressed under all perturbations. These clusters are treated as single network nodes in the subsequent steps. In the second step, only those samples are extracted from the data set that correspond to a perturbation of a chosen gene—the source node—with no other genes perturbed (node 5 in Fig. 4a). From this reduced data set, we identify all nodes in the network that change expression above a given significance level upon perturbing the source node. These significantly responding nodes define a subnetwork for each source node, which is typically much smaller in size than the complete network. In the third step, we collect all perturbation

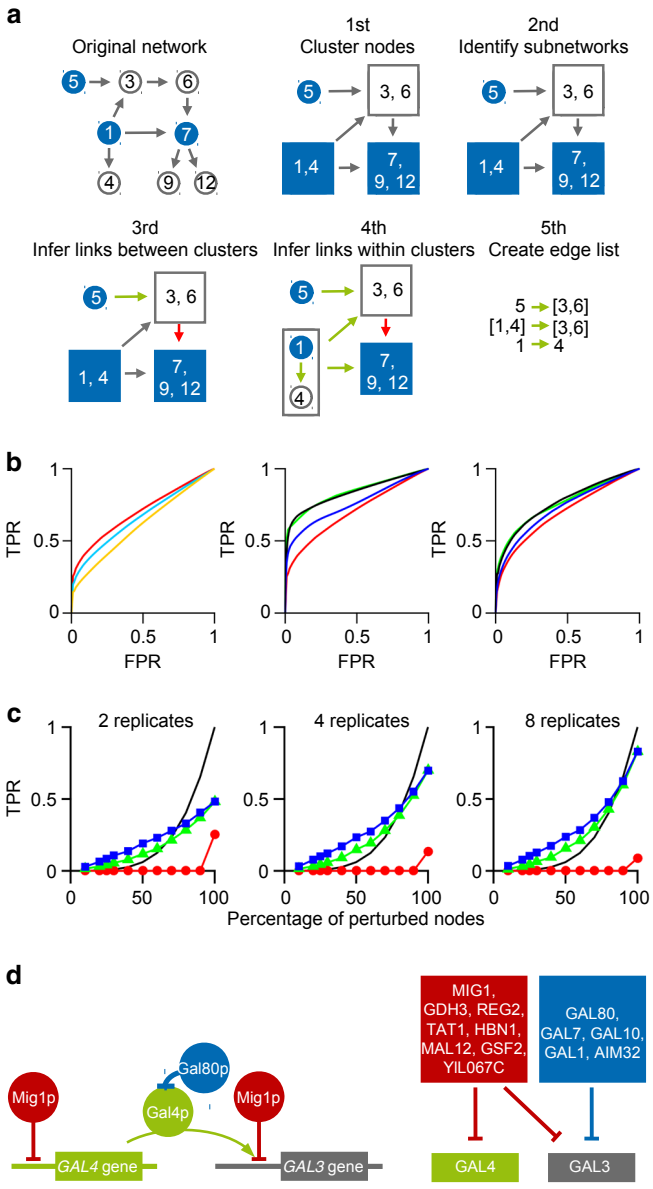


Fig. 4 Performance of our method. **a** Flow-chart showing the algorithmic steps for network inference as explained in the main text. **b** Receiver Operating Characteristic (ROC) curves for 300-node scale-free networks with additive Gaussian measurement noise of 10% of the expression level and 25% of the nodes perturbed. Data were generated using GeneNetWeaver (left and middle panel) as well as using scale-free network structure with mean degree of $\langle k \rangle = 2$ and scaling exponent $\gamma = 2.5$ (right panel, Supplementary Note 3). Here, the true positive rate is computed with respect to the inferable links[39]. Performance of inference methods without sparsity constraints (left panel): PRC (red), partial correlations/linear regression (turquoise) and conditional mutual information (orange). Performance of inference methods with sparsity constraints (middle and right panel): PRC with subnetwork method (green) and Lasso (black) both applied to a subset of significantly responding nodes that were selected with 1% false discovery rate, Lasso regression applied to all 300 nodes (blue), and PRC from left panel (red) for comparison. **c** True positives for the same scale-free network of **b**, with 2, 4 and 8 experimental replicates with 5% false discovery rate for both significantly responding nodes and link strength: PRC (red), PRC with subnetwork method (green), PRC with subnetwork and clustering method (blue), and $F(q)$ (black line). **d** The *S. cerevisiae* GAL network as an example for a gene-regulatory network where phosphorylated Mig1 sets the basal expression levels of Gal4 and one of its many regulatory targets, Gal3. Gal4 protein can activate Gal3 expression but is inactivated upon binding of Gal80 protein. The transcriptome data set contains knockout mutants for GAL80 and MIG1 but not for the remaining GAL genes. A schematic representation of the key molecular mechanisms (left) and links inferred from transcriptome data[3] (right)

data from the complete data set for all nodes that are part of the subnetwork. Before inferring a direct interaction that points from the source node to a given target node in the subnetwork (green arrows in Fig. 4a), we remove all experiments from the data set where the target node is perturbed. The second and third steps essentially realise the counting procedure of inferable links as illustrated in Fig. 2a, with the difference that significant links are identified by PRCs in combination with residual bootstrapping over replicates (Online Methods, Supplementary Note 3). In the fourth step, we collect all clusters of co-expressed genes that contain exactly two nodes, with one of the nodes perturbed and check statistical significance of the directed link between them. In the fifth step, all significant links are collected in an edge list. We refer to these five steps as the clustering method. If we remove all links from the edge list that have more than one node in a source cluster or more than one node in a target cluster, we obtain an edge list that corresponds to links between single genes. This reduced edge list would also arise by skipping the clustering step and we refer to the remaining inference steps that compute links between single genes as subnetwork method.

Performance of the proposed inference algorithm. The Lasso method in combination with bootstrapping has been benchmarked as one of the highest performing network inference methods for in silico generated expression data[29]. The receiver operating characteristic (ROC) curve of the subnetwork method shows better performance than the Lasso method (Fig. 4b, middle and right panel) after adjusting the regularisation parameter of the Lasso method such that the area under the ROC curve is maximised. However, a significant performance boost for the Lasso method can be achieved by applying the second step of our method that removes noisy nodes, resulting in comparable performance of Lasso with the subnetwork method for the case that validation data exist such that the regularisation parameter can be determined (Fig. 4b, middle and right panel).

To get insight into the optimal experimental design for generating data for network inference, we computed the fraction of correctly inferred links and compared them against the fraction of independently perturbed nodes for different numbers of experimental replicates. We compared three different variants of our approach: PRC, PRC together with subnetwork method and PRC together with clustering method (Fig. 4c). As all variants share PRC as underlying inference method (Online Methods), the observed strong increase in performance can be assigned to the sparsity constraint that comes with the subnetwork method or the clustering method. Owing to this constraint, both the subnetwork method and the clustering method can have higher accuracy than the noise-free analytical solution, as the latter does not enforce sparse network structures. The results show that in the presence of 10% measurement noise the amount of available replicates limits the true positive rate, even if 100% of nodes are perturbed. Inference of >80% of the network can only be achieved if the number of replicates is sufficiently high.

To benchmark the performance of our algorithms in comparison to others, we applied our method to the DREAM3 in silico network inference challenge[30]. The provided data set of this challenge has the advantage that full information about the identity of perturbed nodes is given. We ignored the transient information from time series and used the stationary state of time course data to estimate the variation in expression between biological replicates. To identify the significantly responding nodes, we used a Bonferroni corrected significance level of $\alpha = 0.05/N$, where the number of alternative hypotheses—or the number of possible incoming links for a given target node in our case–are bounded by number of possible source nodes in the network, $N-1$. To make sure that we correctly implemented the published performance evaluation method that is based on curve fitting a sampled null hypothesis[31], we followed the proposed curve fitting procedure suggested by the organisers of the challenge by using different exponential family distributions for each tail, and alternatively by using a single t-distribution to fit AUROC null hypothesis samples. The results are shown in Table 1 and Supplementary Data 4. The overall second place among the other 29 inference methods should be interpreted in the light that the better performing algorithm uses extensive hyperparameter tuning, makes use of transient data, and does not scale well with network size[32]. Furthermore, our approach seems to be robust with respect to the chosen significance level as changing α by one order of magnitude did not affect the ranking. However, we emphasise that for 'large p small n' problems, where the number of parameters exceeds the number of independent data points, preprocessing often has a larger effect on performance than the inference method itself[30]. For our algorithm the performance boost is a consequence of generating subnetworks as preprocessing step.

Application to yeast genome knockout data. To evaluate the performance of our approach on real data, we use one of the largest publicly available transcriptome data sets[3], comprising of transcriptomes that cover 6170 genes for 1441 single-gene knockouts that can be utilised for network inference using PRC. We use the galactose utilisation network as a gene-regulatory example, which is one of the best characterised gene-regulatory modules in yeast[33]. The regulatory mechanism of the GAL4 gene as a key regulator is shown in Fig. 4d, left panel. As information about phosphorylation and protein interaction is absent in expression data, the inferred network structure from transcriptome data with GAL4 and GAL80 perturbed is different from the known gene regulation but can identify major regulators and their targets. Whether the gene AIM32—which is not known to be part of the GAL network—is co-regulated with GAL80 or an artefact of the knockout screen is difficult to judge. Both options are possible as AIM32 is located in close vicinity to GAL80 on the genome. By sorting genes with respect to their number of statistically significant outgoing links, we can identify potential key regulators. Besides transcription factors, the regulators with highest statistical significance are factors involved in chromatin remodelling, signalling kinases, and genes involved in ubiquitination (Supplementary Data 1–3). This result—although expected for eukaryotes—is inaccessible for inference methods

Table 1 Ranking and overall scores (in parantheses) among the original participants of the DREAM3 in silico network inference challenge

	10 nodes $\alpha = 0.05/10$	50 nodes $\alpha = 0.05/50$	100 nodes $\alpha = 0.05/100$
Original scoring method	2nd (4.64)	2nd (31.43)	2nd/1st (55.98)
AUROC background fitted with t-distribution	2nd (4.14)	3rd (30.10)	2nd/1st (50.06)

Scores were obtained with the original scoring method and a scoring method in which the AUROC background distribution was fitted with a t-distribution. Here, α denotes the significance level for the identification of nodes that are significantly affected by perturbations

that a priori fix known transcription factors as regulatory sources. However, as the number of deleted genes in this data set comprise just 23% of the genes for which transcript levels have been measured, we can estimate from our simulations that we have inferred <10% of the direct interactions in the transcriptional network of yeast.

Discussion

We have developed an unbiased network inference method for perturbation experiments that target individual nodes in the network. Consequently, node activity data that result from unspecific perturbations cannot be exploited by this algorithm in its present form. As individual gene knockout or knockdowns dominate many large-scale experimental studies of node activities in biological networks[3, 23] and their genome-wide coverage is constantly improving[22, 34], we expect that the biological data sets to which the algorithm can be applied will rapidly increase in the near future. However, currently most of the large-scale knockout or knockdown screens lack complete coverage of mutants and often come with low number of experimental replicates, if any. In this work we have shown that insufficient coverage of perturbed nodes in transciptome data fundamentally limit the amount of links that can be inferred, independently of the employed inference algorithm and that high statistical power requires a significant amount of replicates to drive down effects of experimental variability and measurement noise. We therefore introduced a network inference approach that is able to detect significant links for the case that only a fraction of nodes are perturbed, removes nodes with low signal-to-noise ratio from the network, and makes use of an inference algorithm that is insensitive to measurement noise. Including prior knowledge about network complexity and reducing the effects of noise is crucial for network inference problems, where the number of parameters, e.g., link strengths, scale quadratically with network size and often exceed the number of measured data points. Good scaling behaviour and the absence of time-consuming hyperparameter tuning make our approach an easily applicable network inference tool that shows competitive performance with state-of-the-art methods. However, even when complete coverage of single-gene perturbations together with a high number of experimental replicates of transcriptome data are available, the inferred transcriptional network cannot be directly translated into a model that reflects the biochemical reality of gene regulation. The reason is that gene regulation can involve complex molecular interactions on DNA, RNA, protein and small molecule level that result in direct interactions between mRNA levels. An example of such complex interactions is the observed regulation by the human oncogene IDH1—a metabolic enzyme involved in the citric acid cycle. Mutational loss of normal enzymatic function of IDH1 and production of the metabolite 2-hydroxyglutarate can affect the activity of an epigenetic regulator, which promotes tumorgenesis by reprogramming transcriptional activity on genome scale[35]. Inference of such complex molecular interactions would require a combination of different high-throughput technologies, with the challenge that different methods typically show large differences in sensitivity and coverage[36].

Methods

PRC. We aim at inferring direct interactions between N observable molecular components, such as transcripts or proteins, by measuring their concentrations. We define by $\boldsymbol{y} \in \mathbb{R}^N$ an N dimensional vector that represents the logarithm of these concentrations, which is the natural scale where experimental data are reported. We assume that the available data set has been generated from P perturbation experiments, $\{\boldsymbol{y}_k\}_{k=1}^P$, which may also include experimental replicates. We further assume that the molecular targets of the perturbations are known, as it is the case for gene knockout or knockdown experiments. The elements of the interaction matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ define the strengths of the directed interactions among the molecular components, for example, A_{ij} quantifies the direct impact of component j on component i. Given the available experimental data, our aim is to correctly classify the off-diagonal elements of $\boldsymbol{A}$ as zero or non-zero to obtain the structural organisation of the interaction network. We assume that the observed component abundance on log-scale, $\boldsymbol{y}^{obs}$, differs from the true value, $\boldsymbol{y}$, by additive measurement noise, ϵ, which is characterised by zero mean, $\mathbb{E}[\epsilon] = 0$, and variance, $\mathbb{E}[\epsilon\epsilon^T] = \sigma^2\boldsymbol{I}_N$, with $\boldsymbol{I}_N$ the N dimensional identity matrix. We assume that the observed data can by described to sufficient accuracy by a linear stationary model

$$\begin{aligned} 0 &= \boldsymbol{A}\left(\boldsymbol{y} - \boldsymbol{y}^{ref}\right) + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y}^{obs} &= \boldsymbol{y} + \epsilon, \end{aligned} \tag{1}$$

with $\boldsymbol{A}$ negative definite to ensure stability. Equations of this type typically arise from linear expansion of a non-linear model around a reference state, $\boldsymbol{y}^{ref}$. Linear models are usually preferred for network inference a on larger scale, as the amount of data often limit model complexity and the fact that linear models can give surprisingly good resuits for non-linear cases. The perturbation vector $\boldsymbol{u}$ reflects perturbations that persist long enough to propagate through the network, such as mutations that affect gene activity. Here, $\boldsymbol{u}$ is defined such that for $\boldsymbol{u} = 0$ the system approaches the reference state $\boldsymbol{y} = \boldsymbol{y}^{ref}$. Note that the reference state, $\boldsymbol{y}^{ref}$ is not necessarily the unperturbed state but could be also defined as the average over perturbed and unperturbed states. We assume that the perturbation forces are sampled from a standard normal distribution, with mean $\mathbb{E}[\boldsymbol{u}] = 0$ and covariance matrix $\mathbb{E}[\boldsymbol{u}\boldsymbol{u}^T] = \boldsymbol{I}_N$. The identity matrix is a consequence of the fact that we can absorb the associated standard deviations of the perturbative forces, $\boldsymbol{u}$, in the matrix $\boldsymbol{B} \in \mathbb{R}^{N\times N}$. We introduce normal distributed perturbations for mathematical convenience, as this implies that also $\boldsymbol{y}$ is normal distributed and the resulting maximum likelihood approach is analytically solvable. In general, only the positions of the non-zero elements of $\boldsymbol{B}$ are known from the experimental setup but their actual values are unknown. Using a linear model that operates on log-scale of physical quantities implies that only perturbations can be modelled that act multiplicatively on molecular concentrations. Fortunately, most enzymatic reactions typically fall into this class, such as sequestration and inhibition by other components and also knockout and knockdown experiments can be described on multiplicative level. From Eq. (1) we can derive a relation between the interaction matrix $\boldsymbol{A}$ and the covariance matrix of observed component abundances

$$\begin{aligned} \boldsymbol{C} &:= \mathbb{E}\left[\left(\boldsymbol{y}^{obs} - \boldsymbol{y}^{ref}\right)\left(\boldsymbol{y}^{obs} - \boldsymbol{y}^{ref}\right)^T\right] \\ &= \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{A}^{-T} + \sigma^2\boldsymbol{I}_N \end{aligned} \tag{2}$$

We exploit Eq. (2) to infer directed networks from correlation data. Here, we assume that component abundances are obtained from averaging over a large number of cells. In this case, fast fluctuating perturbations that arise from thermal noise and can be observed only on single-cell level average out. To infer the interaction matrix, $\boldsymbol{A}$, we start with singular value decomposition of the matrix product $\boldsymbol{A}^{-1}\boldsymbol{B}$

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T := \boldsymbol{A}^{-1}\boldsymbol{B} \quad \Rightarrow \quad \boldsymbol{B} = \boldsymbol{A}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{3}$$

with $\boldsymbol{U}$ and $\boldsymbol{V}$ orthogonal matrices and $\boldsymbol{\Sigma}$ a diagonal matrix containing the singular values. The negative definite matrix $\boldsymbol{A}$ has full rank and hence is invertible. In the following, we show that it is possible to infer the strength of a directed link between a sender node j and a receiver node i, if all direct perturbations on receiver node i are removed from the data set and if a significant partial correlation between i and j exists. Removing the perturbation data for node i implies that the matrix $\boldsymbol{B}$ has at least one zero entry. As a consequence, $N_0 \geq 1$ singular values are zero–as in general not all nodes are perturbed–and the corresponding rows of $\boldsymbol{U}$ span the left nullspace of $\boldsymbol{A}^{-1}\boldsymbol{B}$. In the absence of fast fluctuating perturbations, $\gamma = 0$, we can rewrite the covariance matrix as

$$\boldsymbol{C} = \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{A}^{-T} + \sigma^2\boldsymbol{I}_N \tag{4}$$

$$= \boldsymbol{U}(\boldsymbol{\Sigma}^2 + \sigma^2\boldsymbol{I}_N)\boldsymbol{U}^T. \tag{5}$$

Assuming that the observed node activities follow a multivariate normal distribution, we can find estimates for the unknown orthogonal matrix $\boldsymbol{U}$, the singular values $\boldsymbol{\Sigma}$, and the observational noise σ by maximising the log-likelihood function $\mathcal{L}$ under the constraint $\boldsymbol{U}_l^T\boldsymbol{U}_k = \delta_{lk}$, with $\boldsymbol{U}_k$ the k-th column vector of $\boldsymbol{U}$ and δ_{lk} the Kronecker delta. It fact, it suffices to constrain the norm of the vectors, $\|\boldsymbol{U}_k\|$, as the corresponding maximum likelihood solution leads to an eigenvalue problem with $\boldsymbol{U}_k$ as eigenvectors, which can always be made orthogonal. We can therefore define the likelihood function by

$$\mathcal{L} := \ln \prod_{n=1}^{P} \mathcal{N}\left(\boldsymbol{y}_n^{obs} | \boldsymbol{y}^{ref}, \boldsymbol{C}\right) + \sum_{k=1}^{N} \lambda_k\left(\boldsymbol{U}_k^T\boldsymbol{U}_k - 1\right) \tag{6}$$

$$= -\tfrac{P}{2}\{M\ln 2\pi + \ln|\boldsymbol{C}| + \mathrm{tr}(\boldsymbol{C}^{-1}\boldsymbol{S})\} + \sum_{k=1}^{N} \lambda_k\left(\boldsymbol{U}_k^T\boldsymbol{U}_k - 1\right) \tag{7}$$

Here, $\boldsymbol{S} := \frac{1}{P}\sum_{n=1}^{P}\left(\boldsymbol{y}_n^{obs} - \boldsymbol{y}^{ref}\right)\left(\boldsymbol{y}_n^{obs} - \boldsymbol{y}^{ref}\right)^T$ and $\boldsymbol{y}^{ref} := \frac{1}{P}\sum_{n=1}^{P}\boldsymbol{y}_n^{obs}$ denote maximum likelihood estimates of the covariance matrix[37]. From this definition of $\boldsymbol{y}^{ref}$ follows that the initially introduced perturbation vector, $\boldsymbol{u}$, must satisfy, $\frac{1}{P}\sum_{n=1}^{P}\boldsymbol{u}_n = \boldsymbol{0}$. We further defined with λ_k a Lagrange multiplier and denoted by tr(.) the trace of a matrix. In the following calculations, we substitute $\boldsymbol{S}$ by the unbiased sample covariance matrix, $\boldsymbol{S} \to P(P-1)^{-1}\boldsymbol{S}$. Note that $\boldsymbol{V}$ must disappear in the likelihood function as the covariance matrix of $\boldsymbol{u}$ is invariant under any orthogonal transformation $\boldsymbol{u} \to \boldsymbol{V}^T\boldsymbol{u}$.

The maximum of the log-likelihood function is determined by the conditions $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{U}_k = 0$, $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{\Sigma}_{kk} = 0$, and $\mathrm{d}\mathcal{L}/\mathrm{d}\sigma^2 = 0$, which results in

$$\boldsymbol{S}\hat{\boldsymbol{U}}_k = \Lambda_k\hat{\boldsymbol{U}}_k \quad \text{with } \Lambda_1 \le \Lambda_2 \le \ldots \le \Lambda_N \tag{8}$$

$$\hat{\sigma}^2 = \frac{1}{N_0}\sum_{k=1}^{N_0}\Lambda_k \tag{9}$$

$$\hat{\Sigma}_{kk} = \begin{cases} \sqrt{\Lambda_k - \sigma^2} & \text{for} \quad k > N_0 \\ 0 & \text{for} \quad k \le N_0 \end{cases} \tag{10}$$

showing that maximum likelihood estimates of $\hat{\boldsymbol{U}}$, $\hat{\sigma}^2$, and $\hat{\boldsymbol{\Sigma}}$ are determined by the sample covariance matrix $\boldsymbol{S}$. If $N_0 > 1$ and the full-rank sample covariance matrix is significantly different from a block-diagonal form—e.g., the network is not separable in subnetworks–the orientations of the corresponding N_0 eigenvectors are determined by sampling noise in the space orthogonal to remaining $N - N_0$ eigenvectors. In case that we have less data points than nodes in the network—e.g., the number of perturbed nodes times their replicates is smaller than the network size—some of the N_0 smallest eigenvalues become exactly zero and as a consequence the noise level, $\hat{\sigma}$, is underestimated. Although a maximum likelihood solution exists in this case, it is necessary to regularise the covariance matrix, $\boldsymbol{S} \to (1-\epsilon)\boldsymbol{S} + \epsilon\boldsymbol{I}_N$, with ϵ a regularisation parameter[37], as a correct estimate of the noise level is essential for statistical analysis. Note that the derivation of the maximum likelihood solution is mathematically equivalent to the derivation of principle component analysis from a probabilistic perspective[24].

Solving the matrix equation, Eq. (3), for $\boldsymbol{A}$ gives

$$\boldsymbol{A} = \left(\boldsymbol{B}\boldsymbol{V}\boldsymbol{\Sigma}^{+} + \boldsymbol{W}\boldsymbol{\Sigma}^{0}\right)\boldsymbol{U}^T \tag{11}$$

with $\boldsymbol{\Sigma}^{+}$ the pseudoinverse of $\boldsymbol{\Sigma}$. As the matrix $\boldsymbol{A}$ has full rank, we complement $\boldsymbol{\Sigma}^{+}$ with an unknown diagonal matrix $\boldsymbol{\Sigma}^{0}$ that has non-zero values where $\boldsymbol{\Sigma}^{+}$ has zero values and vice versa and complement $\boldsymbol{BV}$ with an unknown orthogonal matrix $\boldsymbol{W}$. Note that by construction, $\boldsymbol{\Sigma}^{+}\boldsymbol{U}^T$ and $\boldsymbol{\Sigma}^{+}\boldsymbol{U}^T$ map from complementary subspaces and thereby ensure that $\boldsymbol{A}$ has full rank. The fact that $\boldsymbol{V}$, $\boldsymbol{W}$ and $\boldsymbol{\Sigma}^{0}$ cannot be determined from $\boldsymbol{S}$ shows that $\boldsymbol{A}$ cannot be computed from a single covariance matrix. A more general case arises when measurement noise is independent but not isotropic, $\sigma^2\boldsymbol{I} \to \sigma^2\boldsymbol{D}$, with $\boldsymbol{D} = \mathrm{diag}(r_1, r_2, \ldots, r_N)$ a diagonal matrix with known positive elements that contains scaled noise variances, $r_i := \sigma_i^2/\sigma^2$, resulting in

$$\boldsymbol{C} = \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{A}^{-T} + \sigma^2\boldsymbol{D} \tag{12}$$

A transformation to isotropic noise is possible by multiplying both sides of Eq. (12) by $\boldsymbol{D}^{-\frac{1}{2}}$, which changes the result Eq. (11) to

$$\boldsymbol{A} = \left(\boldsymbol{B}\boldsymbol{V}\boldsymbol{\Sigma}^{+} + \boldsymbol{W}\boldsymbol{\Sigma}^{0}\right)\boldsymbol{U}^T\boldsymbol{D}^{-\frac{1}{2}} \tag{13}$$

with $\boldsymbol{U}$ the eigenvectors of $\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{S}\boldsymbol{D}^{-\frac{1}{2}}$.

Case $N_0 = 1$. We assume that the i-th node is the only unperturbed node in the network and hence set $B_{il} = 0$ for all l. From Eq. (11) we obtain a unique solution for the i-th row of $\boldsymbol{A}$ relative to the diagonal element, A_{ii},

$$\frac{A_{ij}}{A_{ii}} = \frac{\sum_{k,l=1}^{N} B_{il}V_{lk}\Sigma_{kk} + \sum_{k=1}^{N} W_{ik}\Sigma_{kk}^{0}U_{kj}^T}{\sum_{k,l=1}^{N} B_{il}V_{lk}\Sigma_{kk} + \sum_{k=1}^{N} W_{ik}\Sigma_{kk}^{0}U_{ki}^T} = \frac{U_{kj}^T}{U_{ki}^T}\bigg|_{k=1} = \frac{U_{j1}}{U_{i1}} \tag{14}$$

with U_{j1} the j-th element of the eigenvector that has the smallest eigenvalue. Note that the first term in the brackets vanishes as $B_{il} = 0$ and Σ_{11}^{0} is the only non-zero element of $\boldsymbol{\Sigma}^{0}$. The important point is that any dependency on σ–which affects eigenvalues but not eigenfunctions–has dropped out, making this method asymptotically unbiased with respect to measurement noise. The fact that we can determine the elements of the i-th row of $\boldsymbol{A}$ only relative to a reference value, A_{ii}, is rooted in fact that we have to determine the N parameters $\{A_{i1}, \ldots, A_{ii}, \ldots, A_{iN}\}$ from $N-1$ perturbations. As a consequence, the strengths of the links onto the target nodes cannot be compared directly if their restoring forces or degradation rates, A_{ii}, are different. Generally, only relative values of $\boldsymbol{A}$ can be determined, as the average perturbation strength on node i cannot be disentangled from its restoring force A_{il}–a problem that is typically circumvented by defining $A_{ii} := -1$ for all i[7, 13, 15]. For the case that all nodes in the network are perturbed one-by-one, we can cycle through the network and remove the perturbations that act on the current receiver node, whereas keeping the perturbations on the remaining nodes. By computing the N corresponding covariance matrices and their eigenvectors, we can infer the complete network structure from Eq. (14) if the data quality is sufficiently high. Note that the method makes use of the fact that multi-node perturbations can be realised by superposition of single-node perturbations, which is a special property of linear models.

Case $N_0 > 1$. If more than one node are not perturbed we get from Eq. (11)

$$\frac{A_{ij}}{A_{ii}} = \frac{\sum_{k=1}^{N_0} W_{ik}\Sigma_{kk}^{0}U_{kj}^T}{\sum_{k=1}^{N_0} W_{ik}\Sigma_{kk}^{0}U_{ki}^T} \tag{15}$$

Non-unique solutions of Eq. (15) can arise if a given fraction of the variance of the receiver node i can be explained by more than one sender node, for example, when a perturbed node j targets two nodes with index i and l. In this case it is unclear from the node activity data whether i is affected directly by j or indirectly through l, or by a combination of both routes. If node l is not perturbed or only weakly perturbed, a statistical criterion is needed to decide about inferability or identifyability of the link $j \to i$, which can be computed numerically as follows: To find out whether j transmits a significant amount of information to i that is not passing through l, we first remove node j from the observable nodes of the network but keep its perturbative effect on other nodes in the data set. We then determine the link strengths $\boldsymbol{A}'$ for the remaining network of size $N-1$. To construct a possible realisation of $\boldsymbol{A}'$ we set in Eq. (15) the non-zero values of $\boldsymbol{\Sigma}^{0}$ to unity and use $\boldsymbol{W} = \boldsymbol{U}$ to arrive at the expression

$$\frac{A'_{il}}{A'_{ii}} = \frac{\sum_{k=1}^{N_0} U'_{ik}U'_{lk}}{\sum_{k=1}^{N_0} U'_{ik}U'_{ik}} \tag{16}$$

with $\boldsymbol{U}'$ determined from the sample covariance matrix with the j-th column and j-th row removed. Fixing $\boldsymbol{W}$ and $\boldsymbol{\Sigma}^{0}$ to seemingly arbitrary values does not affect the result we are after. If l is the only unperturbed node besides i, then in the A' system l can now be treated as perturbed—as it may receive perturbations from the unobserved node j—and thus Eq. (14) applies. If l is part of many unperturbed nodes that are affected by j, then the knowledge how much each of these nodes contributes to the variance of the target node i (which is determined by $\boldsymbol{W}$ and $\boldsymbol{\Sigma}^{0}$) is irrelevant as we are only interested in the total effect of the alternative routes on node i. Using the inferred link strength from Eq. (16) we can rewrite Eq. (2) as a two-node residual inference problem between j and i, where we obtain a lower bound for link strength from node j to i by using the variation of i that could not be explained by $\boldsymbol{A}'$. This concept is similar to computing partial correlations. Defining by $\tilde{\boldsymbol{A}}$, $\tilde{\boldsymbol{B}}$ and $\tilde{\boldsymbol{D}}$ the 2×2 analogues to the full problem we obtain

$$\tilde{\boldsymbol{C}} = \tilde{\boldsymbol{A}}^{-1}\boldsymbol{B}\boldsymbol{B}^T\tilde{\boldsymbol{A}}^{-1} + \sigma^2\tilde{\boldsymbol{D}} \tag{17}$$

with $\tilde{\boldsymbol{C}}$ the covariance matrix of the vector $\tilde{\boldsymbol{y}}^{obs} = \left(y_j^{obs}, y_i^{obs} + \sum_{l\neq\{i,j\}} A'_{il}(A'_{ii})^{-1}y_l^{obs}\right)^T$ and $\tilde{D}_{11} = r_j$, $\tilde{D}_{22} = r_i + \sum_{l\neq\{i,j\}} A'^2_{il}A'^{-2}_{ii}r_l$, using the scaled variances $r_i = \sigma_i^2/\sigma^2$. Note that $A_{ii} < 0$ for all i as these elements represent sufficiently strong restoring forces that ensure negative definiteness of $\boldsymbol{A}$ and that we have $0 = A'_{ii}y_i^{obs} + \sum_{l\neq i} A'_{il}y_l^{obs}$ from Eq. (1) in the stationary case. An estimate for the minimum relative link strength from node j to node i can be calculated from Eq. (13) and is given by

$$\frac{\tilde{A}_{12}}{\tilde{A}_{11}} = \frac{\tilde{U}_{21}\tilde{D}_{22}^{-1/2}}{\tilde{U}_{11}\tilde{D}_{11}^{-1/2}} \tag{18}$$

Eq. (18) can be considered as an asymptotically unbiased response coefficient between node 1 as target node and node 2 as source node, as again any dependency on σ^2 has dropped out. An estimate for the maximum relative link strength from node j to node i follows from Eq. (18) with the off-diagonal elements of $\boldsymbol{A}'$ set to zero. We classify a link as non-inferable if there exists (i) a significant difference between the minimum und maximum estimated link strength and (ii) a minimum link strength that is not significantly different from noise.

Computational complexity of PRC. The computational cost for computing PRCs scales as $\mathcal{O}\left(N_{sub}^3\right)$, with N_{sub} the size of the subnetwork under consideration. However, as we infer directed networks, we first have to remove the perturbations on each target node before its incoming links can be inferred. The cycling through up to $N_{sub}-1$ perturbed target nodes increases the computational complexity to $\mathcal{O}\left(N_{sub}^4\right)$ in the worst case. As we have generated a subnetwork for perturbed node and used residual bootstrapping to infer statistically significant links, the total computational complexity is given by $\mathcal{O}\left(N_{boot}N_{per}\left\langle N_{sub}^4\right\rangle\right)$, where $\langle . \rangle$ denotes averaging over all subnetworks, N_{per} the number of perturbed nodes, and N_{boot} the number of bootstrap samples. If the travelling distance of perturbations (correlation length) in the network is significantly shorter than the network diameter, such that $N_{sub}/N \to 0$ in the limit of large networks, $N \to \infty$, the computational complexity scales linearly with network size. In contrast, using Lasso to infer directed links requires $\mathcal{O}\left(N_{boot}N_{sig}^4\right)$ operations, as the more efficient Graphical Lasso method[38] is only applicable to undirected networks. Whether our method is computationally

more efficient than Lasso depends on the inference problem. However, for the networks investigated in this work our method required significantly lesscomputational time than inference via Lasso using parallel computing.

Fraction of inferable links. Inferability of a directed link between source and target node requires that the remaining network may not contain the same information that is transmitted between them. A sufficient condition is that all information that the remaining network receives from the source node is destroyed by sufficiently strong perturbations. If the target node is not perturbed, information from the source node may reach the remaining network through the target node. In this case also the targets of the target node must be perturbed (Fig. 2a). Counting network motifs that satisfy these conditions gives the number of inferable links. If the network size, N, is significantly larger than the number of outgoing links for both the source and target nodes, we can approximate the fraction of inferable links, $F(q)$, by the expression (Supplementary Note 1)

$$F(q) \approx \sum_{k=1} \sum_{l=0} \sum_{m=0}^{\min(k-1,l)} \left[q^{k+1} + (1-q)q^{z}\right] P(k,l,m|k \to l)$$

Here, $P(k, l, m|k \to l)$ is the conditional probability of finding two connected nodes in the directed network, where the source node has $k \geq 1$ outgoing links, the target node has $l \geq 0$ outgoing links, and both share m nodes as common targets of their outgoing links. The first term in the brackets corresponds to the case that independent perturbation data for node B exists (Fig. 2a, left panel) and the second term to the case where independent perturbation data for node B are absent (Fig. 2a, right panel). In the calculation of $F(q)$ we assumed that the links in the network are represented by noiseless, linear functions with non-zero slope and that ensure that information of source nodes is neither destroyed nor absorbed in the process of transmission.

Data preparation. Kemmeren et al.[3] provided a transcriptome data set of a *Saccharomyces cerevisiae* genome-wide knockout library (with mutant strains isogenic to S288c). This data set comprises transcript levels of 6170 genes for 1484 deletion mutants. The data are presented as the logarithm of the fluorescence intensity ratios (M-values) of transcripts relative to their average abundance across a large number of wild-type replicates, resulting in logarithmic fold changes of mutant/wild-type gene expression levels compared with a wild-type reference level. Kemmeren et al. also used a dye swap setup for several experiments to average out the effect of a possible dye bias. Their chip design measures most of the genes twice per biological sample, thus allowing to estimate the technical variance. The preprocessing of the data is described in Kemmeren et al.[3], Supplementary Information.

Residual bootstrapping. We make use of the 748 measured wild-type experimental replicates to determine the natural variation among biological replicates, $\delta_{in} := \log_2(r_{in}) - \langle \log_2(r_{in}) \rangle_n$, with $r_{in} := x_{in}/x_i^{pool}$, x_{in} the expression of gene i in wild-type replicate n, x_i^{pool} the expression level of gene i measured after pooling over wild-type replicates, and $\langle . \rangle_n$ denoting the average over replicates. To generate the bootstrap samples we randomly select 200 different δ_{in} from the replicates for each gene i, and add these values to the log fold changes of the perturbed expression levels, $\langle \log_2(r_{in}^{pert}) \rangle_n$, with $r_{in}^{pert} := x_{in}^{pert}/x_i^{pool}$ and the average is taken over the two replicates for each knockout.

Sparsity constraints by removing noisy nodes. As network inference typically comes with an insufficient amount of independent perturbations and experimental replicates we run into the problem of overfitting the data. In this case, noisy information from many network nodes is collected to explain the response of a given target node. L^1-norm regularised regression (Lasso) systematically removes many links, where each link explains only a small part of the variation of the target node, in favour of few links, where each link contributes significantly. In our approach we remove noisy nodes and thus their potential outgoing links, where the critical noise level is determined by the variability among biological replicates. In the presence of noise, our algorithm removes weakly responding nodes from the network. We thereby assume that the existence of many indirect interactions between source and target node by first distributing the signal of the source node among many weakly responding nodes and then collecting these weak signals to generate a significantly responding target node is much less likely than the existence of a single direct interaction. However, in the noise-free case we run into the same problem as Lasso to determine the right cutoff (regularisation parameter).

Synthetic data. Synthetic data were generated using our own model and GeneNetWeaver[25]—an open access software that has been designed for benchmarking network inference methods. With GeneNetWeaver, networks were generated from a model that closely resembles the structure of the yeast regulatory network[25], and steady state levels of node activities were computed using ordinary differential equations. In our data generating model, we first generated scale-free networks with an exponent of 2.5 and an average degree of 2. Then, we solved a system of ordinary differential equations with non-linear regulatory interactions between nodes to obtain steady state values of node activities, e.g., transcript levels. For both models, logarithmic fold changes of node activities were calculated (transcriptional levels upon perturbation relative to wild levels), and gaussian noise was added.

Code availability. MATLAB and Python codes for the network inference algorithm and the data preprocessing steps are available on request.

Data availability. The data sets analysed during the current study are described in ref.[3] and are available from Gene Expression Omnibus https://www.ncbi.nlm.nih.gov/geo/ under the accession numbers GSE42527, GSE42526, GSE42215, GSE42217, GSE42241 and GSE42240.

Received: 11 November 2016 Accepted: 1 December 2017
Published online: 09 January 2018

References

1. Smanski, M. J. et al. Synthetic biology to access and expand nature's chemical diversity. *Nat. Rev. Microbiol.* **14**, 135–149 (2016).
2. Esvelt, K. M. & Wang, H. H. Genome-scale engineering for systems and synthetic biology. *Mol. Syst. Biol.* **9**, 641–641 (2014).
3. Kemmeren, P. et al. Large-scale genetic perturbations reveal regulatory networks and an abundance of gene-specific repressors. *Cell* **157**, 740–752 (2014).
4. Costanzo, M. et al. The genetic landscape of a cell. *Science* **327**, 425–431 (2010).
5. Wilhelm, M. et al. Mass-spectrometry-based draft of the human proteome. *Nature* **509**, 582–587 (2014).
6. Meinshausen, N. et al. Methods for causal inference from gene perturbation experiments and validation. *Proc. Natl Acad. Sci.* **113**, 7361–7368 (2016).
7. Kholodenko, B. N. et al. Untangling the wires: a strategy to trace functional interactions in signaling and gene networks. *Proc. Natl Acad. Sci.* **99**, 12841–12846 (2002).
8. F, F. & Q, Z. Learning sparse causal gaussian networks with experimental intervention: regularization and coordinate descent. *J. Am. Stat. Assoc.* **108**, 288–300 (2013).
9. Newman, M. E. J. *Networks* (Oxford University Press, 2010).
10. Menche, J. et al. Uncovering disease-disease relationships through the incomplete interactome. *Science* **347**, 1257601–1–1257601–7 (2015).
11. Marbach, D. et al. Wisdom of crowds for robust gene network inference. *Nat. Methods* **9**, 796–804 (2012).
12. Feizi, S., Marbach, D., Médard, M. & Kellis, M. Network deconvolution as a general method to distinguish direct dependencies in networks. *Nat. Biotechnol.* **31**, 726–733 (2013).
13. Barzel, B. & Barabási, A.-L. Network link prediction by global silencing of indirect correlations. *Nat. Biotechnol.* **31**, 720–725 (2013).
14. Weigt, M., White, R. A., Szurmant, H., Hoch, J. A. & Hwa, T. Identification of direct residue contacts in protein-protein interaction by message passing. *Proc. Natl Acad. Sci.* **106**, 67–72 (2009).
15. Bastiaens, P. et al. Silence on the relevant literature and errors in implementation. *Nat. Biotechnol.* **33**, 336–339 (2015).
16. Bansal, M., Belcastro, V., Ambesi-Impiombato, A. & di Bernardo, D. How to infer gene networks from expression profiles. *Mol. Syst. Biol.* **3**, 1–10 (2007).
17. Hill, S. M. et al. Inferring causal molecular networks: empirical assessment through a community-based effort. *Nat. Methods* **13**, 310–318 (2016).
18. Friedman, N., Linial, M., Nachman, I. & Pe'er, D. Using Bayesian networks to analyze expression data. *J. Comp. Biol.* **7**, 601–620 (2000).
19. Granger, C. W. J. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica* **37**, 424 (1969).
20. Schreiber, T. Measuring information transfer. *Phys. Rev. Lett.* **85**, 461–464 (2000).
21. Oates, C. & Mukherjee, S. Network inference and biological dynamics. *Ann. Appl. Stat.* **6**, 1209–1235 (2012).
22. Blomen, V. A. et al. Gene essentiality and synthetic lethality in haploid human cells. *Science* **350**, 1092–1096 (2015).
23. Costanzo, M. et al. A global genetic interaction network maps a wiring diagram of cellular function. *Science* **353**, 1381–1394 (2016).
24. Tipping, M. E. & Bishop, C. M. Probabilistic principal component analysis. *J. R. Stat. Soc.* **61**, 611–622 (1999).
25. Schaffter, T., Marbach, D. & Floreano, D. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics* **27**, 2263–2270 (2011).
26. Omranian, N., Eloundou-Mbebi, J. M. O., Mueller-Roeber, B. & Nikoloski, Z. Gene regulatory network inference using fused LASSO on multiple data sets. *Sci. Rep.* **6**, 1–14 (2016).
27. Alipanahi, B. & Frey, B. J. Network cleanup. *Nat. Biotechnol.* **31**, 714–715 (2013).

28. Friedman, J., Hastie, T. & Tibshirani, R. Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.* **33**, 1–19 (2010).
29. Al-Momani, M. Shrinkage and Penalty Estimation Strategies in Some Spatial Models. *Electronic Theses and Dissertations,* 1–239 (2013).
30. Marbach, D. et al. Revealing strengths and weaknesses of methods for gene network inference. *Proc. Natl Acad. Sci.* **107**, 6286–6291 (2010).
31. Prill, R. J. et al. Towards a rigorous assessment of systems biology models: the DREAM3 challenges. *PLoS ONE* **5**, e9202 (2010).
32. Ruan, J. A top-performing algorithm for the DREAM3 gene expression prediction challenge. *PLoS ONE* **5**, 1–8 (2010).
33. Stockwell, S. R., Landry, C. R. & Rifkin, S. A. The yeast galactose network as a quantitative model for cellular memory. *Mol. BioSyst.* **11**, 28–37 (2015).
34. Shalem, O. et al. Genome-scale crispr-cas9 knockout screening in human cells. *Science* **343**, 84–87 (2014).
35. Vogelstein, B. et al. Cancer genome landscapes. *Science* **339**, 1546–1558 (2013).
36. Williams, E. G. et al. Systems proteomics of liver mitochondria function. *Science* **352**, aad0189–1–aad0189–14 (2016).
37. Schäfer, J. & Strimmer, K. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Stat. Appl. Genet. Mol. Biol.* **4**, 1–32 (2005).
38. Friedman, J., Hastie, T. & Tibshirani, R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**, 432–441 (2008).
39. Siegenthaler, C. & Gunawan, R. Assessment of network inference methods: how to cope with an underdetermined problem. *PLoS ONE* **9**, e90481 (2014).

Acknowledgements

We acknowledge help from Patrick Kemmeren in interpreting the yeast transcriptome data, the High Performance Computing Plattform of the Heinrich-Heine University and DFG funding by SPP 1395 and the Cluster of Excellence in Plant Sciences (grant no. EXC 1028).

Author contributions

A.S.K. and M.K. conceived the PRC method. N.H. and M.K. developed the counting procedure for inferable links and the Inferability measure. C.F.B. and M.K. developed the clustering and subnetwork methods and analysed the transcriptome data. N.H. wrote Supplementary Note 1. M.K. wrote Supplementary Note 2. C.F.B. wrote Supplementary Note 3.

Additional information

Supplementary Information accompanies this paper at https://doi.org/10.1038/s41467-017-02489-x.

Competing interests: The authors declare no competing financial interests.

Reprints and permission information is available online at http://npg.nature.com/reprintsandpermissions/

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Supplementary Note 3

Input data to network inference algorithm

Saccharomyces cerevisiae genome-wide knockout library

Description of the data set

Kemmeren et al. [1] provided a transcriptome data set of a *Saccharomyces cerevisiae* genome-wide knockout library (with mutant strains isogenic to S288c [2]). This data set, hereafter simply referred to as the yeast deletome data, comprises of transcript levels of 6170 genes for 1484 deletion mutants (hereafter referred to as perturbation experiments). The data of 1441 of these perturbation experiments can be used for network inference as the transcript levels of the deleted genes was measured with the microarray chip design. The data is presented as the logarithm of the fluorescence intensity ratios of red (R) and green (G) labelled microarray targets, referred to as M-values: $M = \left(\frac{R}{G}\right)$. For the regular experimental setup, DNA from the deletion mutants was labelled red, and DNA that was pooled from several wild type batches was labelled green. The aim of pooling the wild type DNA was to average out biological fluctuations, so that this DNA pool could be used to define a reference gene expression level. This allows the interpretation of the M-values as logarithmic fold changes of mutant gene expression levels compared to a wild type reference. Kemmeren et al. also used a dye swap setup for several experiments to average out the effect of a possible dye bias. The chip design used by Kemmeren et al. measures each gene twice per biological sample, thus allowing estimation of the technical variance.

Preprocessing of data

We used the preprocessed data (M-values) provided by Kemmeren et al. for our analyses. The preprocessing steps are described elsewhere [1]. We re-arranged the original preprocessed data layout to make it compatible with our network inference algorithm. Specifically, M-values corresponding to dye-swap experiments were multiplied by -1 and only knock-out experiments corresponding to genes that were also included in the chip design were kept in the data set.

Distribution of biological and technical variance (figure 3a and figure 3b)

Since there is insufficient information about the biological processes leading to variation among biological replicates (biological noise), we assume that the process is equal for all genes. By bringing the wild type data to unit variance for each gene, we show that the variation among the wild type data logarithmic fold changes is

approximately Gaussian distributed (figure 3a). By taking the mean logarithmic fold change of each biological replicate (wild type data only) and then calculating the differences of the technical replicates to this mean, the distribution of the technical variance can be found (figure 3b).

Correlations among wild type experiment M-values (figure 3c)

The histograms shown in figure 3c were created from the upper triangular correlation matrix (without diagonal elements) of wild type experiment M-values. The inserted figure was created by breaking up correlations between genes via random shuffling of the experiments.

Simulated Gene Expression data

To estimate how well network inference works on data which is similar to the data set provided by Kemmeren et al. [1], the input data for the simulations was generated as described below. In short, steady state solutions of ordinary differential equations were used to generate absolute gene expression levels, which were then turned into logarithmic fold changes, to which measurement noise was added.

Using scale free networks and steady state solutions of ordinary differential equations to simulate absolute gene expression levels

We chose scale free network models with exponent 2.5 to simulate a comparable network structure to yeast; the average degree was set to 2. Of all links, 80 % were set as inhibiting. Non-linear regulatory interactions were simulated by solving ordinary differential equations. First, wild type expression levels were calculated by solving a system of equations of the form shown below. The Hill-coefficients h_{ki} were sampled from a uniform distribution between 1 and 2. The constant K_{ki} was set to 0.5. The linear degradation term λ_i was set to 2 to assure stability. The coefficients a_{ki} and b_{ki} were set to 1 and -1 respectively for inhibiting links and to 0 and 1 respectively for activating links.

$$\dot{y}_i = \sum_{k \in \{1, \ldots, N\}/i} \left(a_{\mathrm{ki}} + b_{\mathrm{ki}} \frac{y_k^{h_{\mathrm{ki}}}}{K_{\mathrm{ki}}^{h_{\mathrm{ki}}} + y_k^{h_{\mathrm{ki}}}} \right) + u_i - \lambda_i y_i$$

Here, u_i denotes the basal gene expression rate. To compute the response y_{ij} of a gene i to a knock-out of gene j, the above shown system of ODEs was modified by setting the basal expression rate for the knocked out gene to zero and by removing all links onto and away from the gene j.

Using GeneNetWeaver to simulate absolute gene expression data

For better comparability with other publications, gene expression data was also generated with the program "GeneNetWeaver" [3]. This data was only used for comparing inference methods using ROC curves. From the "gold standard" yeast network in the program, random subnetworks were extracted with at least 100 regulators (random vertex seed; greedy neighbor selection). Then, the "Generate Kinetic Model" option was used with removal of auto-regulatory interactions to allow the generation of data sets. Data sets were generated with the following options: deterministic (ODEs) model, knock-out & wild type experiments, no time series, no noise added.

M-values (Logarithmic fold changes)

To simulate logarithmic fold changes that were similar to the M-values (logarithmic fold changes) of the yeast deletome data set described above, the M-values of yeast knock-out mutants were investigated for the knocked out genes. Due to measurement noise, these values never reach negative infinity as would be expected in the absence of technical noise. The median minimal absolute fluorescence intensity averaged over all knocked out genes was estimated to be $2^{-2.5}$. This value was added to all simulated absolute gene expression level values before calculating the logarithmic fold changes. That is, the gene expression response of a gene i towards a perturbation of a gene j, expressed as logarithmic fold change, was calculated as $M_{ji} = log_2\left(\frac{y_{ji}+2^{-2.5}}{y_i+2^{-2.5}}\right)$, with y_{ji} the absolute expression level of gene i when gene j is perturbed, and y_i the expression level of gene i for the non-perturbed (wild type) state, as defined above. Replicates were produced through duplication of the logarithmic fold changes for each perturbation experiment.

Simulating measurement noise

Although the reference node activities in the data set provided by Kemmeren et al. [1] are correlated, we did not simulate correlated reference node activities. This is because we were lacking information about the processes that generate biological variance as well as the actual network structure of *S. cerevisiae*. Because the biological noise dominated the technical noise, we did not distinguish both noise types in our simulations. Rather, we simulated noise by adding gaussian distributed random numbers to all simulated M-values.

DREAM3 in silico challenge data

As explained in more detail elsewhere [4], participants were given 3 data sets called "null-mutant", "heterozygous" and "trajectories" for each network. The "null-mutant" and "heterozygous" data sets contained wild type gene expression levels as well as the steady state gene expression levels of single gene knock-outs for all genes. In both cases, it was indicated which data belonged to which knock-out (perturbation target). The "trajectories" data sets contained several time-series of gene expression but did not contain information about the perturbation target. We used the "null-mutant" data sets as well as the near-stationary part ($t \geq 170$) of the trajectories data. The former was used to center the gene expression levels (node activities) and to calculate the covariance matrix. The latter was used to estimate the overall noise level.

Workflow

In the following, work flow steps are listed in the order they were possibly used. Depending on the analysis, not all of the steps were used. For example, for the analysis of DREAM3 data, the steps for clustering and for the identification of significant links were skipped.

Data centering

For all analyses (yeast deletome, simulations & DREAM3 challenge), the node activity data was centered to the average of the corresponding reference node activities (not to the average of all node activities).

Identification of significantly affected nodes

To identify which nodes were significantly affected by a perturbation, it was checked whether the node activities were significantly different from the background noise. This was done by comparing node activities magnitude of perturbation experiments to the corresponding reference node activity variance using two-tailed t-tests (yeast deletome data & simulated data) or two-tailed z-tests (DREAM3 challenge data).

Clustering

Nodes were grouped into clusters if they were not sufficiently linearly independent. The following steps were followed:

1. All node activities were normalized to unit reference node activity variance.

2. A $(P \times N)$ node activity matrix was formed by merging experimental replicates (here, P denotes the number of experiments and N denotes the number of genes). Replicates (biological and technical) were merged by averaging and multiplication by the square root of the number of biological replicates.

3. For each pair of nodes k and l, a test statistic T_{kl} was compared to a 5 % FDR cutoff. The test statistic was calculated as the square of the smaller singular value of the $P \times 2$ matrix consisting of those two columns of the node activity matrix which corresponded to the nodes k and l. For simplicity, we assumed that the reference node activities were Gaussian distributed and that the node interactions were sufficiently linear. Then, under the Null Hypothesis of linear dependence, the test statistic follows a distribution which we approximated with a Chi-square distribution with P degrees of freedom (the expected value of the $\mathcal{X}^2_P$ distribution over-estimates the expected value of the small eigenvalue, but this is negligible if sufficiently high confidence is claimed to reject the null hypothesis). An FDR of 5 % was chosen to define a significance cutoff.

4. Clusters were finally formed by grouping unperturbed nodes with perturbed nodes on which they were linearly dependent. If an unperturbed node was linearly dependent on multiple perturbed nodes, this node was grouped with the perturbed node onto which it was the most linearly dependent. Remaining unperturbed nodes were grouped with other unperturbed nodes if they were not sufficiently linearly independent.

5. For each cluster, either the perturbed node, or, if no perturbed node existed in the cluster, the node with the largest signal-to-noise ratio was chosen to be the cluster representative, and the corresponding node activity data was used for further analysis.

Links between clusters were found by inferring links between cluster representatives. Links within clusters can only be inferred within two-node clusters if there is exactly one perturbed node and if the unperturbed node is the only node affected by the perturbed node. Note that all perturbation experiment node activities were used to create the $(P \times N)$ node activity matrix and that no bootstrapping was done.

Subnetwork method

To infer a link from a node j onto a target node i using the subnetwork method, the network is temporarily limited to only those nodes that are significantly affected

when node j is perturbed (the cutoffs used here were the same ones that were used for identifying the significantly affected nodes). This means that the inference of several links is skipped, and the corresponding link strengths are set to zero and are not included in the calculation of the FDR to determine a cutoff for link significance.

Identification of significant links

Partial Response Coefficients (PRCs) were calculated for the respective subnetworks. Then, inferable links were tested for significance using z-tests. The link strength's variances were estimated by bootstrapping over the node activity data (described below). The false discovery rate (FDR) was adjusted to 5 % to account for multiple hypothesis testing [5].

Bootstrapping

Residual bootstrapping was used (this was done to get a speed-up compared to parametric bootstrapping, in which residuals are drawn de novo from a distribution rather than used multiple times but in randomized order). First, residuals were calculated from the reference node activity data and then added to the average node activity values of each perturbation experiment. Depending on the analysis, we followed one of two separate approaches for calculating the residuals from the data for reasons described in the following.
First, because we had found that wild type experiment logarithmic fold changes are highly correlated in the yeast deletome data, we sought to maintain the correlations among the reference node activity residuals to improve inference performance.
Second, because we did not simulate correlated measurement noise in our simulations, the effect of random correlations among the reference node activities resulting from limited sample sizes was sought to be minimized. This was achieved by shuffling the the order of reference experiments among the nodes.
Both procedures are explained in the following section.

Residual bootstrapping procedure illustrated

Both bootstrapping approaches are illustrated in the following example, in which it is described how residual bootstrapping is performed on the node activity data for one perturbation experiment. (Residual bootstrapping of reference node activity data is analogous). The node activity data for N reference experiments of a two-node network with nodes i and j is represented as column vectors $\mathbf{x}_i^{WT} = \left(x_{i,1}^{WT}, \ldots, x_{i,N}^{WT}\right)^T$ and $\mathbf{x}_j^{WT} = \left(x_{j,1}^{WT}, \ldots, x_{j,N}^{WT}\right)^T$, and node activity data for a single perturbation experiment with two replicates is represented as $\mathbf{x}_i^{KO} =$

$\left(x_{i,1}^{KO},\ x_{i,2}^{KO}\right)^T$and $\mathbf{x}_j^{KO} = \left(x_{j,1}^{KO},\ x_{j,2}^{KO}\right)^T$. The residuals for the nodes are calculated to be $\mathbf{r}_i=(r_{i,1},\ldots,r_{i,N})^T = \bar{x}_i^{WT} - \mathbf{x}_i^{WT}$ and $\mathbf{r}_j=(r_{j,1},\ldots,r_{j,N})^T = \bar{x}_j^{WT} - \mathbf{x}_j^{WT}$, with $\bar{x}_j^{WT}$ and $\bar{x}_i^{WT}$ the mean of the corresponding reference node activities. To leave correlations among reference node activities intact, we have the following expression for an arbitrary bootstrap sample for the perturbation node activities: $\mathbf{x}_i^{KO,boot} = \left(\bar{x}_i^{KO}, \bar{x}_i^{KO}\right)^T - (r_{i,p}, r_{i,q})^T$ and $\mathbf{x}_j^{KO,boot} = \left(\bar{x}_j^{KO},\ \bar{x}_j^{KO}\right)^T - (r_{j,p}, r_{j,q})^T$, with p and q being random (possibly equal) integers between 1 and N, and $\bar{x}_i^{KO}$and $\bar{x}_j^{KO}$the average node activities of the perturbation experiment. To destroy correlations among reference node activity residuals, the residuals are shuffled randomly across bootstrap samples. We have the following expression for an arbitrary bootstrap sample for the perturbation node activities: $\mathbf{x}_i^{KO,boot} = \left(\bar{x}_i^{KO}, \bar{x}_i^{KO}\right)^T - (r_{i,p}, r_{i,q})^T$ and $\mathbf{x}_j^{KO,boot} = \left(\bar{x}_j^{KO},\ \bar{x}_j^{KO}\right)^T - (r_{j,u}, r_{j,v})^T$, with $p,\ q, u$ and v being random (possibly equal) integers between 1 and N.

Applications

Analysis of the yeast deletome data

Work flow steps for the Inference of the GAL network

The network was limited to all genes that could in principle be affected by perturbations of the following nodes: GAL3, GAL4, GAL80 and MIG1. A false discovery rate of 10^{-3} was used as a threshold for the identification of significantly affected nodes. With this threshold, 44 nodes remained in the network, that were then clustered. Then, subnetworks were determined, followed by inference of links between clusters. Only clusters that contained at least one of the above mentioned nodes were kept for the graph shown in figure 4d.

Note on the identification of significantly affected nodes

Because we had found that, among the node activities, the biological variance was larger than the technical noise, only the number of biological replicates was used to estimate the degrees of freedom. To account for the multiple hypothesis testing, we adjusted the overall false detection rate (FDR) [5]. For the analysis of the *Saccharomyces cerevisiae* genome-wide knockout library, we followed the recommendation by Kemmeren et al. [1] (supplementing information) to exclude several genes from further analysis. Furthermore, we excluded all Pseudogenes and dubious ORFs listed on yeastgenome.org [6]. Because we had found that the distribution of normalized wild type experiment logarithmic fold changes can be well described by a t-distribution with approximately 11 degrees of freedom,

this number was used as the degrees of freedom for the t-tests instead of the number of biological wild type experiments. Although the microarray design used by Kemmeren et al. [1] could be used to disentangle technical and biological noise, the biological noise clearly dominated over the technical noise. Hence, we did not disentangle the variances for our analyses but rather used the overall variance. Nodes that were significantly affected in at least one perturbation experiment were kept in the network; all other nodes were removed from further analysis.

Lists of inferred links (Supplementary data 1 and 2)

To generate a list of the most significant links inferred from the whole yeast deletome data, a consensus list was created that reflects the links inferred from the data by using two different methods of bootstrapping (see also the section on bootstrapping). The reason for this is that we are lacking a sophisticated model describing the process that generates biological noise. A consensus list should thus represent a more reliable model for link inference, as only the most significant links that were found by all methods survive the selection process. We created two different consensus lists (suppl_data1_consensus_stringent.xlsx and suppl_data2_consensus_moderate.xlsx), each corresponding to a certain cutoff to select significantly affected nodes. That is, for the stringent cutoff, the network size was smaller than for the moderate cutoff because less nodes were found to be significantly affected by perturbations. The cutoffs correspond to false discovery rates of 10^{-8} and 10^{-6}. The lists were merged by keeping only links that appeared in both lists and by then keeping the (at most 500) most significant links with the highest average Z-score. Gene descriptions from the Saccharomyces Genome Database [6] were added to the list to allow easier investigation. We found that, among the most significant links, there are links whose source and target nodes are either adjacent on the genome or which are paralogs. These links are most likely false positives that reflect inspecificities of the gene deletion process used to generate the yeast deletion collection [2].

List of inferred hub nodes (Supplementary data 3)

From the consensus link list with the stringent cutoff for the selection of significantly affected nodes, we selected the 10 nodes with the highest number of significant outgoing links (significance cutoff: 5 % false discovery rate). The mean and median link strengths of the outgoing links were calculated for each of these 10 hub nodes. Gene descriptions from the Saccharomyces Genome Database [6] were added.

Simulations

Effect of noise-induced bias on relative link strength (figure 3d)

To show that network inference methods are biased towards measurement noise, we simulated node activity data for the case of infinitely many replicates. In that case, all parameters are estimated perfectly because uncertainty resulting from a limited number of observations are averaged out. The covariance matrix of node activities was calculated according to the following formula:

$$\mathrm{C}_{\mathrm{exact}} = \left(A^T\right)^{-1} \left(B^T B\right) \left(A\right)^{-1} + \sigma^2 I_N.$$

Where B is a diagonal matrix with element $b_{\mathrm{ii}} = 1$ if node i is perturbed. The signal-to-noise ratio was adjusted by changing the value of σ^2.

Comparison of simple, clustering and subnetwork methods (figure 4c)

The Inferability curves represent averages over 4 different networks and 12 perturbation samples. The network size was 60 nodes. Gene expression data was simulated from scale free networks and a non-linear model (as described below).

Receiver Operating Characteristic (ROC) curves (figure 4b)

Simulation setup The ROC curves were averaged (as described in the paragraph below) over 24 perturbation samples and 4 network structures. The network size was 300 nodes. In each perturbation sample, 25 % of nodes were randomly selected and perturbed individually. The noise-to-signal ratio was adjusted to 10 %. Data was generated with both GeneNetWeaver [3] and scale-free networks with a non-linear model to generate stationary data. For each perturbation sample, only data corresponding to significantly perturbed nodes was used to calculate the covariance matrices (1 % FDR), the only exception being Lasso regression applied to the data of all nodes. All methods (except Lasso applied to the data of all nodes) received almost the same covariance matrix C as input: because of the randomness of bootstrap sampling, small numerical differences between the covariance matrices that PRC and the other methods received may have occurred. Binary classification of inferred links was done solely based on links pointing away from perturbed nodes because only those links are potentially inferable. This is similar to the method described by Siegenthaler and Gunawan [7]. To infer a link onto a certain node, the data corresponding to a perturbation of that node was removed prior to calculating the covariance matrix. Because both Lasso regression and the subnetwork method set certain link strengths to exactly zero, it would be impossible obtain a ROC curve that smoothly approaches a false positive rate of 1. This is why we set all link test statistics that were equal to zero to random values smaller than the

smallest non-zero link test statistic. When creating the ROC curve, this procedure corresponds to random guessing for all links that were set to exactly zero.

ROC curve averaging In the following, the false positive rate is denoted by FPR and the true positive rate is denoted by TPR. To generate an averaged ROC curve, the tuples (FPR, TPR) of the individual ROC curves from each inferred network were assigned to 50 bins according to the FPR of each tuple. Bins were filled up in a way such that approximately equally many tuples were assigned to all bins. Then, for each bin, the FPR and TPR was calculated. These 50 averaged tuples (FPR, TPR) were used for the averaged ROC curve.

ROC curves for Subnetwork method and Lasso We generated 10 different ROC curves for the subnetwork method, each curve corresponding to a different cutoff for the selection of significantly affected nodes (based on which the subnetworks were created). For Lasso, we generated ROC curves for 10 different regularization coefficients. To model a smooth transition between the cutoffs or regularization coefficients, we interpolated (cubic spline) between the ROC curves in a way such that the resulting hybrid curve had a possibly larger area under the curve than the individual ROC curves. The 10 different cutoffs for selection of significant nodes correspond to the following FDRs: 1.00, 0.89, 0.78, 0.67, 0.56, 0.45, 0.34, 0.23, 0.12, and 0.01. The 10 different regularization coefficients were: 0, 0.001, 0.0018, 0.0032, 0.0056, 0.010, 0.0178, 0.0316, 0.0562, and 0.1000.

Comparison of non-regularized inference methods Test-statistics T_{ji} corresponding to a link from a node j to a node i were calculated according to the following formulas. The standard deviations of the test statistics were estimated via residual bootstrap.

1. Regression: $T_{\mathrm{ji}} = \frac{|G_{\mathrm{ji}}|}{\mathrm{std}(G_{\mathrm{ji}})}$, where $G_{\mathrm{ji}} = \frac{(C)^{-1}_{\mathrm{ji}}}{(C)^{-1}_{\mathrm{ii}}}$.

2. Conditional Mutual Information: $T_{\mathrm{ji}} = \frac{|G_{\mathrm{ji}}|}{\mathrm{std}(G_{\mathrm{ji}})}$, with $G_{\mathrm{ji}} = \log_2\left(\frac{\det(A) \times \det(B)}{\det(C) \times \det(D)}\right)$. Here, A, B and D are covariance matrices with rows and columns of certain nodes removed. A: node j removed. B: node i removed. D: nodes i and j removed.

3. PRC: $T_{\mathrm{ji}} = \frac{|G_{\mathrm{ji}}|}{\mathrm{std}(G_{\mathrm{ji}})}$, where $G_{\mathrm{ji}} = \frac{\sum_{k=1}^{N_0} U'_{\mathrm{ik}} U'_{\mathrm{jk}}}{\sum_{k=1}^{N_0} U'_{\mathrm{ik}} U'_{\mathrm{ik}}}$, where N_0 is the number of unperturbed nodes and U_i is the i^{th} eigenvector of the covariance matrix. Note that links were not identified as artifacts.

Comparison of regularized inference methods For a fair comparison, we used 10 different regularization coefficients to generate ROC curves for Lasso regression, keeping only the ROC curve corresponding to the regularization coefficient that gave the best performance. The following is a brief derivation of the formulas used for regularized regression using the L_1 and L_2 norms. The simulation setup is described afterwards. Consider the following equation, which is essentially equation 1 of the main paper except that perturbations are not restricted to single nodes, which is expressed through the perturbation strength matrix U:

$$\dot{Y} = YA^T + U$$

$$Y^{\text{obs}} = Y + \epsilon$$

For simplicity, we assumed here that the reference state is 0. Here, $Y$are the true node activities that are masked by measurement noise ϵ to yield observed node activities Y^{obs}, and the matrix A^Tdenotes the link strengths. In the steady state and under the assumption that the measurement uncertainties ϵ are Gaussian distributed, a maximum a posteriori (MAP) function of the parameters A^T and U given the observed data Y^{obs} can be defined. The logarithm of this function is:

$$a\left(A^T, U|Y^{\text{obs}}\right) = K + \sum_{i=1}^{N}\sum_{n\in S_i}\left[\left(\sum_{j=1}^{N} Y_{\text{nj}}^{\text{obs}} A_{\text{ji}}^T\right) + U_{\text{ni}}\right]^2 \frac{1}{\sigma_{\text{ii}}^2} \quad (1)$$

$$+ r^2 \sum_{i}^{N}\left(A_{\text{ii}}^T + \mu_i\right)^2 + v^2 \sum_{i}^{N}\sum_{n\in T} U_{\text{ni}}^2 \quad (2)$$

Here, K is a normalization constant from the distributions that vanishes upon differentiation. The experiment indices n run over the set of all experiments except the ones in which node i is perturbed, S_i. N is the number of nodes; P is the number of experiments; $Y_{\text{nj}}^{\text{obs}}$ is the matrix of node activities (each row corresponds to a sample and each column corresponds to a node); σ_{ii}^2 is a variance coefficient and r^2, v^2 are regularization coefficients that correspond to weights of the prior distributions. Note that the first prior distribution is only defined over the diagonal elements of the network matrix A. It can be shown that the following estimator maximizes the MAP function:

$$\left(\hat{A}^T\right)_{\text{ji}} = \frac{\left(\Phi^T\Phi\right)_{\text{ji}}^{-1}}{\left(\Phi^T\Phi\right)_{\text{ii}}^{-1}}\mu_i.$$

Here, Φ denotes the observed node activities with samples corresponding to a perturbation of node i removed (the number of rows of Φ is less than or equal to the number of rows of Y^{obs}, depending on whether node i is perturbed). This means that the data may need to be re-organized to infer links onto each node.

To arrive at this result, one needs to maximize $a\left(A^T, U|Y^{\text{obs}}\right)$ with respect to A. This is possible because the diagonal elements of the network matrix A are regularized, which does not allow for arbitrary combinations of either A or U. The following steps lead to the above stated solution:

1. Set the derivative of the log-MAP function w.r.t. U_{ni} to zero: $\frac{\delta a\left(A^T, U|Y^{\text{obs}}\right)}{\delta U_{\text{ni}}} = 0$.

2. Solve for U_{ni} to obtain the MAP estimator $\hat{U}_{\text{ni}}$. Plug this expression into formula for log-MAP function.

3. Set this new expression of the log-MAP function w.r.t. $A^T_{\text{ji}}$to zero: $\frac{\delta a\left(A^T, U|Y^{\text{obs}}\right)}{\delta A^T_{\text{ji}}} = 0$.

4. Solve for A^T_{ji} to obtain the MAP estimator$\hat{A}^T_{\text{ji}}$. To achieve this, one has to apply the Sherman-Morrison-Woodbury formula and let the regularization coefficient r^2 go to infinity, which essentially sets all A^T_{ii} equal μ_i.

Note that, for $\mu_i = -1$, the set of parameters A^T_{ji} (except for A^T_{ii}, which become equal to μ_i) are simply the ordinary least squares (OLS) estimators that minimize the following expression:

$$F = \|\Omega\beta - \Phi_i\|_2^2$$

Here, Ω is a matrix that is equal to Φ except that it is missing column i, and β is a vector that is equal to A^T_i except that it is missing the element equal to A^T_{ii}. The L_1 Norm ("Lasso" regularization) and L_2 Norm ("Tikhonov-Miller" regularization) impose further constraints on this extreme value problem. The corresponding formulas are:

$$F_{L1} = \|\Omega\beta - \Phi_i\|_2^2 \; + \; \gamma_{L_1} \left\|A^T_i\right\|_1$$

and

$$F_{L2} = \|\Omega\beta - \Phi_i\|_2^2 \; + \; \gamma_{L_2} \left\|A^T_i\right\|_2^2$$

for the L_1 and L_2 Norm, respectively.

DREAM3 in silico challenge

Brief description of the challenge

In the DREAM3 in silico challenge, participants were given the task to infer causal interactions from simulated gene expression data. The challenge was split into 3 sub-challenges, corresponding to the inference of networks with 10, 50 and 100

nodes. In each sub-challenge, 5 networks had to be inferred. Each of these networks had been created by sampling from the set of already known gene regulatory interactions from either *E. coli* or yeast using the tool GeneNetWeaver [3], which had also been used to simulate gene expression data. For each of the 5 networks, the p-values of AUROC and AUPR values had to be estimated, yielding 10 p-values for each sub-challenge. Then, the average of the negative logarithms (base 10) of the p-values was used as the overall score for each sub-challenge, which resulted in one score each for the 10, 50 and 100 node network sub-challenge.

Work flow steps to obtain Partial Response Coefficients (PRC)

After centering the node activity data, significantly affected nodes were determined to obtain subnetworks. The significance level to determine which nodes were significantly affected by a perturbation was adjusted to the network size via Bonferroni correction.
Due to a lack of experimental replicates in the data, we did not perform residual bootstrapping; instead, the raw partial response coefficients (PRC) were used for the inferred network. Furthermore, significant links did not need to be identified since only AUROC and AUPR values had to be calculated.

Calculation of DREAM3 overall scores

We followed the original procedure of the DREAM3 organizers [4], with one modification. The original procedure assumes that a significance measure has been assigned to each inferred link. Because our subnetwork method sets some PRCs to exactly zero (insignificant links), the resulting AUROC and AUPR curves would not be as smooth; furthermore, we observed that the distribution of background AUROC and AUPR values became multimodal for the 10-node network. To circumvent this, we first took the absolute value of all PRCs to obtain a link significance measure. Then, we set all zero PRCs to values smaller than the smallest non-zero link significance measure. Unfortunately, this introduces a randomness in the inferred network that can substantially influence the AUROC value. This is because the AUROC measure is sensitive to links that purely reflect noise. We averaged out this randomness by sampling over the random values that were used to replace the zeros. To obtain p-values for AUROC and AUPR values, we followed the original procedure as correctly described in [8], and fitted parametric probability distributions to 100,000 background AUROC and AUPR values. Those background AUROC and AUPR values were obtained by comparing the inferred network to randomly shuffled versions of the reference (gold standard) networks. The DREAM3 organizers used two distribution functions to fit the left and right side of the unimodal background AUROC and AUPR distributions separately. We

also investigated the effect of using a t-distribution to fit the AUROC background values, with which we obtained similar p-values.

Supplementary data 4: DREAM3 in silico network inference challenge results

The table (suppl_data4_dream3_scores.xlsx) contains the overall scores as well as AUROC and AUPR values for all networks of each sub-challenge. The first column shows the significance level for significantly affected nodes before Bonferroni-correction. The second column shows the scores when the original method [8] was used for fitting distributions to background AUROC and AUPR values. The third column shows the scores when the original method [8] was used for fitting distributions to the background AUPR values but a t-distribution was used to fit the background AUROC values.

Remarks on Table 1

Table 1 shows the DREAM3 scores from Supplementary data 4 that correspond to a significance level of $\alpha = 0.05$ for the identification of significantly affected nodes (before Bonferroni correction). It includes both the scores that were calculated with the original method as well as with a t-distribution to fit the AUROC background values (described above).
For the 100 node network, it could not be determined whether our approach scored 1st or 2nd. The reason for this is that the score for the highest performing participant in the DREAM3 network inference challenge was published as infinity, as "the p-value for this performance was below the precision" of the calculation of the DREAM3 organizers [4].

Software

For the analyses of the yeast deletome and simulations, we used MATLAB and Statistics Toolbox and Parallel Computing Toolbox Release R2014b, The MathWorks, Inc., Natick, Massachusetts, United States. For the analysis of the DREAM3 challenge data, we used Python 2.7 with numpy 1.11.0, scipy 0.17.0 and mpmath 0.19.

References

[1] Kemmeren, P. *et al.* Large-Scale Genetic Perturbations Reveal Regulatory Networks and an Abundance of Gene-Specific Repressors. *Cell* **157**, 740–752 (2014).

[2] Giaever, G. *et al.* Functional profiling of the Saccharomyces cerevisiae genome. *Nature* **418**, 387–391 (2002).

[3] Schaffter, T., Marbach, D. & Floreano, D. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics* **27**, 2263–2270 (2011).

[4] Prill, R. J. *et al.* Towards a rigorous assessment of systems biology models: The DREAM3 challenges. *PLOS ONE* **5**, e9202 (2010).

[5] Benjamini, Y. & Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc.* 289–300 (1995).

[6] Cherry, J. M. *et al.* Saccharomyces Genome Database: the genomics resource of budding yeast. *Nucleic Acids Res.* **40**, D700–D705 (2012).

[7] Siegenthaler, C. & Gunawan, R. Assessment of Network Inference Methods: How to Cope with an Underdetermined Problem. *PLOS ONE* **9**, e90481 (2014).

[8] Stolovitzky, G., Prill, R. J. & Califano, A. Lessons from the DREAM2 challenges - a community effort to assess biological network inference. *Ann. N.Y. Acad. Sci.* **1158**, 159–195 (2009).

1.3 Appendix: On the statistics of a gene clustering method for network compression

Biological motivation

The greater the number of nodes in a network that are individually perturbed, the more links can be inferred (see 1.2). Some unperturbed nodes receive information from only one other node; that is, their node activities can be expressed as a function of the node activities of a single other node. If this is the case, the unperturbed nodes can be thought of as informational clones of this other node (figure 8). This can hinder network inference because it becomes impossible to determine if information comes from a perturbed node or one of its clones. However, merging nodes that carry the same information into clusters enables inference of links between clusters.

Commonly used clustering methods such as k-means [68] do not distinguish between perturbed and unperturbed nodes and have other objectives besides merging nodes with identical information. Therefore, they cannot be used for increasing the ratio of perturbed to unperturbed nodes. The clustering method introduced in section 1.2 identifies informational clones based on the linear dependence of associated node activity data. This method relies on the χ^2 distribution as an approximation to test when nodes should be clustered given noisy data. In the following, the rationale behind the clustering method is described in more detail. First, the noise-free case is described. Then, the noisy case is introduced to motivate the need for statistical testing. In the remaining part of this section, it is analyzed when the χ^2 distribution is a reasonable approximation.

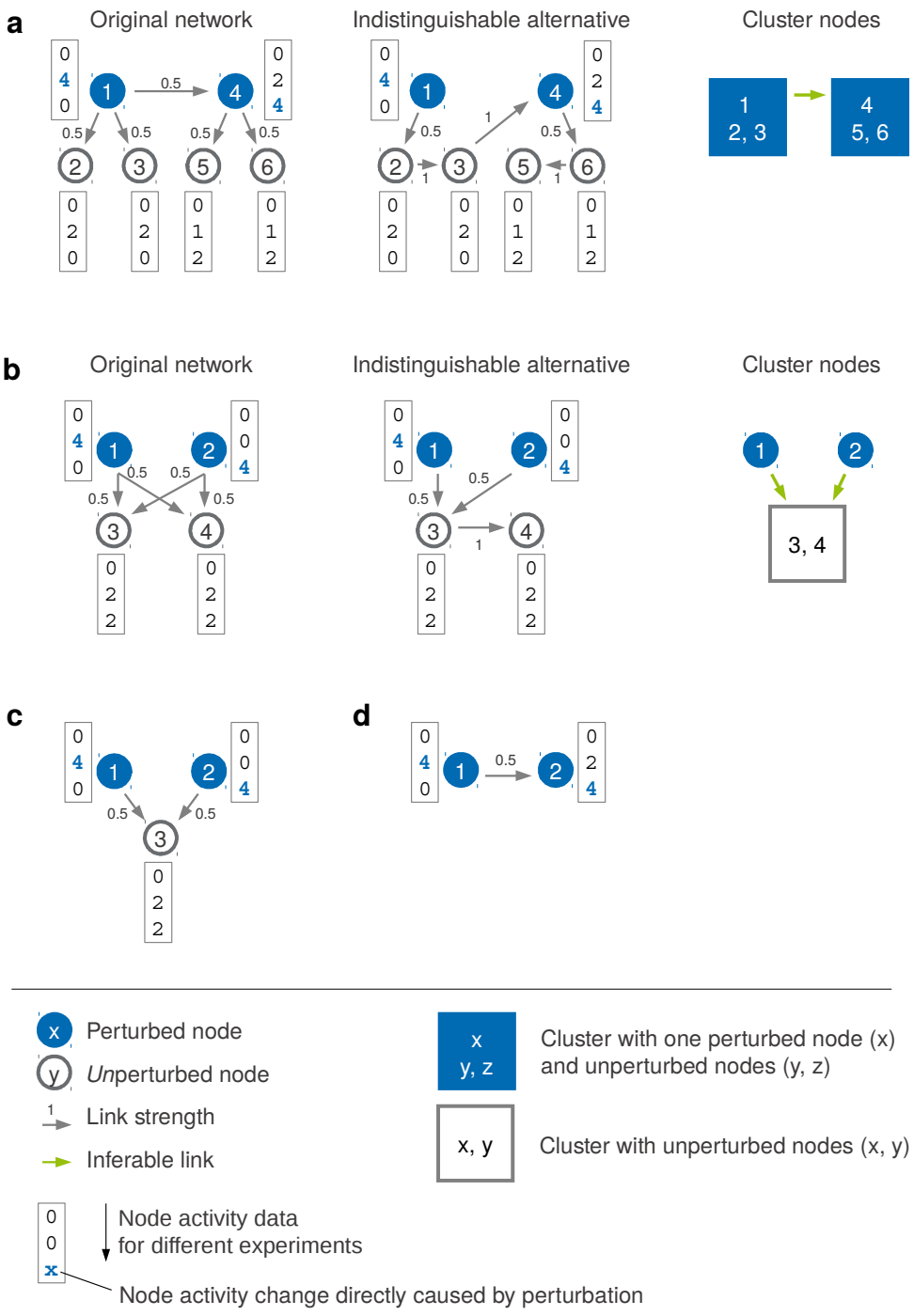


Figure 8: Illustration of a network inference problem and relevance of gene clustering. **a:** Nodes 2 and 3 receive information (the effect of perturbations) only from node 1, making them informational clones of node 1. Accordingly, nodes 5 and 6 are informational clones of node 4. No links can be inferred in this network because there are alternative structures that can explain the data. Forming node clusters enables inference of links between clusters. **b:** Clustering of unperturbed nodes can improve network inference as well. **c:** The unperturbed node 3 is not an informational clone of either node 1 or node 2 because its node activities cannot be expressed in terms of the node activities of just node 1 or node 2. In this case, node 3 cannot be assigned to a cluster. **d:** Both nodes are perturbed and thus receive independent information; they cannot be informational clones and cannot be clustered.

Mathematical motivation

It is assumed here that the links between nodes in a network can be described by linear functions. In the following, X denotes an $n \times 2$ matrix of node activities of two genes of interest, with element X_{ki} denoting the node activity of node i in experiment k relative to a reference state*, and $n > 2$. If measurement noise is absent, linear dependence can be established based on if and how the following equation is fulfilled (X_1 and X_2 are the first and second columns of X, and a is a real number):

$$X_1 = aX_2 \tag{3}$$

Three general cases be distinguished:

i Equation 3 can not be fulfilled for any a. The nodes receive information from independent sources, which can either be a perturbation or information from a different perturbed node (figure 9 a, b and c). In this case, the nodes should not be clustered.

ii There is an $a \neq 0$ such that eq. 3 is fulfilled. In this case, the nodes are informational clones (figure 9 d, e) and should be clustered.

iii Equation 3 can be fulfilled, but only if $a = 0$. Either both nodes are not perturbed or one node is perturbed but there is no link to the other node (figure 9 f, g, respectively). Either way, no link can possibly be inferred between the nodes, and the nodes should be removed from the network.

Cases (i) and (ii) can be distinguished based on the rank, or, equivalently, on the eigenvalues of matrix $C = X^T X$: if and only if one eigenvalue of C is equal to zero, the node activities are non-trivially linearly dependent (accordingly, [109, p. 7, p. 33, p. 34]) and the nodes are informational clones.

Measurement noise, however, complicates the detection of each case. In section 1.2, it was shown that the reference node activities derived from microarray data were approximately Gaussian distributed. This suggests that experiment node activities can be modeled as being corrupted by additive Gaussian noise. That is, it can be assumed that the observed experiment node activities X'_{ki} are equal to the actual node activities X_{ki} plus i.i.d. Gaussian noise ϵ_{ki}, i.e. $X' = X + \epsilon$. The eigenvalues of matrix $C' = X'^T X'$ then follow probability distributions and the small† eigenvalue is greater than zero even if two nodes are linearly dependent. Linear dependence must hence be dealt with probabilistically.

In the following, the distribution of the small eigenvalue of matrix $C' = (cX + \epsilon)^T (cX + \epsilon)$ is investigated. Here, c is a scaling factor, the elements of ϵ are i.i.d. $\mathcal{N}(0, 1)$ Gaussian distributed, and the

*As the node activities are given with respect to a reference state, unaffected nodes an activity of zero.
†When there are only two nodes, there are only two eigenvalues: the small and the large eigenvalue.

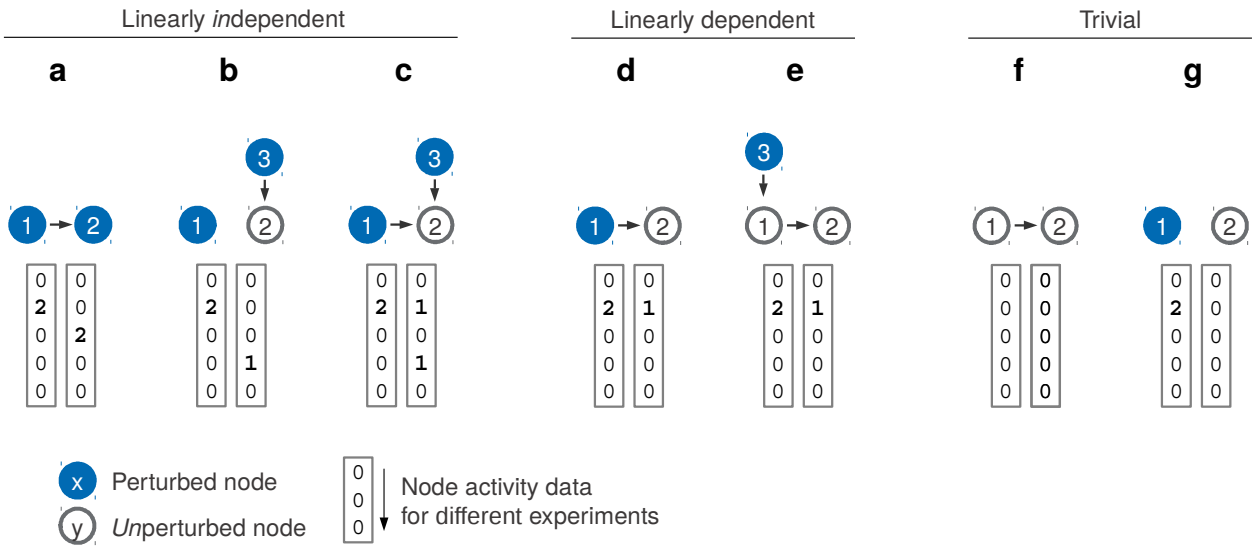


Figure 9: Illustration of assessment of linear dependence (see main text for a description).

columns of X are linearly dependent by a factor γ, that is, X has the following form:

$$X = \begin{bmatrix} X_{11} & \gamma X_{11} \\ X_{21} & \gamma X_{21} \\ \ddots & \ddots \\ X_{n1} & \gamma X_{n1} \end{bmatrix} \tag{4}$$

In the following, it is assumed that $\gamma \approx 1$. The scaling factor c can be interpreted as the signal strength affecting the node activities. Two extreme cases were considered: $c \to \infty$ and $c = 0$. The former case was investigated using simulations only. The latter case was investigated both analytically and with simulations. For $c = 0$, X vanishes. The elements of C' are then only determined by noise and follow a Wishart distribution[110]. The cumulative distribution function (cdf) of the smallest eigenvalue of such a matrix has been derived before[111], but the involved integrals are complicated and difficult to solve numerically.

Results

Note: two choices of X were investigated, both with comparable results. In one case, the first column of X was filled with ones and in the second case, the first column contained Gaussian distributed random numbers. Figures 10 and 11 were generated based on the latter choice of matrix X.

The distribution of the small eigenvalue was investigated at different signal strengths c. It was found that both the distribution and the mean of the small eigenvalue depend on the experiment node activity signal strength (figure 10). This result was obtained by calculating the small eigenvalues

for 10^3 samples of C' for each investigated signal strengths c. The samples of C' were generated by drawing samples of ϵ (since $C' = X'^T X'$ and $X' = X + \epsilon$). Three different values of n were used to generate the $n \times 2$ matrix ϵ: $n = 3$, $n = 10$ and $n = 50$.

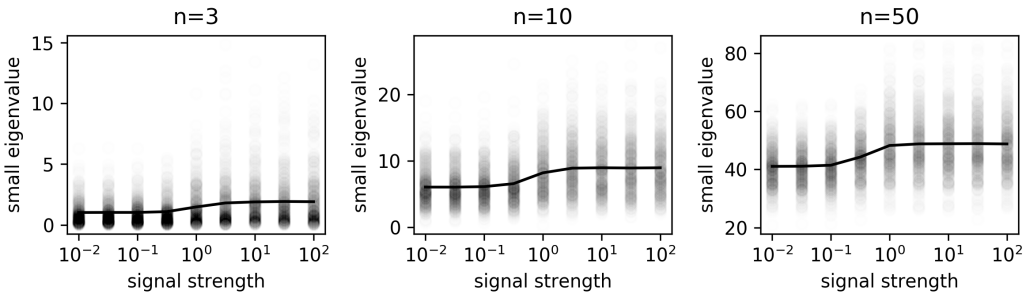


Figure 10: Small eigenvalues of samples of matrix C' at different signal strengths c (translucent dots were used to allow a simple representation of the density). The line shows the mean of the corresponding eigenvalues. As the amount of noise is the same for all data points and is equal to 1, the signal strength can also be interpreted as the signal to noise ratio.

Then, the two extreme cases of $c = 0$ and $c \to \infty$ were investigated in more detail. For signal strengths $c = 0$ and $c = 10^6$ (this large number was chosen to approximate $c \to \infty$), 10^4 samples of C' were generated followed by calculation of the small eigenvalue. Then, histograms were created based on the simulated data and used to approximate cumulative distribution functions. This way, it was found for $c \to \infty$ that the distribution of the small eigenvalue follows a χ^2 distribution with $n - 1$ degrees of freedom. For the case $c = 0$, it was was found by simulation and also shown analytically that the small eigenvalue follows a distribution whose integral is given by equation 5 (see derivation section). Figure 11 shows three exemplary evaluations of both cdfs (lines) alongside simulated results (crosses) for $n = 3$, $n = 10$ and $n = 50$. The analytical results match the simulated results.

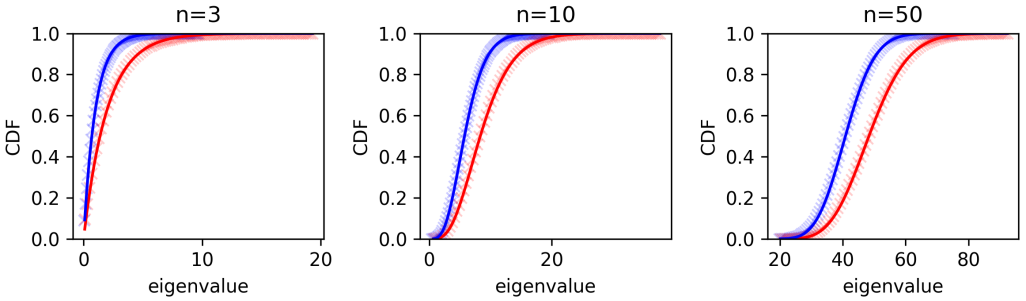


Figure 11: Cumulative distribution functions of the small eigenvalue at zero (blue) and very large (red) signal strengths. Crosses and lines correspond to the simulated and analytical results, respectively, and n are the degrees of freedom.

The geometric interpretation why the small eigenvalue follows a χ^2_{n-1} distribution for the case $c \to \infty$ is as follows (illustration in figure 12). If the node activities of two nodes are linearly dependent in the way explained above and no noise is present, they are two points on a line in a n-dimensional space (the line goes through the origin). If noise has corrupted the measurement, the points are

slightly off the line as determined by the amount of noise, and the large eigenvector* roughly points into the average direction of the points. As the signal strength goes to infinity, however, the direction of the large eigenvector converges to the direction of the line, that is, it will always point into the same direction for different samples of X'; the dimensionality of the subspace in which the large eigenvector varies then becomes equal to 1. Since matrix C' is symmetric, the small and large eigenvectors are orthogonal. This means that the dimensionality of the remaining subspace in which the small eigenvalue is free to vary becomes equal to $n-1$.

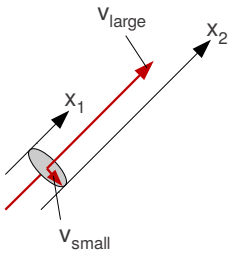


Figure 12: Illustration demonstrating why the distribution of the small eigenvalue converges to a χ^2_{n-1} distribution. The two points x_1 and x_2 in n-dimensional space are far away from the origin, and the eigenvector corresponding to the large eigenvalue, v_{large} does not change much for different samples of x_1 and x_2. That is, v_{large} always points into the same direction. The dimensionality of the subspace in which the eigenvector corresponding to the small eigenvalue (v_{small}) can vary is thus restricted to $n-1$.

*short form for "the eigenvector corresponding to the large eigenvalue"

Discussion

For intermediate node activity signal strengths, that is, $0 < c < \infty$, the cdf must be somewhere between both above-mentioned distributions. It may be possible to find an exact expression of the cdf for intermediate signal strengths. However, the signal strengths are typically unknown. This either requires estimation or utilization of the conditional distribution of the small eigenvalue given the large eigenvalue.

As long as the formula for such a cdf remains unavailable, the choice of which of the above-mentioned distributions to use depends on the following consideration: if it is acceptable that nodes are clustered more easily, the χ^2_{n-1} distribution should be used. The reason is that it takes larger values of the small eigenvalue to reach significant difference from the expected value. It is therefore more difficult to reject the Null Hypothesis that the two node activities are linearly dependent.

In practice, using the χ^2_{n-1} distribution should yield adequate results if the signal to noise ratio is sufficiently high. It can be obtained from figure 10 that the distribution of the small eigenvalue already converges at a signal to noise ratio of 10 (given a link strength of $\gamma \approx 1$).

Derivation

Theorem: Let X be an $n \times 2$ matrix with $X_{ji} \sim \mathcal{N}(0, 1)$, and let $C = X^T X$. The cumulative distribution function (cdf) of the smallest eigenvalue of matrix C is given by:

$$Pr\{X \leq x\} = F_\lambda(x) = 1 - K\left[H_R(\alpha, x) - H_L(\alpha, x)\right], \tag{5}$$

where

$$\begin{aligned}
K &= \frac{\pi^{\frac{1}{2}}}{2^n \Gamma(\frac{n}{2}) \Gamma(\frac{n-1}{2})} \\
\alpha &= \frac{n-3}{2} \\
H_L(\alpha, x) &= \frac{1}{\alpha+1} 2^{2(\alpha+1)} \Gamma(\alpha+1, \frac{x}{2})^2 - \frac{1}{\alpha+1} \Gamma(\alpha+1+\alpha+2, x) \\
H_R(\alpha, x) &= (\alpha+1) 2^{2(\alpha+1)} \Gamma(\alpha+1, \frac{x}{2})^2 + 2\Gamma(2(\alpha+1), x)
\end{aligned}$$

Proof: Note the following identities[112]:

- $\Gamma(\alpha, x) = \int_x^\infty t^{\alpha-1} e^{-t}\, dt$, which defines the upper incomplete Gamma function

- $\int_a^b t^{\alpha-1} e^{-t}\, dt = \Gamma(\alpha, a) - \Gamma(\alpha, b)$
- $\int_x^\infty t^{\alpha-1} e^{-\frac{t}{2}}\, dt = 2^c \Gamma(c, \frac{x}{2})$, which can be shown via integration by substitution.

The joint probability density function of the eigenvalues of an $m \times m$ Wishart matrix C (that is, $C = X^T X$, with X being an $n \times m$ matrix with elements $X_{ji} \sim \mathcal{N}(0, 1)$) as given by[113]:

$$f_\lambda(\lambda_1, \ldots, \lambda_m) = K \prod_{i=1}^{m} e^{-\frac{\lambda_i}{2}} \lambda_i^\alpha \prod_{i<j}^{m} (\lambda_i - \lambda_j) \tag{6}$$

The cdf of the smallest eigenvalue for the case $m = 2$ can then be obtained by integrating over the domain $D = \{(\lambda_1, \lambda_2) : x \leq \lambda_1 < \lambda_2 \leq \infty\}$:

$$\begin{aligned}
Pr\{x \leq X\} &= \iint_D f_\lambda(\lambda_1, \lambda_2)\, d\lambda_1\, d\lambda_2 \\
&= \int_x^\infty \int_x^{\lambda_2} K(\lambda_2 - \lambda_1) \lambda_1^\alpha e^{-\frac{\lambda_1}{2}} \lambda_2^\alpha e^{-\frac{\lambda_2}{2}}\, d\lambda_1\, d\lambda_2 \\
&= K \left(\underbrace{\int_x^\infty \int_x^{\lambda_2} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} \lambda_1^\alpha e^{-\frac{\lambda_1}{2}}\, d\lambda_1\, d\lambda_2}_{L(\alpha, x)} - \underbrace{\int_x^\infty \int_x^{\lambda_2} \lambda_2^\alpha e^{-\frac{\lambda_2}{2}} \lambda_1^{\alpha+1} e^{-\frac{\lambda_1}{2}}\, d\lambda_1\, d\lambda_2}_{R(\alpha, x)} \right)
\end{aligned}$$

To proceed further, the left and right helper functions $L(\alpha, x)$ and $R(\alpha, x)$ are derived separately:

$$\begin{aligned}
L(\alpha, x) &= \int_x^\infty \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} \int_x^{\lambda_2} \lambda_1^\alpha e^{-\frac{\lambda_1}{2}}\, d\lambda_1\, d\lambda_2 \\
&= \int_x^\infty \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} 2^{\alpha+1} \left(\Gamma(\alpha + 1, \frac{x}{2}) - \Gamma(\alpha + 1, \frac{\lambda_2}{2}) \right) d\lambda_2 \\
&= \underbrace{\int_x^\infty \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} 2^{\alpha+1} \Gamma(\alpha + 1, \frac{x}{2}) d\lambda_2}_{M} - \underbrace{\int_x^\infty \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} 2^{\alpha+1} \Gamma(\alpha + 1, \frac{\lambda_2}{2}) d\lambda_2}_{H_L(\alpha, x)}
\end{aligned}$$

Since it can be shown that M also appears in $R(\alpha, x)$ (see below), this term does not need further treatment here.

$$
\begin{aligned}
H_L(\alpha,x) &= \int_x^{\infty} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} 2^{\alpha+1} \left(\frac{\Gamma(\alpha+2,\frac{\lambda_2}{2}) - \left(\frac{\lambda_2}{2}\right)^{\alpha+1} e^{-\frac{\lambda_2}{2}}}{\alpha+1} \right) d\lambda_2 \\
&= \frac{1}{\alpha+1} 2^{\alpha+1} \int_x^{\infty} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} \Gamma(\alpha+2,\frac{\lambda_2}{2}) d\lambda_2 \quad - \\
&\quad \frac{1}{\alpha+1} 2^{\alpha+1} 2^{-(\alpha+1)} \int_x^{\infty} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} d\lambda_2 \\
&= \frac{1}{\alpha+1} 2^{2(\alpha+1)} \Gamma(\alpha+2,\frac{\lambda_2}{2})^2 - \frac{1}{\alpha+1} \Gamma((\alpha+1)+(\alpha+2),x)
\end{aligned}
$$

$$
\begin{aligned}
R(\alpha,x) &= \int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} \int_x^{\lambda_2} \lambda_1^{\alpha+1} e^{-\frac{\lambda_1}{2}} \, d\lambda_1 \, d\lambda_2 \\
&= \int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} 2^{\alpha+2} \left(\Gamma(\alpha+2,\frac{x}{2}) - \Gamma(\alpha+2,\frac{\lambda_2}{2}) \right) d\lambda_2 \\
&= \underbrace{\int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} 2^{\alpha+2} \Gamma(\alpha+2,\frac{x}{2}) d\lambda_2}_{M} - \underbrace{\int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} 2^{\alpha+2} \Gamma(\alpha+2,\frac{\lambda_2}{2}) d\lambda_2}_{H_R(\alpha,x)}
\end{aligned}
$$

$$
\begin{aligned}
H_R(\alpha,x) &= \int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} 2^{\alpha+2} \left((\alpha+1)\Gamma(\alpha+1,\frac{x}{2}) + \left(\frac{\lambda_2}{2}\right)^{\alpha+1} e^{-\frac{\lambda_2}{2}} \right) d\lambda_2 \\
&= (\alpha+1) 2^{\alpha+2} \int_x^{\infty} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} \Gamma(\alpha+1,\frac{\lambda_2}{2}) d\lambda_2 \quad + \\
&\quad 2^{\alpha+2} 2^{-(\alpha+1)} \int_x^{\infty} \lambda_2^{\alpha+1} e^{-\frac{\lambda_2}{2}} \lambda_2^{\alpha} e^{-\frac{\lambda_2}{2}} d\lambda_2 \\
&= (\alpha+1) 2^{2(\alpha+1)} \Gamma(\alpha+1,\frac{\lambda_2}{2})^2 + 2\Gamma((\alpha+1)+(\alpha+1),x)
\end{aligned}
$$

Finally:

$$
\begin{aligned}
Pr\{X \le x\} &= 1 - Pr\{x \le X\} \\
&= 1 - K\left[L(\alpha,x) - R(\alpha,x)\right] \\
&= 1 - K\left[(M - H_L(\alpha,x)) - (M - H_R(\alpha,x))\right] \\
&= 1 - K\left[H_R(\alpha,x) - H_L(\alpha,x)\right]
\end{aligned}
$$

■

The upper incomplete gamma function can readily be computed without the need for repeated numerical integration. This follows directly from the recurrent properties of both lower and upper

incomplete gamma functions. Specifically, for the upper incomplete gamma function, the following identity can be found[112]:

$$\Gamma(s+1,x) = s\Gamma(s,x) + x^s e^{-x}$$

As $\alpha = \frac{n-3}{2}$, with n an integer, computation of the incomplete gamma function must start at either $s = 0$ or $s = \frac{1}{2}$, so that[112] $0\,\Gamma(0,x) = 0$ and $\frac{1}{2}\Gamma(\frac{1}{2},x) = \frac{1}{2}\left[1 - erf(\sqrt{x})\right]$.

Furthermore, the eigenvalues λ_1 and λ_2 of C can be found analytically since the characteristic polynomial is a quadratic equation. With C_{ij} as an element of matrix C, the solution to the eigenvalue problem $|C + I\lambda| = 0$ is:

$$\lambda_{1|2} = \frac{C_{11} + C_{22}}{2} \pm \sqrt{\frac{C_{22}^2 - 2C_{11}C_{22} + C_{22}^2 + 4C_{21}C_{21}}{4}}$$

2 Supervised learning of biological sequence motifs

2.1 Summary of the obtained results

Biological sequence motifs enable specific molecular interactions and are the basis of many highly-ordered, fundamental biological processes. For example, information processing by gene regulatory networks is enabled by the specific association of transcription factors with their target genes. As the exact motif position in sequences such as those derived from ChIP-seq experiments can not be precisely located, motifs can be regarded as translation-invariant data features. The previous state-of-the-art method for inferring motifs from high-throughput data, DeepBind, was based on a convolutional neural network (CNN). However, this method required several filters to learn a single motif, and the filters also had to be longer than the motif, making it difficult to interpret them.

When a CNN using a single filter with the same length as the motif was trained to discriminate between sequences with and without a motif embedded at a random position, the CNN (figure 13 a) performed approximately as well as a fully connected network with one hidden layer (figure 13 c). A novel CNN architecture that employed all circularly shifted variants of the filter (figure 13 b) performed substantially better (figure 13 c).

When the filter weights of both the CNN and the CNN with circular filters were initialized with the actual motif followed by network training, the log-likelihoods of both networks on a test data set were very similar and correspond to the maximum likelihood solutions on the training data set (figure 13 d). This indicates that, although both CNN architectures had the capacity to learn the motif, the regular CNN was prone to local optima. It can be shown that these local optima correspond to filters that contain truncated and shifted motifs, which are learned as a consequence of gradient descent optimization and the limited filter sizes of regular CNNs. CNNs with circular filters however do not have filter edges and hence cannot learn truncated motifs, given sufficiently large filter sizes. This makes them more likely to converge to the global optimum, explaining the above-mentioned performance difference between the architectures.

In practice, CNNs with circular filters enable robust prediction of transcription factor binding sites even with a very low number of training examples or long sequences. They also enable easy interpretation of these binding sites.

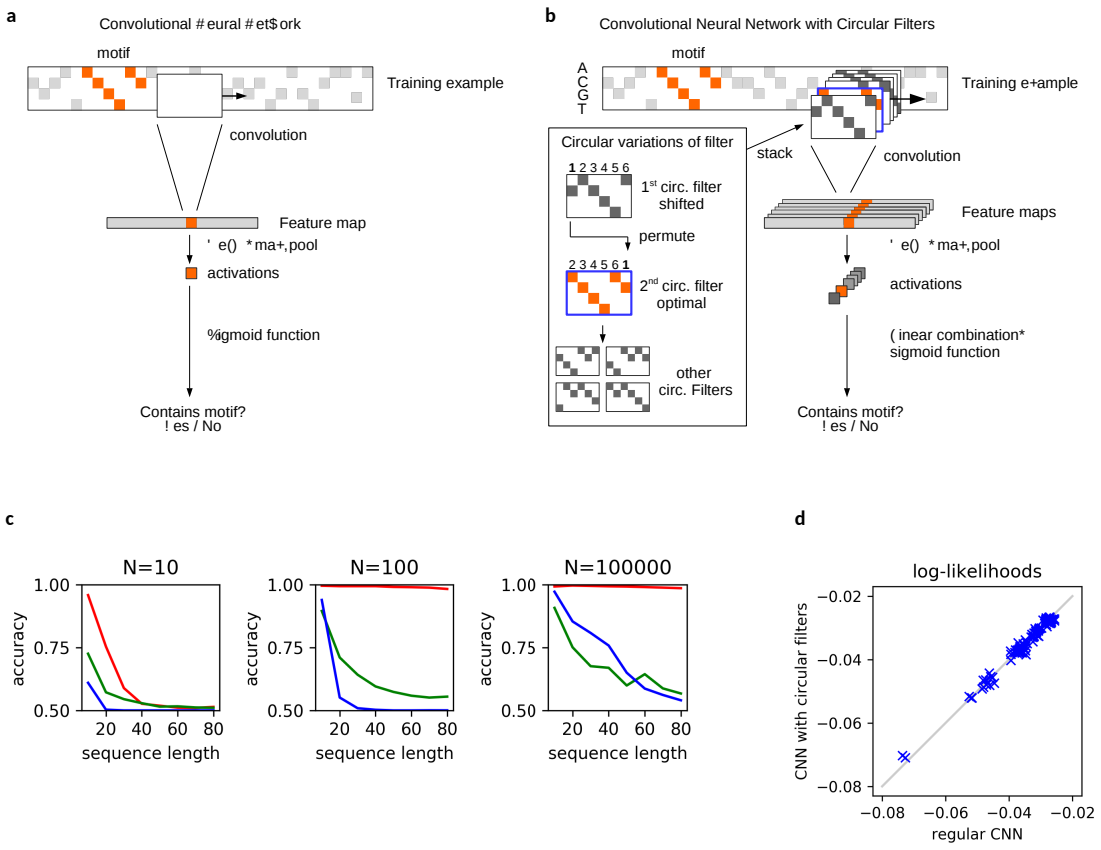


Figure 13: **a**: Illustration of a regular convolutional neural network (CNN) for motif inference. A training example is convolved to yield a feature map, of which the largest value is mapped to the probability that the sequence contains a motif.
b: Illustration of CNN with circular filters. Here, the sequence is convolved with all circularly shifted variants of the same filter.
c: Effect of sequence length and the number of positive examples on motif inference. The objective was to infer motifs of length 6 nucleotides hidden in the sequences. Lines show median test accuracies based on 100 data sets. Red: CNN with circular filters; green: CNN; blue: network with one fully connected layer. Both CNN architectures used filter lengths corresponding to 6 nucleotides.
d: The filters of a regular CNN and a CNN with circular filters were initialized with the actual motif and then trained on the same training data set. This way, both models converged the optimal model parameters (maximum likelihood solutions). Then, the average loss function value (log-likelihood) on a test data set was evaluated for both models. It can be seen that both models achieved almost identical loss function values.

2.2 Manuscript I: Circularly shifted filters enable data-efficient motif inference

Authors: C. F. Blum & M. Kollmann
This manuscript was submitted to *Bioinformatics*

Contribution of Christopher Blum:

- wrote the manuscript
- designed the figures
- developed and assessed the method
- compared the method to existing methods

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Sequence analysis

Neural networks with circular filters enable data efficient sequence motif inference

Christopher F. Blum [1], and Markus Kollmann [1,*]

[1] Institute for Mathematical Modeling of Biological Systems, Heinrich-Heine University of Düsseldorf, 40225 Düsseldorf, Germany

*To whom correspondence should be addressed.

Abstract

Motivation: Nucleic acids and proteins often have localized sequence motifs that enable highly specific interactions. Due to the biological relevance of sequence motifs, numerous inference methods have been developed. Recently, convolutional neural networks (CNNs) achieved state of the art performance because they can approximate complex motif distributions. These methods were able to learn transcription factor binding sites from ChIP-seq data and to make accurate predictions. However, CNNs learn filters that are difficult to interpret, and networks trained on small data sets often do not generalize optimally to new sequences.

Results: Here we present circular filters, a novel convolutional architecture, that contains all circularly shifted variants of the same filter. We motivate circular filters by the observation that CNNs frequently learn filters that correspond to shifted and truncated variants of the true motif. Circular filters enable learning of non-truncated motifs and allow easy interpretation of the learned filters. We show that circular filters improve motif inference performance over a wide range of hyperparameters. Furthermore, we show that CNNs with circular filters perform better at inferring transcription factor binding motifs from ChIP-seq data than conventional CNNs.

Availability: Example code is available at https://github.com/christopherblum

Contact: markus.kollmann@hhu.de

1 Introduction

A fundamental property of biological macromolecules such as DNA, RNA and proteins is their ability of highly specific interactions. These high specificities are often associated with localized motifs in the primary structure of these macromolecules. Intricate motif variations, indirect effects on binding specificity and noisy data make motif inference from high-throughput data a challenging task (Man and Stormo, 2001; Rohs *et al.*, 2010; Kidder *et al.*, 2011; Berger and Bulyk, 2009; Weirauch *et al.*, 2013).

As DNA, RNA or proteins participate in almost all processes that have biotechnological or biomedical relevance, numerous methods have been developed to infer motifs and binding specificities from high-throughput data sources (Weirauch *et al.*, 2013). These methods range from derivatives of nucleotide frequency counting procedures such as *k*-mer and position weight matrix (PWM) methods to, most recently, convolutional neural networks (CNNs) (Alipanahi *et al.*, 2015; Zeng *et al.*, 2016). Unlike PWM methods, CNNs do not require aligned input sequences because they convolve input sequences with multiple sliding weight matrices called filters. Although this sliding operation resembles *k*-mer methods, CNNs do not rely on predefined *k*-mers. Instead, they learn the weight matrices by maximizing an objective function. This function is either a distance measure between observed and predicted fluorescence intensities or read counts or class-conditional probabilities of labelled sequences. While position weight matrices allow intuitive interpretation of the weight matrices in terms of information content with respect to a background model, CNNs distribute a motif's distributional information across the multiple filters in a non-trivial way, preventing immediate interpretation.

Convolutional neural network architectures have achieved state of the art performance at predicting fluorescence intensities derived from protein binding microarray data (Alipanahi *et al.*, 2015). It has been reported, however, that training these models with gradient descent methods can be sensitive to weight initialization, impairing generalization ability (Zeng *et al.*, 2016). As Stochastic Gradient Langevin Dynamics (SGLD) has been proposed as a general-purpose method for approximate bayesian inference that can strongly reduce the risk of overfitting, we utilize SGLD for motif inference with CNNs (Welling and Teh, 2011).

Deep neural networks typically require large amounts of training data. Due to the noise of protein binding microarray and ChIP-seq derived data, the number of positive training examples is often limited. A CNN-based motif inference method, DeepBind, addresses this problem by artificially increasing the number of negative training examples with data augmentation, but still relies on a sufficient amount of positive training examples, that is, sequences with high fluorescence intensities or read counts. (Alipanahi *et al.*, 2015; Simard *et al.*, 2003).

Here, we present a neural network algorithm that utilizes a novel convolutional architecture called circular filters that enables efficient data utilization and easy interpretation of the learned filters. First, it is shown that conventional CNN filters often contain shifted and truncated motifs. We then introduce circular filters as a natural solution to this problem and show that motif inference improves for a wide range of hyperparameters, allowing better inference when data is scarce. Finally, we demonstrate that a CNN with circular filters and SGLD yields accurate predictions for ChIP-seq derived data, resulting in the state of the art algorithm for motif inference from smaller-sized data sets.

2 Results

2.1 Convolutional neural networks frequently learn truncated motifs

While studying motif inference with CNNs, we observed that the learned filter weights frequently did not correspond to the complete motif. Instead, the filters often contained truncated and shifted versions of the motif (Figure 1a). To quantify this behaviour, we conducted simulations in which a known motif had to be inferred from a set of short sequences. Then, it was counted how frequently the filter contained to a shifted version of the motif. We found that CNNs learned shifted and truncated motifs more frequently than the true motifs (Figure 1b).

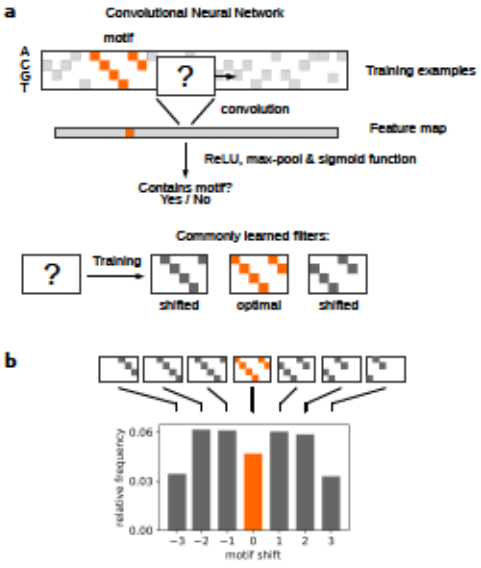


Fig. 1. a) Illustration of a conventional convolutional neural network (CNN). The network is trained to discriminate between sequences with and without motifs in a supervised manner. Comparison of the learned filters with the actual motif reveals that they often contain a truncated and shifted version of the motif. b) Shifted motif variants are being learned more frequently than the correct, unshifted motif by conventional CNNs. Bars show relative frequency of shifts derived from CNN filters that were trained on 4096 different data sets corresponding to all possible 6-mers, with 100 positive examples in each data set. The motifs pointing to the bars are for illustration purposes only and do not represent the actually learned motifs.

2.2 Circular filters improve robustness of sequence motif inference for simualted data

This observation motivated us to develop a novel convolutional kernel that already contains all circularly shifted variants of the same underlying filter, which we refer to as circular filters (Figure 2a). If one of the filters in this kernel learns a shifted motif, there is another filter variant that is able to learn the full motif, provided the filters have at least the size of the motif. Based on the feature map with the largest activation, the filter variant that contains the non-shifted motif can be recovered. In simulations we found that CNNs with circular filters rarely learn shifted motifs (Figure 2b). When trained on 100 positive examples to infer motifs

of 6 nucleotides length that are randomly located in a 40 nucleotide long sequence, a CNN with circular filters learned the correct motifs 5.5 times more often than a CNN without circular filters (Supplementary Table S1). Even when the CNN with circular filters was trained with only 5 positive examples, the correct motif was found 1.9 times more often ($p < 10^{-27}$).

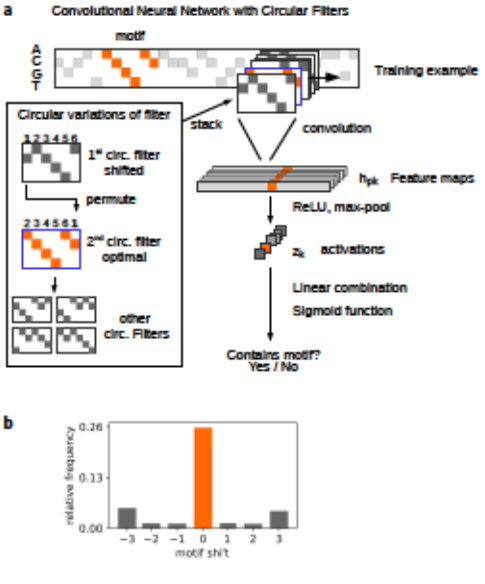


Fig. 2. a) Illustration of a convolutional neural network with circular filters. The circular filter consists of all possible circular variants of the same underlying filter. Convolution with one circular filter of length N yields N feature maps that are linearly combined after max-pooling. b) The correct motif is learned more frequently than shifted motifs. Bars show relative frequency of shifts derived from CNN filters that were trained on 4096 different data sets corresponding to all possible 6-mers, with 100 positive examples in each data set.

We quantified the effect of circular filters on motif recognition by comparing network architectures with and without circular filters for a variety of hyperparameter combinations. These included the number of positive training examples, L_2-regularization strength and the amount of noise injected into parameter updates. As the original architecture with circular filters (Figure 2a) forms a linear combination of the activations which increases the number of parameters compared to the architecture without circular filters (Figure 1a), we also designed two architectures without additional parameters. These architectures use a simple sum or max-out of the activations (Goodfellow *et al.*, 2013). Overall, both the CNN with circular filters and the CNN with circular filters and sum of activations performed significantly better than the regular CNN in 74% and 62% of all cases, respectively ($p < 10^{-4}$). The CNN with circular filters and max-out however performed significantly worse than the regular CNN in 56% of cases ($p < 10^{-4}$). A more detailed comparison is given in Supplementary Table S2.

2.3 Circular filters lead to state of the art performance for motif inference

To investigate if circular filters improve inference of biological motifs, we trained a CNN with circular filters to infer transcription factor binding sites from ChIP-seq data (Dunham *et al.*, 2012). The area under the receiver operator characteristic (AUROC) on evaluation data sets was used as a performance measure. The state of the art method for motif inference, DeepBind, was used as a reference (Alipanahi *et al.*, 2015). Performance of both methods was assessed for different filter sizes, numbers of filters, and number of positive training examples.

We found that the CNN with circular filters led to AUROC values greater than those of DeepBind for a variety of parameter combinations (Figure 3, Supplementary Table S3). The difference in performance was especially large when only a single filter was used and 500 positive training examples were available (Figure 4). With more and larger filters and less positive training examples, the difference in performance was generally smaller.

Furthermore, we found that the CNN with circular filters produced AUROC values above 0.5 more often than DeepBind (Supplementary Table S3). Especially when only one filter was used, DeepBind yielded AUROC values smaller than 0.5 both when either 5 or 500 positive training examples were provided. In contrast, the CNN with circular filters displayed this behaviour only when 5 positive training examples were available.

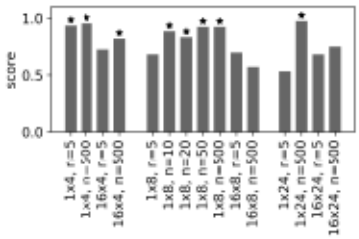


Fig. 3. Performance comparison between a CNN with circular filters and DeepBind on ChIP-seq data published by the ENCODE consortium. The score indicates the ratio of times that the CNN with circular filters led to AUROC values larger than those obtained the state of the art method for inferring sequence motifs. Labels indicate number and size of filters as well as the number of positive examples, e.g. 1×4, $n = 5$ means that one filter of length 4 was used on a data set with 5 positive examples. Asterisks indicate p-values below 10^{-5} (Binomial test).

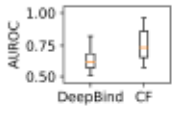


Fig. 4. Circular filters help to infer and predict transcription factor binding sites. Comparison based on ChIP-seq data sets published by the ENCODE consortium. Both algorithms were trained on 500 examples and used 1 filter of length 8 nucleotides. Box-plot whiskers show 2.5% and 97.5% quantiles.

3 Methods

3.1 Investigated network architectures

3.2 Comparing learned filters and true motifs

Synthetic data sets were generated for all 4096 6-mers. Each data set consisted of either 5 or 100 positive and $10,000$ negative examples. The positive examples were created by embedding a 6-mer in random sequences of 40 nucleotides length each. The set of $10,000$ negative examples was created by randomly sampling (with replacement) from the set of positive examples and then randomly shuffling the nucleotide order of each sequence. All sequences were converted to a one-hot representation to be used as input by the network.

Two network architectures were investigated: the regular CNN and the CNN with circular filters (Figure 1a and Figure 2a, respectively). The networks were trained for $10,000$ steps at a learning rate of 0.01.

The learned filters were converted to a one-hot representation indicating the largest weight at each filter position and then and compared to the original 6-mer motif. In this comparison, it was checked if the learned filters corresponded to shifted and truncated versions of the original motif. Specifically,

- shifts -3, -2 and -1: it was checked if the l rightmost nucleotides of the filter were equal to the l leftmost nucleotides of the motif hidden in the sequences, with $l \in 3, 4, 5$,
- shift 0: it was checked if all 6 nucleotides of the filter were equal to the complete motif hidden in the sequences
- shifts 1, 2, 3: it was checked if the l leftmost nucleotides of the filter were equal to the l rightmost nucleotides of the motif hidden in the sequences, with $l \in 3, 4, 5$.

The number of times, k, with which shifted motifs were learned from the $n = 4096$ data sets was tested for statistical significance using the Binomial distribution $B_{n,p}(k)$, where $p = \frac{1}{4^l}$ is the probability to observe a motif of length l by chance. Since some of the resulting p-values were too small to be calculated, the smallest k necessary for a p-value less than or equal to 10^{-5}, k_{signif}, was calculated instead. For example, the value of the leftmost bar in the barplot in Figure 1b at $0.034 \approx \frac{141}{4096}$ is significant because a shift of -3 was observed 141 times and k_{signif} was 100. Values for all k_{signif} can be obtained from Table 1.

To assess whether the CNN with circular filters learned unshifted motifs significantly more often than the regular CNN, we modelled the null hypothesis with a binomial distribution $B_{n,\hat{p}}(k)$, with $n = 4096$ and $\hat{p}$ the estimated probability of the regular CNN learning an unshifted filter, and k the number of times the CNN with circular filters learned an unshifted motif.

3.3 Influence of circular filters on motif recognition

To investigate the effect of circular filters on motif inference for different hyperparameter settings (Supplementary Table S2), we trained and evaluated CNNs with and without circular filters on simulated data for a variety (grid) of hyperparameter combinations. We then tested for which hyperparameter settings the utilization of circular filters resulted in significantly better performance. The following hyperparameters were investigated: values of 0, 0.1, 1.0, 10 and 100 were used for both the SGLD noise variance scaling factor and the L_2-regularization strength, filter lengths were either 3, 4, 6 or 12 nucleotides, and either 5, 50 or 500 positive training examples were used for training.

We generated synthetic data sets of labelled sequences with and without a motif (positive and negative examples, respectively) for training and evaluation. First, 32 random motifs of 6 nucleotides length were generated (Supplementary Table S4). For the training data, three data sets corresponding to the number of positive training sequences (5, 50 and 500) were generated for each motif. Depending on this number, the corresponding number of random nucleotide sequences of 40 nucleotides length was generated, and a motif was then placed at a random position in each of these sequences. These sequences corresponded to the positive examples. Negative examples were generated by randomly sampling $10,000$ times from the positive examples and then randomly shuffling the nucleotide order of each sequence. Evaluation data sets were created for each of the 32 motifs and consisted of 1000 positive and 1000 negative examples. All sequences were converted to one-hot coding.

Networks were trained for 40000 training steps at a learning rate of 0.01. Mini-batches consisted of randomly (with replacement) selected 10 positive examples and 10 negative examples. The cross-entropy between sequence labels and predicted sequence labels was used as objective function and was minimized using SGLD (Welling and Teh, 2011). AUROC values were calculated based on predictions on the evaluation data sets.

We then calculated the ratio with which the CNNs with circular filters resulted in higher AUROC values than the CNN without circular filters for a particular hyperparameter. For example, to determine whether circular filters at an L_2 regularization strength of 0.1 overall improved performance, we kept the L_2 regularization strength fixed at 0.1 and then counted how often the CNNs with circular filters led to larger AUROC values than the CNN without circular filters for all remaining hyperparameter combinations of SGLD noise variance scaling factor, number of positive training examples, filter length and all 32 motifs. To test the obtained ratio, r, for statistical significance ($H_0 : r > 0.5$), we bootstrapped over the 32 motifs ($100,000$ boostrap samples) and used a significance level of $\alpha = 10^{-4}$.

3.4 Performance comparison between DeepBind and a CNN with circular filters using ENCODE data

The performance of DeepBind and the CNN with circular filters was evaluated for several combinations of filter length, number of filters and number of positive training examples (Figure 3). Both algorithms were provided with exactly the same training and test data.

Data preprocessing is described elsewhere in more detail (Alipanahi *et al.*, 2015). In short, input data was derived from 48 randomly selected transcription factor ChIP-seq experiments for which the data was published by the Encyclopedia of DNA Elements (ENCODE) Consortium (Dunham *et al.*, 2012). These ChIP-seq data sets comprised of mapped sequence reads and read numbers for the most significant peaks. Regions around read peaks (starting 50 nucleotides before and ending 50 nucleotides after each peak) were extracted and sorted according to the read number in descending order. As the first 1000 sequences had the largest read numbers, it was assumed that they contained a binding site for the respective transcription factor. These sequences constituted the positive examples. For each ChIP-seq experiment, the first 500 even-numbered positive examples were allocated to the corresponding test data set. Depending on the setting, either the first 5, 10, 20, 50 or 500 odd-numbered sequences were allocated to the corresponding training data set.

Negative examples for the training and test data sets were created based on the corresponding positive examples. For the training data, $10,000$ sequences were randomly sampled with replacement from the positive examples. Then, the nucleotide order within the individual sequences was shuffled using dinucleotide shuffling, a method that shuffles the nucleotide order while preserving the frequency of neighbored nucleotides (Altschul and Erickson, 1985). For the test data, two data sets were created by first copying all 500 positive examples and then shuffling the nucleotide using either random shuffling and dinucleotide shuffling.

Because our computational power was limited, we restricted our comparison to randomly selected 48 out of the 506 ChIP-seq data sets that were used in the comparison by the DeepBind authors.

Both DeepBind and the CNN with circular filters assigned a score to each sequence, indicating the likelihood that the sequence contained a motif. By assigning a score to each of the 500 positive and 500 negative examples in the test data sets, AUROC values could be calculated.

Before any further testing, only runs in which both CNN with circular filters and DeepBind had AUROC values greater than 0.01 were used. This was found to be necessary because for some runs, DeepBind assigned a class conditional probability of 0.5 to all positive and negative examples, resulting in AUROC values equal to 0.0. This behaviour has been reported before (Zeng *et al.*, 2016). For each parameter combination and ChIP-seq experiment, the number and fraction how often CNN with circular filters had AUROC values above the corresponding DeepBind AUROC values was obtained and tested for significance (Figure 3 and Supplementary Table S3). The null hypothesis was modelled with a binomial distribution $B_{n,p}(k)$ with n the number experiments and $p = 0.5$. Because it can happen that neural networks do not converge to the global minimum during training, ultimately yielding AUROC values either close to or even below 0.5, a second statistic was calculated as well that disregarded such runs. For this statistic, only the runs in which both CNN with circular filters and DeepBind led to AUROC values greater than 0.5 were used to calculate the number of times CNN with circular filters yielded AUROC values greater than those of DeepBind.

DeepBind settings The default DeepBind training parameters were used, except for an override of the filter length and the number of filters. Furthermore, the original routine for generating negative examples de novo was replaced by a routine to load the negative examples from the hard drive to ensure better comparability. To obtain a platform-independent instance of DeepBind, DeepBind was run in a Docker (Merkel, 2014) container using Nvidia Docker. This was necessary because former attempts towards a platform-independent implementation were still depending on a certain CUDA-version (Zeng *et al.*, 2016; Nickolls *et al.*, 2008).

Settings for CNN with circular filters Models were trained using a learning rate of 0.01 for 40,000 training steps. We used a batch size of 20 consisting of 10 positive and 10 negative examples, all randomly sampled (with replacement) from the respective training data.

For hyperparameter tuning, the model was trained for 5 different regularization strengths. The training data was split up into a training and a validation data set (80% and 20% of the data, respectively). Every 50 steps, it was determined whether the error on the validation set was lower than before, in which case the current model was stored for later restoring. Then, the model corresponding to the regularization strength that yielded the lowest error on the validation set was used for evaluation on the test data set, for which the the model that yielded the lowest validation error was restored.

4 Algorithm

4.1 One-hot coding

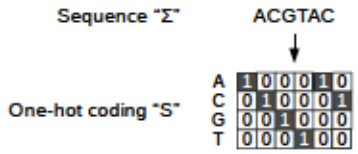


Fig. 5. Illustration: a nucleic acid sequence represented as one-hot coding.

A sequence $\Sigma = \Sigma_1, \ldots, \Sigma_{L_S}$ with elements Σ_j coming from an ordered set with limited finite cardinality $\Sigma_i \in D, |D| = N$ can be represented as a $N \times L_S$ one-hot coding matrix S with elements S_{ij} that are equal to 1 if Σ_j is the i-th element in D and 0 otherwise (Figure 5).

4.2 Formal definition of 1-dimensional convolution with one circular filter

The unpadded convolution of a one-hot coded sequence S with length L_S and N features and one circular filter of length L_F and N features to a hidden node h_{pk}, $p \in \{1, \ldots, L_S - L_F + 1\}, k \in \{1, \ldots, L_F\}$, can formally be defined as:

$$h_{pk} = \sum_{i=1}^{L_f} \sum_{j=1}^{N} S_{j,(i+p-1)} F_{jm}, \text{ with } m = (i + k - 1) \pmod{L_F}$$

4.3 Formal definitions of the CNNs with circular filters

We model the class-conditional probability $p(C_{\text{motif}}|S_n)$ that a sequence with one-hot coding S_n belongs to class C_{motif} ("contains motif") as a convolutional network. Input to the network is a $N \times L_S$ one-hot coding S_n, where L_S is the length of the sequence and N is the number of features (that is, the number of different nucleotides or amino acids). The input sequence S_n is convolved without padding, followed by a Rectifying linear unit (ReLU) and max-pooling to yield activations z that are mapped onto the class-conditional probabilities depending on the architecture. The objective function was the batch-averaged cross-entropy between predicted and observed classes. In the following, the $N \times L_F$ matrix F is the convolutional filter of length L_F, w_k or w are weights and b is a bias, and $\sigma(x) = \frac{1}{1+e^{-x}}$ denotes the sigmoid function.

The CNN without circular filters (Figure 1a) is a chain of functions in the following order:

$$\begin{aligned} h_p &= \sum_{i=1}^{L_f} \sum_{j=1}^{N} S_{j,(i+p-1)} F_{ji} \\ a_p &= \max(0, h_p) \\ z &= \max\{a_1, \ldots, a_{L_S - L_F + 1}\} \\ p(C_{\text{motif}}|S_n) &= \sigma(wz + b) \end{aligned}$$

The CNN architectures with circular filters used here all convolve the input with circular filters but differ in the operation applied to the activations z_k, specifically:

$$h_{pk} = \sum_{i=1}^{L_f} \sum_{j=1}^{N} S_{j,(i+p-1)} F_{jm}, \text{ with } m = (i + k - 1) \pmod{L_F}$$

$$a_{pk} = \max\left(0, h_{pk}\right)$$

$$z_k = \max\left\{a_{p,1}, \ldots, a_{p,L_S - L_F + 1}\right\}$$

For the convolutional neural network with circular filters (Figure 2a), we then have:

$$p(C_{\text{motif}}|S_n) = \sigma\left(b + \sum_{k=1}^{L_F} w_k z_k\right)$$

Whereas for the convolutional neural network with circular filters and sum of the activations, we have:

$$p(C_{\text{motif}}|S_n) = \sigma\left(b + w \sum_{k=1}^{L_F} z_k\right)$$

Finally, for the convolutional neural network with circular filters and max-out, we have:

$$p(C_{\text{motif}}|S_n) = \sigma\left(b + w \max\left\{z_1, \ldots, z_{L_F}\right\}\right)$$

5 Discussion

5.1 Motif inference differs from deep learning disciplines

We showed in a simple simulation that a shallow convolutional neural network architecture frequently learns non-optimal filters that correspond to shifted and truncated versions of the underlying inferable pattern. The networks were trained with stochastic gradient descent, indicating that the non-optimal filters correspond to locally optimal variable combinations. This is in agreement with the well-studied behavior of gradient descent optimization which is highly sensitive to initial conditions and prone to local optima. The local optima observed here seem to be more difficult to escape compared to other machine learning disciplines where CNNs are applied.

5.2 Circular filters enable robust motif inference

The filter weights of conventional CNNs are trained by starting from random seeds. For motif inference, this seed determines at which position in the filter the inferred motif will be developed during training. This means that it can happen that only one side of a motif can be learned because an edge of the filter has been reached. With circular filters, there are no edges and the position from where the motif develops does not matter anymore.

It has been observed before that that slight modifications in network structure alongside negligibly more parameters can substantially improve inference performance (Ioffe and Szegedy, 2015; He *et al.*, 2015). These findings have demonstrated the importance of hard-wired prior knowledge about the underlying problem. Also circular filters can be regarded as hard-wired prior knowledge: sequence motifs are locally correlated data features that require convolution to be learned and optimization with gradient descent requires sufficient space to develop the motif from initial weights.

Since the three investigated architectures with circular filters only differed in the function applied to the activations, this must be the reason for the observed performance differences. For the CNN with circular filters and max-out, backpropagation of the classification error can only occur to the circular filter variant that led to the maximum activation. After applying the parameter updates, it can happend that another circular filter variant leads to the maximum acitivation in the next training step. This may complicate parameter optimization, explaining the lower performance compared to the architecture without circular filters. For the other architectures, the classification error can backpropagate to all filter variants, allowing the motif to be learned in any of the circular filters. However, also filter variants that do not contain the inferable pattern can contribute to the classification error, which can be harmful if some training sequences also randomly contain other patterns. For the CNN with circular filters, it can be adjusted by a linear combination how much the filter variants contribute to the classification, which is not the case for the CNN with circular filters and sum of activations, which may explain the difference in performance.

5.3 Deep models are not necessary for modeling TF-DNA specificities

It has been implied that deep neural networks were necessary to model transcription factor specifitities because a convolutional neural network architecture achieved state of the art performance at predicting probe intensities derived from protein binding microarray data (Alipanahi *et al.*, 2015). Although neural networks are complex function approximators and a deep CNN with 152 layers achieved state of the art performance at classifying images, it was demonstrated that deeper architectures actually perform worse at learning sequence motifs than simpler architectures (Hornik, 1991; He *et al.*, 2015; Zeng *et al.*, 2016).

A likely reason is that biological sequences are not composed of complex hierarchies of patterns such as those in images. In fact, mutually exclusive sequence patterns or spatial relationships can already be modeled with two layers, and to the best of our knowledge, no transcription factor has yet been found which binds to mutually exclusive motifs. This likely makes truly deep architectures unnecessary and possibly deleterious because more parameters need to be trained. Also, most transcription factors bind to motifs of 30 nucleotides length, a size that can be captured with simple convolutional filters (Stewart *et al.*, 2012). Because of the aforementioned reasons, it is unsurprising that the discriminative implementation of DeepBind has only four layers, which are convolutional, pooling, ReLU and fully connected layers.

5.4 Summary

When applied to biological data, a CNN with circular filters performed at least as good as the current state of the art algorithm for several combinations of filter number and size. Even for small sample sizes, the CNN with circular filters was able to infer correct motifs more often than a CNN without circular filters trained with 20 times more examples. Overall, our findings show that circular filters enable more efficient use of data for sequence motif inference.

Acknowledgements

Computational services were provided by the High Performance Computing Plattform of the Heinrich-Heine University in Düsseldorf, Germany.

Funding

Funding for this work was provided by the IMPRS MPI Cologne as well as by the DFG through the Cluster of Excellence in Plant Sciences (grant no. EXC 1028).

References

Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. (2015). Predicting the sequence specificities of dna- and rna-binding proteins by deep learning. *Nat Biotech*, **33**(8), 831–838. Computational Biology.

Altschul, S. F. and Erickson, B. W. (1985). Significance of nucleotide sequence alignments: a method for random sequence permutation that preserves dinucleotide and codon usage. *Mol. Biol. Evol.*, **2**(6), 526–538.

Berger, M. F. and Bulyk, M. L. (2009). Universal protein-binding microarrays for the comprehensive characterization of the DNA-binding specificities of transcription factors. *Nat Protoc*, **4**(3), 393–411.

Dunham, I., Kundaje, A., Aldred, S. F., *et al.* (2012). An integrated encyclopedia of DNA elements in the human genome. *Nature*, **489**(7414), 57–74.

Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013). Maxout networks. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1319–1327, Atlanta, Georgia, USA. PMLR.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, **abs/1512.03385**.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, **4**(2), 251 – 257.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR.

Kidder, B. L., Hu, G., and Zhao, K. (2011). ChIP-Seq: technical considerations for obtaining high-quality data. *Nat. Immunol.*, **12**(10), 918–922.

Man, T. K. and Stormo, G. D. (2001). Non-independence of Mnt repressor-operator interaction determined by a new quantitative multiple fluorescence relative affinity (QuMFRA) assay. *Nucleic Acids Res.*, **29**(12), 2471–2478.

Merkel, D. (2014). Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, **2014**(239).

Nickolls, J., Buck, I., Garland, M., and Skadron, K. (2008). Scalable parallel programming with cuda. *Queue*, **6**(2), 40–53.

Rohs, R., Jin, X., West, S. M., *et al.* (2010). Origins of specificity in protein-DNA recognition. *Annu. Rev. Biochem.*, **79**, 233–269.

Simard, P. Y., Steinkraus, D., and Platt, J. C. (2003). Best practices for convolutional neural networks applied to visual document analysis. *Proceedings of the Seventh International Conference on Document Analysis and Recognition*.

Stewart, A. J., Hannenhalli, S., and Plotkin, J. B. (2012). Why transcription factor binding sites are ten nucleotides long. *Genetics*, **192**(3), 973–985.

Weirauch, M. T., Cote, A., Norel, R., *et al.* (2013). Evaluation of methods for modeling transcription factor sequence specificity. *Nat. Biotechnol.*, **31**(2), 126–134.

Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 681–688, USA. Omnipress.

Zeng, H., Edwards, M. D., Liu, G., and Gifford, D. K. (2016). Convolutional neural network architectures for predicting DNA-protein binding. *Bioinformatics*, **32**(12), i121–i127.

2.3 Appendix: Investigation of motif inference from long sequences

Motivation

In section 2.2, a novel convolutional architecture was introduced that enabled data-efficient motif inference. However, an effect of sequence length on performance was not investigated. Furthermore, it remained to be shown whether convolutional neural networks (CNNs) and CNNs with circular filters have comparable model capacities since they have a comparable number of parameters.

Methods

To investigate the effect of sequence length on model performance, 100 different data sets were generated for sequences between 10 and 80 nucleotides length. Each training data set consisted of either 10, 100 or 100,000 positive training examples and 100,000 negative training examples, and each test data set consisted of 1,000 positive and 1000 negative examples. Positive examples were created by embedding a random 6-mer in random sequence backgrounds. The set of 10,000 negative examples was created by randomly sampling (with replacement) from the set of positive examples and then randomly shuffling the nucleotide order of each sequence. All sequences were converted to a one-hot representation to be used as input by the network. Three network structures were investigated: a CNN with circular filters, a regular CNN (see section 2.2), and a fully connected network. The latter used a matrix of size $4L_S \times L_S - L_F + 1$, with L_S the length of the sequence and L_F the filter length, to map the $4 \cdot L_S$ one-hot coded features (there are 4 possible nucleotides at each position of the sequence) of the input sequence to hidden nodes. The hidden layer size of $L_S - L_F + 1$ was chosen to make it equally large as in the convolutional architectures. Then hidden layer was then treated as in the regular CNN and the CNN with circular filters. Networks were trained with mini-batch gradient descent with balanced batches containing 50 positive and 50 negative training examples at a learning rate of 0.1 for 10,000 steps.

To compare the CNN with circular filters and the regular CNN in terms of likelihood function values on a test data set for models close to the maximum likelihood solution on training data sets, 100 different data sets corresponding to random 6-mers were generated. Each training data set contained 100,000 positive and negative examples, and each test data set contained 1,000 positive and negative examples. Filter weights were initialized with the actual motif, and networks were trained for 10,000 steps at learning rate 0.1 and for an additional 10,000 steps at learning rate 0.01 to reduce noise induced by the limited batch size of stochastic gradient descent. Again, balanced batches of size 100 were used.

Results

When few positive training examples were available, the performance of all methods strongly decreased with sequence length (see figure 13 c in the summary section at the beginning of this chapter). Both at 100 and 100,000 positive training examples, the performance of the CNN with circular filters remained nearly constant with increasing sequence length, close to a median accuracy of 1.0, whereas the performance of the fully connected network and the regular CNN quickly dropped.

When a CNN and a CNN with circular filters were trained on the same data and initialized with near-optimal weights, their likelihood functions on test data sets were almost identical (figure 13 d in the summary section).

Discussion

The results indicate that CNNs with circular filters, trained with mini-batch gradient descent, enable motif inference that is much less affected by sequence length than other methods. Furthermore, the fact that a CNN and a CNN with circular filters perform equally well at their maximum likelihood solutions suggests that network optimization is the cause for the observed performance differences when both models are trained with randomly initialized weights. Overall, the results presented here support the findings of section 2.2.

2.4 Appendix: Investigation of a sequence background model

Motivation

A sufficiently large number of training examples can help to avoid overfitting. Generating adequate negative samples can be difficult, especially when sequences contain structured backgrounds. If, for example, the task was to find a pattern in gene sequences, creating additional negative examples by random sequence shuffling might lead to the case where the difference between genes and the random background (noise) is learned instead of the pattern. Such cases might be avoided by using methods other than random shuffling to maintain some background distribution features. In fact, the principle of noise contrastive estimation [114] suggests that the ability of a model to learn noise declines with the distributional distances between positive and negative training examples. This means that, with a shuffling method that preserves some features of the positive training examples when generating negative training examples, one would expect better model performance as the models are forced to learn minute differences between positive and negative training examples. DeepBind [35], a convolutional network for motif inference, uses dinucleotide shuffling [115] to create negative training examples. That is, DeepBind takes sequences known to have a motif (positive training examples) and randomly permutes the nucleotide order of each sequence in such a way that the dinucleotide frequency ("AA", "AC", "AG",..."TT") is maintained.

The effect of the differences between positive and negative training examples on motif inference has not yet been investigated. A simple method for adjusting this difference could be implemented by generating negative examples based on positive examples in such a way that only a certain fraction of all nucleotides are randomly shuffled.

Methods

48 randomly selected data sets published by the DeepBind [35] authors were selected. From their published positive training examples, either the first 5, 20 or 100 sequences were used as positive training data. Negative training examples were created by randomly shuffling every n-th nucleotide so that, on average, either 20, 35, 50, 65 or 80% of nucleotides were permuted in each sequence. The test data sets were the original data sets published by the DeepBind authors. A convolutional neural network with 3 circular filters of length 8 was trained to discriminate between positive and negative examples by minimizing the cross-entropy between predictions and targets. The models were trained for $40,000$ steps at a learning rate of 0.01. Then, performance on the test sets was assessed.

Results

The results show that reduced shuffling frequencies did not improve motif inference with convolutional neural networks.

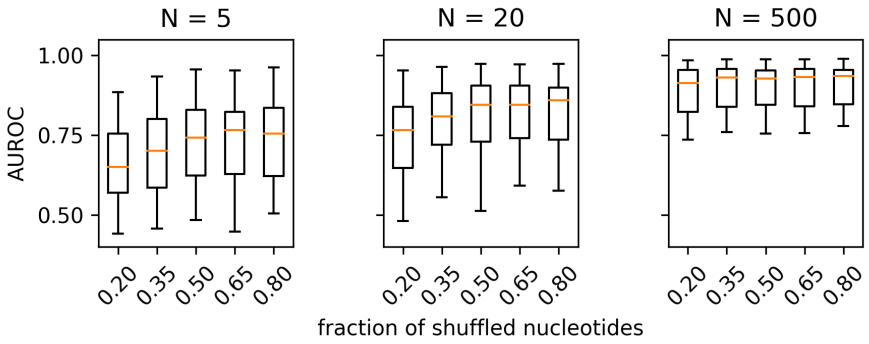


Figure 14: Effect of the amount of random nucleotide shuffling on sequence classification performance for different numbers (N) of positive training examples. Negative examples were created based on positive examples by randomly shuffling every n-th nucleotide so that a certain overall shuffling rate was achieved.

Discussion

The fact that reduced shuffling frequencies did not improve motif inference performance could mean that the investigated motifs here were not embedded in a structured background. However, since detailed knowledge about the sequence background is missing, there is not enough evidence in support of this explanation. In order to gain evidence from future analyses, simulated data sets could be investigated that allow creation of well-defined background properties.

3 Learning the determinants of bacterial translational efficiency

3.1 Summary of the obtained results

Translational efficiency, the phenomenon that synonymous mRNAs are translated at different rates, has been the subject of studies for more than 30 years. Several large-scale data sets have recently been produced that aiming to reveal the relationship between mRNA sequence and translational efficiency. Despite these efforts, a precise map between sequence and translational efficiency has not yet been found.

In section 3.2, a novel algorithm is presented that combines folding energies as sequence-derived features and yields more accurate predictions of translational efficiency than established methods for sequences from the same distribution. Moreoever, this algorithm correctly predicted that two synonymous substitutions in a sequence would result in a 15-fold change in its translational efficiency.

In the appendix to this manuscript (section 3.3), an analysis of the underlying dataset is described. Evidence is presented that the translational efficiency values were not properly normalized*, presumably as a consequence of the high-throughput method with which the data set was created. The inability to estimate noise levels in the data set and the limited sequence variability as well as a bias resulting from hand-designed constructs could impede the generalization ability of methods trained on this data set.

To date, high-throughput technologies have resulted in more than 280 million publicly available bacterial gene sequences. Unfortunately, labels indicating associated translational efficiencies are typically missing for these sequences, preventing direct supervised learning of translational efficiency. In section 3.4, evidence is presented indicating that ribosomal genes alone cannot not be used as a proxy for highly translationally efficient genes: ribosomal genes are highly expressed and thus highly conserved, resulting in insufficient sequence variation to learn determinants of translational efficiency. This suggests that either increasing the sequence variability by incorporating more sequences corresponding to families of highly expressed genes or utilization of unsupervised learning methods could solve the problem.

*This does not change the results of section 3.2; it merely explains some of the noise in the dataset.

3.2 Manuscript II: Predicting gene expression changes in E. coli from mRNA sequence information

Authors: L. Zhao, N. Abedpour, C. F. Blum, P. Kolkhof, M. Beller, M. Kollmann & E. Capriotti

Contribution of Christopher Blum:

- verified reproducibility of an underlying data set
- provided descriptive statistical analysis of the data

Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Gene Expression

Predicting gene expression level in *E. coli* from mRNA sequence information

Linlin Zhao[1], Nima Abedpour[2], Christopher Blum[1], Petra Kolkhof[1], Mathias Beller[3], Markus Kollmann[1,*] and Emidio Capriotti[4,*]

[1]Institute for Mathematical Modeling of Biological Systems, Department of Biology. Heinrich Heine University D sseldorf. Universitaetsstr. 1, 40225 D sseldorf, Germany. [2]Department of Translational Genomics, University of Cologne, Weyertal 115b, 50931 Cologne, Germany. [3]Systems Biology of Lipid Metabolism, Department of Biology. Heinrich Heine University D sseldorf. Universitaetsstr. 1, 40225 D sseldorf, Germany. [4]BioFolD Unit, Department of Pharmacy and Biotechnology (FaBiT),, University of Bologna, Via F. Selmi 3, Bologna, 40126, Italy.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: The accurate characterization of the translational mechanism is crucial for enhancing our understanding of the relationship between genotype and phenotype. In particular, predicting the impact of the genetic variants on gene expression will allow to optimize specific pathways and functions for engineering new biological systems. In this context, the development of accurate methods for predicting the translation efficiency and/or protein expression from the nucleotide sequence is a key challenge in computational biology.

Methods: In this work we present *PGExpress*, a new regression method for predicting the log2-fold-change of the translation efficiency of an mRNA sequence in *E. coli*. *PGExpress* algorithm takes as input 12 features corresponding to the predicted RNA secondary structure and anti-Shine-Dalgarno hybridization free energies. The method was trained on a set of 1,772 sequence variants (WT-High) of 137 essential *E. coli* genes. For each gene, we considered 13 sequence variants of the first 33 nucleotides encoding for the same amino acids followed by the superfolder GFP. Each gene variant is represented sequence blocks that include the Ribosome Binding Site (RBS), the first 33 nucleotides of the coding region (C33), the remaining part of the coding region (CC), and their combinations.

Results: Our gradient-boosting-based tool (*PGExpress*) was trained using a 10-fold gene-based cross-validation procedure on the WT-High dataset. In this test *PGExpress* achieved a correlation coefficient of 0.57, with a Root Mean Square Error (RMSE) of 1.4. When the regression task is cast as a classification problem, *PGExpress* reached an overall accuracy of 0.73 a Matthews correlation coefficient 0.47 and an Area Under the Receiver Operating Characteristic Curve (AUC) of 0.80. When compared with *RBSCalculator, PGExpress* results in better performance in the prediction of the log2-fold-change of the translational efficiency and its variation on the WT-High dataset. Finally, we validated our method by performing in-house experiments on five newly generated mRNA sequence variants. The predictions of the expression level of the new variants are in agreement with our experimental results in *E. coli*.

Availability: http://folding.biofold.org/pgexpress

Contact: markus.kollmann@hhu.de, emidio.capriotti@unibo.it

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The ability to predict translation efficiency in bacteria is important to define the relation between genotype and phenotype, and to engineer new organisms optimized for producing biomaterials (Kyle, et al., 2009), fuels (Toone and de Winde, 2013) and natural products (Krivoruchko and Nielsen, 2015). The information to regulate the translation process is encoded in the mRNA nucleotide sequence. The preference for specific combinations of nucleotides in the coding region, which refers to codon bias, has a strong effect on protein expression and formation (Li, et al., 2012; Mortimer, et al., 2014; Plotkin and Kudla, 2011; Pop, et al., 2014). Changes in the nucleotide sequence and codon usage can affect the mRNA folding process, which is a key determinant of protein expression. The ability of RNA strands to fold and form stable structures influences all the steps of the translation process: the structures at untranslated regions (UTR), especially at ribosome binding sites (RBS), act as a barrier for the ribosome to dock on the transcripts, then slow down the translation initiation (Duval, et al., 2013); the local structures in coding sequences (CDS) interplays with tRNA abundance to smoothen the translation elongation (Gorochowski, et al., 2015); structures also affect mRNA localization and turnover (Mortimer, et al., 2014). The Shine-Dalgarno (SD) sequence encoded in the mRNA is another key factor for translation regulation. Indeed, when the SD sequence is located in untranslated regions (UTRs), it promotes the binding of ribosomes and accelerates translational initiation (Kozak, 2005; Shine and Dalgarno, 1974; Shultzaberger, et al., 2001). Contrarily, its presence in the coding region can reduce the translational elongation rate in bacteria (Li, et al., 2012). Thus, the understanding of the mechanism of bacterial translation will result in accurate predictions of protein expression from mRNA sequence (Gingold and Pilpel, 2011). In this work we primarily considered the measure of translation efficiency, which provides a quantitative estimation of the of translation process, independent from the transcription. The translation efficiency is defined as the ratio of protein to mRNA abundance, which corresponds to the amount of protein produced by a single molecule of mRNA (Tuller, et al., 2010a; Tuller, et al., 2010b).

In the past, many studies and software tools have been developed for predicting protein expression based on mRNA sequence. Tools to tailor the untranslated region (UTR) to achieve a desired protein expression level were also introduced (Na and Lee, 2010; Reeve, et al., 2014; Rodrigo and Jaramillo, 2014; Seo, et al., 2014). For instance, the RBS calculator (Salis, 2011), UTR designer (Seo, et al., 2014), and RBS designer (Na and Lee, 2010) which are statistical models considering free energies for key molecular interactions in translation initiation and the formation of mRNA local structures provided an estimation of the translation efficiency. In general, the predictions from these methods show good correlation with their experimental data respectively. Recently Bonde and colleagues (Bonde, et al., 2016) studied the relationship between SD sequences and protein expression by measuring expression levels of ~3,000 UTRs in the presence of different SD variants. Their empirical method (EMOPEC) was able to predict the protein expression level of newly designed sequences in 91% of the cases. Focusing on the UTR regions, the available tools limit our understanding of the general picture of translational mechanism and our ability to engineer the whole mRNA molecule. Recently, Goodman and colleagues (Goodman, et al., 2013) measured the expression level of more than 14,000 synthetic gene variants in *E. coli* to quantify the effects of N-terminus codons as well as different combinations of promoter and Ribosome Binding Sites (RBSs). They found that rare codons in the N-terminus increased the stability of the RNA structure resulting in decreased gene expression level. The gene variants tested by Kosuri and co-workers (Goodman, et al., 2013) included variations in both UTR and coding sequences, which made the data suitable for investigating the effects from coding sequences as well. We make use of their data to capture regulatory factors from both the UTR and coding region of the mRNA molecule.

For estimating the contributions of different RNA regions on gene expression, we represented the sequences by the predicted global and local RNA folding free energies to define the main features contributing to the translation efficiency. Since mRNA structure impacts each step of translation (Kozak, 2005; Mortimer, et al., 2014), it represents one of the most important features to consider. The RNA folding free energy is a classical scoring function used for the prediction or RNA secondary structure. Many tools for predicting RNA secondary structure implement dynamic programming algorithm for minimizing the free energy (Capriotti and Marti-Renom, 2008). Many studies showed that different regions of mRNA preserve specific structural preferences (Kudla, et al., 2009; Mortimer, et al., 2014). Kudla and colleagues found that the predicted folding free energy of the first ~40 nucleotides of the mRNA has a significant correlation with the GFP protein abundance (Kudla, et al., 2009). In a recent study, it was observed that structures at the end of 5′ UTR and the beginning of 3′UTR are well conserved and the coding region is more structured than UTRs (Mortimer, et al., 2014). Thus, the free energy associated to the formation of local structures is also an important predictive feature. In addition, since the SD sequence shows different regulating effects, we also predicted the hybridization free energy (also referred as binding energy) between the anti-SD sequence and different regions of the mRNA. The predicted folding and hybridization free energies were combined to represent the translational features of the mRNA.

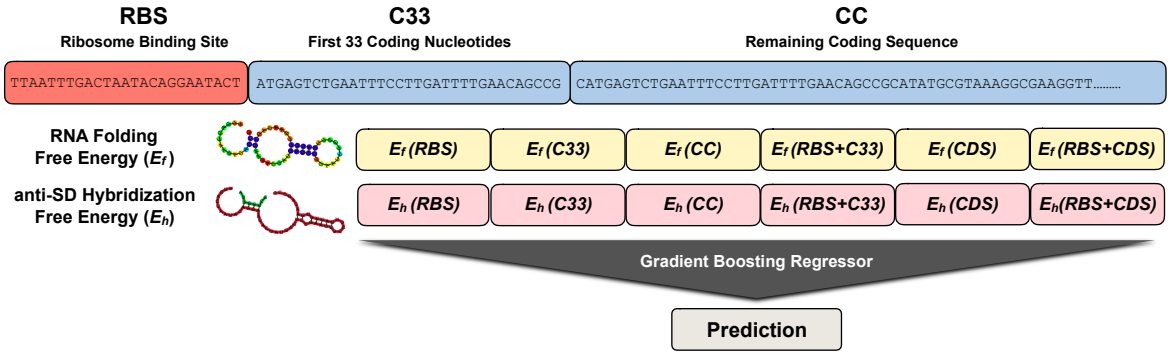
In this work we present *PGExpress* (***P***redicting ***G***ene ***Express***ion), a new gradient boosting-based algorithm predicting translation efficiency of mRNA sequences. *PGExpress* is a regression method that predicts the log2-fold-change of translation efficiency with respect to the median value observed experimentally. Our method relies on the calculation of the minimum RNA secondary structure free energy as representations of the local and global mRNA structures and the minimum free energy of hybridization between anti-SD sequence and mRNA, which corresponds to the binding affinity of the ribosome with different strands of mRNA. The performance of *PGExpress* has been tested on previously published datasets and new experimental data generated in-house.

2 Methods

2.1 Datasets

The data used in this work consists of protein expression and/or translation efficiency measures of genes and their variants in *E. coli*. The data was collected both from the literature (Goodman, et al., 2013) and experimental tests in our lab. The data from Kosuri and collaborators (Kosuri-All) is a collection of protein expression (PE) and translation efficiency (TE) measures from ~14,000 gene variants (Goodman, et al., 2013). More information about the gene expression measures considered in this work is reported in section 1 of the Supplementary Materials. Each variant is a combination of the Promoter with high and low strength (High, Low), the Ribosome Binding Site (Wild-Type, Weak, Mid and Strong RBSs) and the first 33 nucleotides of the coding region (C33) of 137 essential *E. coli* genes followed by the superfolder GFP (sfGFP) coding sequence (see Supplementary Materials, section Experimental data). From the Kosuri-All dataset we extracted five subsets (WT-High,

Method



For training and testing the regression algorithm the values of the protein expression and translation efficiency are converted in log2-fold-change with respect to their median values in the WT-High dataset (2,988 and 2,355 respectively). For evaluating the performance of the method as a binary classifier, the previous median values are used as classification thresholds. Finally, to test the performance of *PGExpress*, we measured in our lab the protein expression level of five randomly selected variants from the Kosuri-All dataset (Exp-Set). We used the Exp-Set to check the agreement between the data in Kosuri-All and our measures. Then, we generated a validation set, namely Exp-Mut, which is composed of new variants derived from the five sequences in Exp-Set. The sequences of the ten gene variants are reported in Table S1.

The Kosuri-All dataset analyzed in this work is provided in Supplementary File 1. A summary of its composition is reported in Table S2.

2.2 Algorithm description

Here we present a regression method (*PGExpress*) to predict the log2-fold-change of the gene translation efficiency (L2TE) from sequence information. *PGExpress* is based on gradient boosting regression algorithm that takes in input a 12-elements vector composed by six predicted RNA folding and six anti Shine-Dalgarno (SD) hybridization free energies per nucleotide. In detail, each gene variant is divided in three sequence blocks: the Ribosome Binding Site (RBS), which consists of ~25 nucleotides preceding the coding sequence, the first 33 nucleotides of the coding region (C33) and the remaining part of the coding sequence starting from nucleotide 34 (CC). Thus, each gene is represented by six sequence fragments including the three blocks previously defined (RBS, C33 and CC), and the combinations of RBS with C33 (RBS+C33), C33 with CC (CDS) and RBS with the whole coding sequence (RBS+CDS). For each block we predicted the RNA secondary structure and the anti-

2.3 Feature analysis

To estimate the predictive power of each feature, we calculated the linear regression between the RNA folding and anti-SD hybridization free energies of the five sequence blocks (RBS, C33, RBS+C33, CDS and RBS+CDS) and the log2-fold-change of the translation efficiency in the WT-High dataset. In this analysis we did not consider the C-terminal region of the coding sequence (CC) because it corresponds to the sfGFP for all the variants in the Kosuri-All dataset. Furthermore, we compared the performance of our best approach (*PGExpress*) against five methods including different combinations of the 12 input features. These methods are:

- **BFolding**: most discriminative RNA folding free energy
- **BBinding**: most discriminative anti-SD hybridization free energy
- **Folding6:** RNA folding free energies of the six blocks
- **Binding6:** anti-SD hybridization free energies of the six blocks.
- **BFoldBind:** most discriminative RNA folding and anti-SD hybridization free energies.

2.4 Algorithm optimization

PGExpress is based on a gradient boosting regression algorithm (GradientBoostingRegressor) implemented in the *scikit-learn* package (Pedregosa, et al., 2011). It has been optimized considering different numbers of estimators (10, 50, 100, 200 and 500) and maximum depth values for the regression estimator (1, 3, 5, and 7). The *scikit-learn* GradientBoostingRegressor class was run using the least squares regression as loss function and the default values for all the remaining parameters.

2.5 Training and testing

To estimate the performance of *PGExpress* and the alternative methods, we performed several tests. First, we tested *PGExpress* using a gene-based 10-fold cross-validation approach on the WT-High dataset to keep all the variants belonging to the same gene in the same subset. For each test we calculated the performance using the evaluation measures defined in section 2 of the Supplementary Materials. The reported scores represent the average values obtained over five 10-fold cross-validation tests. The results obtained on the Kosuri-All (Weak-High, Mid-High and Strong-High) and Experimental (Exp-Set, Exp-Mut) datasets were calculated after removing from the training set all the data related to the genes present in the testing set. This procedure reduced the overfitting due to the presence of data from sequences with high similarity both in training and testing sets. To check for this source of bias, we also performed the all-against-all global alignments (1,558,513) among the RBS+C33 regions of all the gene variants. The global alignments of the nucleotide sequences were calculated using the *align0* algorithm from the *fasta2.0* package (Myers and Miller, 1988).

2.6 Comparison with RBS Calculator

For assessing the quality of our predictions we compared our results with those obtained by *RBSCalculator* (Salis, et al., 2009). For the comparison we calculated the performances of the methods both in predicting the value (regression mode) and *sign* function (binary classifier) of the log2-fold-change of the translation efficiency. The predictions of *RBSCalculator* were scaled by calculating the log2-fold-change with respect to the median value of the translation efficiency on WT-High dataset (L2TE). A further comparison of the methods evaluated their performance in predicting the log2-fold-change with respect to the wild-type ($L2TE_{wt}$). For this task we scored the performance of *PGExpress* and *RBSCalculator* as binary classifiers excluding gene variants with absolute $L2TE_{wt}$ close to zero. More details about this test are reported in section 2.4 Supplementary Materials.

2.7 Engineering new testing sequences

For validating our algorithm, we generated new sequences and measured their protein expression level. In this case, considering the protein expression level, we reduced the complexity of the experiment that did not require to measure mRNA expression. Thus, we selected a subset of gene variants either with positive or negative log2-fold-change of protein expression (L2PE) with respect to its median value of the High-WT dataset (2,988). For checking the similarity between our experiments and those performed by Kosuri and colleagues (Goodman, et al., 2013), we measured the expression level of five randomly selected gene variants (Exp-Set) from High-WT dataset. In the next step, we generated five new sequences not included in the Kosuri-All dataset mutating at most one nucleotide in RBS or three codons in coding region. Finally, we randomly selected a set of five gene variants (Exp-Mut), three of which show a significant variation of the predicted L2PE ($|L2PE_{wt}| \geq 3$) either from positive to negative (*dapB* and *lpxK*) or negative to positive (*zipA*) and two cases (*lgt* and *murF*) where the expression level remains in the same class. The sequences of the ten tested gene variants are reported in Table S1.

2.8 Experimental expression measure

DNA sequences consisting of promoter, Ribosome Binding Site (RBS), and 33 coding nucleotides (including ATG start site) of five different genes were synthesized (Genscript, Piscataway, USA) with flanking AscI and NdeI restriction sites. The DNA fragments were excised from the shuttle vector and directionally cloned into the pJ251-GERC vector obtained from Addgene (Kosuri, et al., 2013). A unique EcoRI restriction site was engineered in between the 5′ region of the AscI site and the respective promoter sequence. Using the EcoRI site we identified the positive clones. Final gene variants were verified via Sanger sequencing. The correct variants were transformed in MG165 *E. coli* cells and starter cultures were grown over night at 37 °C. The next day cultures were diluted 1:1000 in 100 µL LB medium in optical quality black walled 96-well plates (PerkinElmer, Waltham, MA, USA) in quadruplicate and overlayed with 40 µL mineral oil. Bacteria were grown at 30 °C. Bacterial growth was followed by measuring the optical density at 600 nm (OD600) as proxy. The different combination of promoter, RBS, and coding region regulate the expression levels of the superfolder green fluorescent protein (sfGFP). Expression of the red fluorescent protein (mCherry) was controlled by a constitutive promoter (PLtetO-1) shared by all gene variants (Kosuri, et al., 2013). sfGFP and mCherry fluorescence levels were measured with a monochromator equipped BioTek Synergy Mx (BioTek, Winooski, USA) plate reader. Every five minutes a fluorescence measurement was performed.

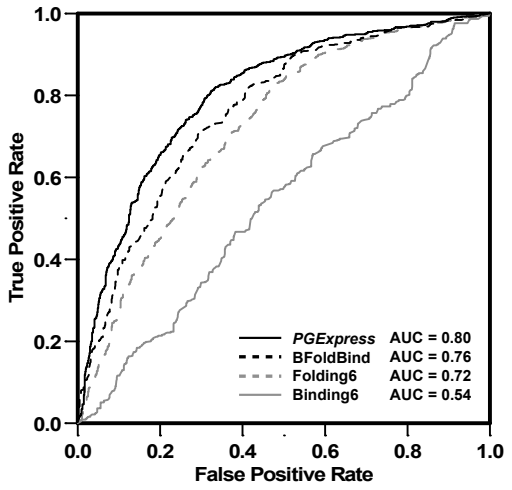
3 Results

3.1 Regression and input features

The selection of the data from Kosuri and co-workers allowed us to develop a machine learning method (*PGExpress*) for predicting the log2-fold-change of the translation efficiency based on the sequence information. Before performing our tests, we analyzed the Kosuri-All dataset and focused on the gene variants in the WT-High subset. This set is composed of sequences with promoter with high binding affinity (BBaJ23100) and wild-type RBSs (Ribosome Binding Sites). The choice of WT-High dataset is supported by the observation that the correlation between the level of protein and RNA expression is higher than in WT-Low dataset (Fig. S1). Indeed, the correlation coefficients are 0.72 and 0.51 in WT-High and WT-Low sets, respectively. Thus, we selected the WT-High subset as a main reference set for this work for estimating the predictive power of our machine learning approach. To avoid the overestimation of the performance we performed a gene-based 10-fold cross-validation test. Keeping the variants from the same gene in the same subsets, we excluded the presence of sequences with high level of identity both in training and testing. Thus, we calculated the distribution of the percentage of identity (PID) between the first two blocks (RBS+C33) of the different gene variants. The Fig. S2 shows that only ∼4% of the cases the PID achieved a value between 50% and 60%. To estimate the predictive power of the input features used in *PGExpress*, we performed a linear regression analysis to calculate the correlation between each feature and the log2-fold-change of the translation efficiency (L2TE). The Tables S3 and S4 report the correlation coefficient (r), the root mean square error (RMSE) and the mean absolute error (MAE) obtained fitting the experimental L2TE with the predicted values of RNA secondary structure and anti-Shine-Dalgarno (anti-SD) hybridization free energies. This analysis revealed that overall the free energies of the RBS+C33 sequences resulted in the highest correlation with the log2-fold-change of translation efficiency (L2TE). while, the anti-SD hybridization free energy of the RBS shows highest negative correlation among the binding features (Table S4).

new plots

nce



ls

Method	r	RMSE	MAE	ACC	MC	AUC	N
BFolding	0.39	1.53	1.23	0.67	0.35	0.72	1
BBinding	0.10	1.67	1.40	0.51	0.01	0.52	1
Folding6	0.40	1.54	1.22	0.67	0.35	0.72	6
Binding6	0.08	1.73	1.43	0.52	0.04	0.54	6
BFoldBind	0.50	1.44	1.14	0.70	0.40	0.76	2
PGExpress	0.57	1.38	1.08	0.73	0.47	0.80	12

r, RMSE, MAE, ACC, MC and AUC are defined in Supplementary Materials. N is the number of input features. The input features of BFolding, BBinding, Folding6, Binding6, BFoldBind and *PGExpress* are defined in the section Features analysis.

3.2 Performance with different features

In a second step, we calculated the performance of *PGExpress* and five alternative methods, which included a reduced number of features. The input features for the BFolding, BBinding, Folding6, Binding6 and BFoldBind were described in the section Feature Analysis. In Table 1 we reported the scores of the previous six methods on the WT-High dataset using the gene-based 10-fold cross-validation procedure. The results revealed that the RNA folding free energy corresponding to the RBS+C33 portion of the variants is the most informative feature. Indeed the BFolding method with only one feature reached a correlation coefficient (r) of 0.39. When regression values are converted in binary classification predictions BFolding method achieved an overall accuracy (ACC) of 0.67 a Matthews Correlation Coefficient (MC) of 0.35 and Area Under the Receiver Operating Characteristic Curve (AUC) of 0.72.

The discriminative power of the anti-Shine-Dalgarno (anti-SD) binding free energy is much lower. This is evident by measuring the performance of the BBinding method that resulted in low *r* and MC. The analysis of the results of the Folding6 and Binding6 methods, which include six features of the same type free energy, do not show any substantial increase in the performances with respect to the BFolding and BBinding methods. An improvement in the performance is obtained combining the RBS+C33 RNA secondary structure free energy with the RBS anti-SD hybridization free energy. Indeed the BFoldBind method, which takes in input two features, reached a *r* of 0.5 and AUC of 0.76.

Table 2. Performance of the *PGExpress* on the Kosuri-All subsets.

Dataset	r	RMSE	MAE	ACC	MC	AUC	High
WT-High	0.57	1.37	1.08	0.73	0.47	0.80	0.50
Weak-High	0.66	1.16	0.94	0.77	0.55	0.85	0.53
Mid-High	0.58	1.33	1.02	0.85	0.41	0.84	0.81
Strong-High	0.49	1.40	1.10	0.92	0.47	0.81	0.89

r, RMSE, MAE, ACC, MC and AUC are defined in Supplementary Materials. High is the fraction of gene variants with translation efficiency higher than its median value on the WT-High dataset (L2TE>0).

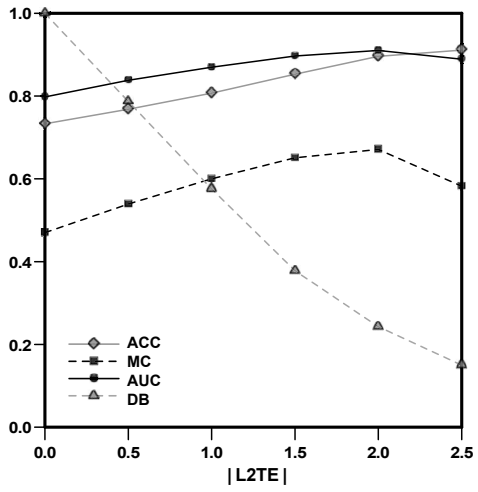
In *PGExpress* we merged the six RNA folding and six anti-SD hybridization free energies. The results in Table 1 show that *PGExpress* achieved a correlation coefficient of 0.57, ACC of 0.73, MC of 0.47 and AUC 0.80 improving the *r* value and the Matthews correlation coefficient of 0.07, and the AUC of 0.04 with respect to BFoldBind. The Receiver Operating Characteristic (ROC) curves for all methods are plotted in Fig. 2. The *PGExpress* method also resulted in lowest values of root mean square error and mean absolute error with are 1.38 and 1.08 respectively. The optimal performance of the *PGExpress* algorithm is obtained considering a maximum depth 5 and 50 estimators (see section 2.4). The results of the optimization procedure are summarized in Table S5.

3.3 Performance on the Kosuri-All subsets

In the next test we focused on the performance of *PGExpress* on three datasets (Weak-High, Mid-High and Strong-High), which contain gene variants with the same 33 starting nucleotides in the coding regions (C33) but three different RBSs (Ribosome Binding Sites). Analyzing the three new datasets, we observed that the distribution of the translation

e
a
2,
o
W
p

new plots



Fi
an
fil

overall accuracy (ACC) and Matthews correlation coefficient (MC) on the Weak-High. Indeed on this dataset *PGExpress* resulted in the highest value of r, MC and AUC which are 0.66, 0.55 and 0.85 respectively. Due to the dataset unbalance, lowest r and highest ACC are obtained on the Strong-High dataset (Table 2).

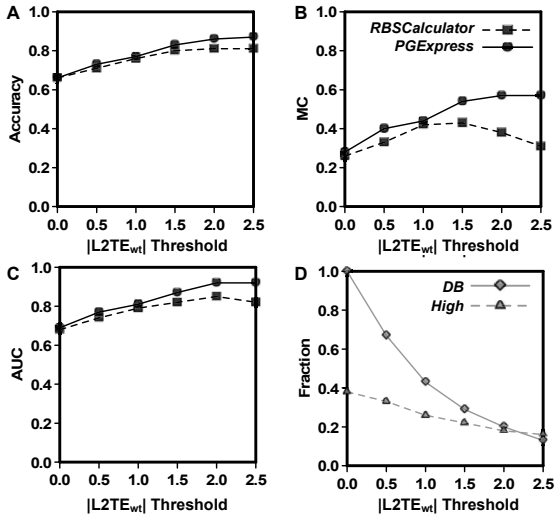
3.4 Selecting high-quality predictions

To better characterize the performance of *PGExpress*, we scored our method filtering-out the less reliable training data and predictions in WT-High dataset. We assumed that gene variants with translation efficiency near the median (M(TE)=2355) constitute the noisy part of the dataset. Thus, we filtered-out progressively the subset of data with absolute log2-fold-change value below a selected threshold (see section 2.3 in Supplementary Materials). The performances of *PGExpress* in binary classification mode after removing the data close to the median value are reported in Fig. 3 and Table S6. We observed that removing 42% of the gene variants with absolute log2-fold-change of the TE lower than 1, *PGExpress* reached an overall accuracy of 0.81 and an AUC of 0.87.

3.5 Comparison with *RBSCalculator*

We compared the performance of *PGExpress* with *RBSCalculator* on the WT-High dataset. The results showed that *PGExpress* reached higher correlation coefficient (r) and Matthews correlation (MC) than *RBSCalculator* (see Table 3). Small difference is observed in the comparison of the value of Area Under the ROC Curve which is ~0.8 for both methods.

Table 3. Comparison between *PGExpress* and *RBSCalculator*.



PGExpress with *RBSCalculator* were also compared calculating their performance in predicting the log2-fold-change with respect to the wild-type sequence ($L2TE_{wt}$) removing gene variants with absolute $L2TE_{wt}$ value below a given threshold. The results show that selecting the subset of variants $|L2TE_{wt}| \leq 0.5$ *PGExpress* reached higher AUC and Matthews Correlation coefficient than *RBSCalculator* (see Fig 4).

3.6 Test on our experimental dataset

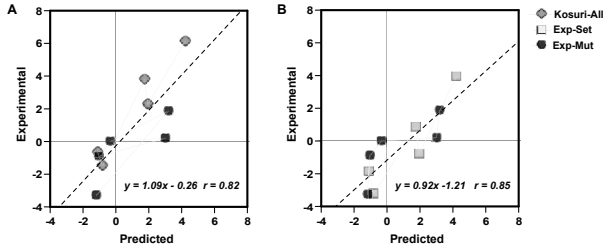
To test the ability of *PGExpress* to predict the gene expression we performed in-house experiments with five gene variants each in the Exp-Set and Exp-Mut datasets (see methods section) and measured the protein expression using the protocol introduced by Kosuri and co-workers (Goodman *et al.*, 2013). In Fig. S4 we plotted the measures of the fluorescence associated to each gene variant normalized by the maximum level of OD600. To make a fair comparison between our results and those reported by Kosuri and collaborators, we used the median value of the protein expression level in Kosuri data as threshold for discriminating between low and high expressed gene variants. Thus, we compared the maximum value of the re-scaled fluorescence (Table S8 and Fig. S5) obtained in our experiment with the median protein expression level in the WT-High dataset (2,998). According to this assumption, we verified that for four gene variants over five (Exp-Set), our experiments match those performed by Kosuri and colleagues (Table 4). The only difference is observed for a variant of the *lgt* gene (lgt-23), which is classified to have a protein expression higher than the median value in the Kosuri-All dataset, whereas our experiments revealed a low protein expression. Nevertheless the prediction of *PGExpress* agrees with the results reported in Kosuri-All dataset. Finally we evaluate the accuracy of *PGExpress* predictions in classification mode on the Exp-Mut dataset, verifying that our predictions are correct for all the new gene variants.

the protein expression level for the gene variants
Exp-Mut datasets.

osuri-All	Experiment	Prediction	Class
3.8	0.8	1.7	↑
2.3	-0.8	2.0	↑°
6.1	3.9	4.2	↑
-0.7	-1.9	-1.1	↓
-1.5	-3.2	-0.8	↓
-	-3.3	-1.2	↓
-	1.9	3.2	↑
-	0.0	-0.3	↓*
-	-0.9	-1.0	↓
-	0.2	3.0	↑

ge of protein expression (L2PE) with respect to its Kosuri's dataset (Goodman, et al., 2013). Experiment: l from protein expression levels from our in-house edicted L2PE of protein expression from *PGExpress*. log2-fold-change of protein expression. ↑ and ↓ repre- e and negative values of the L2PE. *Our experimental ression for the lgt-23 gene variant is in disagreement aset. ° The prediction of log2-fold-change of protein ariant is in agreement with the experimental measure e experimental value of L2PE for lpxK-Mut is slightly nces of all variants are reported in Table S1. The results e for predicting L2PE are reported in Table S9.

Fit 1



predicted to have low expression level and, our in-house measure of protein expression (2,996), is only few digits below the median value of protein expression (2,998). Comparing the experimental and predicted value of the log2-fold-change of protein expression, *PGExpress* achieved a correlation coefficient 0.82 and 0.85 when the predictions on Exp-Mut dataset are merged respectively with the predictions on Kosuri-All (Figure 5A) and Exp-Set (Figure 5B) datasets. Our results show that in both cases there is a strong correlation between *PGExpress* prediction and the experimental data. For this specific task we trained the *PGExpress* algorithm on the log2-fold-change of protein expression with respect to its median valued (L2PE) from Kosuri's dataset. The results of the optimization procedure for the prediction of the log2-fold-change of protein expression are summarized in Table S9.

4 Discussion

In this work we presented *PGExpress*, a gradient boosting regression method for predicting the log2-fold change of the translation efficiency of mRNA from predicted free energy features. The method uses the folding free energies of six sequence blocks which represent the local and global mRNA folding structures. The six blocks include RBS, C33, CC sequence and their combinations. Among them the predicted folding energy the block of RBS+C33 provides the most informative feature. This is in agreement with previous findings that the folding structure around starting codon has a strong effect on translation. By adding folding free energies from other blocks, the prediction accuracy increased. This might indicate that, although other regions of the gene have an impact on translation, the structure of the 5' region constitutes the main contribution to the translation rate. For instance, the presence of a folded SD sequence near a starting codon might slow down the translation process reducing the probability of the ribosomes to bind or elongate. Accordingly, the minimum hybridization free energies were used to represent the effect of the SD sequence predicting the hybridization energy between the mRNA and the anti-SD sequence. Although the minimum hybridization free energy itself shows a weak correlation with the translation efficiency, the interplay between all folding and hybridization free energies allowed to improve the performance of our predictor. This indicates that the mRNA structures and SD sequences regulate translation in a cooperating manner.

Thus, our results show that optimized version of *PGExpress* reaches a correlation coefficient of 0.57 in regression mode and an AUC of 0.80 as binary classifier. A comparative assessment of predictions revealed that *PGExpress* achieved better performances than *RBSCalculator* in the prediction of the log2-fold-change of the translation efficiency and its variation with respect to the wild-type,

', we test the sensitivity of *PGExpress* to small changes in the
tide sequences. For this purpose we measured the expression level
gene variants that differ in few nucleotides from the original
ces from Kosuri-All dataset. Our analysis show that *PGExpress* is
correctly predict the expression level of all new variants, most of
(4/5) resulted in an opposite expression level with respect to the
l sequence. Strikingly, is the case of the *dapB* variant which
ed >15-fold lower protein expression with only 2 synonymous
ons (see Tables S1 and S8). This observation confirms the robust-
f our method, which supports its practical application in biotech-
. Compared with other methods that are merely focusing on the
of UTRs, we integrated the main effecting factors from the per-
e of whole sequence, which enabled us to predict translation
ncy accurately and to engineer new sequences at the whole sequence level.

Future directions of our work will include the analysis of new features to improve the prediction of the translation efficiency of wild-type genes in *E. coli*, and the development of tools for identifying key nucleotides to control protein expressions. We believe that our *in-silico* approach can have strong impact on biotechnological applications reducing the experimental effort to engineer optimized organisms.

Acknowledgements

E.C. acknowledges Genifx - Genome Informatics Service at the University of Alabama at Birmingham, AL (USA) for computational resources. We acknowledge Sriram Kosuri, George Church and collaborators for sharing their experimental data.

Funding

E.C. acknowledges the Institute for Mathematical Modeling of Biological Systems at the University of Düsseldorf (Germany) for financial support.

Conflict of Interest: none declared.

References

Bonde, M.T., *et al.* Predictable tuning of protein expression in bacteria. *Nat Methods* 2016;13(3):233-236.

Capriotti, E. and Marti-Renom, M.A. Computational RNA structure prediction. *Curr Bioinform* 2008;3(1):32-45.

Gingold, H. and Pilpel, Y. Determinants of translation efficiency and accuracy. *Mol Syst Biol* 2011;7:481.

Goodman, D.B., Church, G.M. and Kosuri, S. Causes and effects of N-terminal codon bias in bacterial genes. *Science* 2013;342(6157):475-479.

Kosuri, S., *et al.* Composability of regulatory sequences controlling transcription and translation in Escherichia coli. *Proc Natl Acad Sci U S A* 2013;110(34):14024-14029.

Kozak, M. Regulation of translation via mRNA structure in prokaryotes and eukaryotes. *Gene* 2005;361:13-37.

Krivoruchko, A. and Nielsen, J. Production of natural products through metabolic engineering of Saccharomyces cerevisiae. *Curr Opin Biotechnol* 2015;35:7-15.

Kudla, G., *et al.* Coding-sequence determinants of gene expression in Escherichia coli. *Science* 2009;324(5924):255-258.

Kyle, S., *et al.* Production of self-assembling biomaterials for tissue engineering. *Trends Biotechnol* 2009;27(7):423-433.

Li, G.W., Oh, E. and Weissman, J.S. The anti-Shine-Dalgarno sequence drives translational pausing and codon choice in bacteria. *Nature* 2012;484(7395):538-541.

Lorenz, R.*, et al.* ViennaRNA Package 2.0. *Algorithms Mol Biol* 2011;6:26.

Mortimer, S.A., Kidwell, M.A. and Doudna, J.A. Insights into RNA structure and function from genome-wide studies. *Nat Rev Genet* 2014;15(7):469-479.

Myers, E.W. and Miller, W. Optimal alignments in linear space. *Comput Appl Biosci* 1988;4(1):11-17.

Na, D. and Lee, D. RBSDesigner: software for designing synthetic ribosome binding sites that yields a desired level of protein expression. *Bioinformatics* 2010;26(20):2633-2634.

Pedregosa, F.*, et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 2011;12:2825-2830.

Plotkin, J.B. and Kudla, G. Synonymous but not the same: the causes and consequences of codon bias. *Nat Rev Genet* 2011;12(1):32-42.

Pop, C.*, et al.* Causal signals between codon bias, mRNA structure, and the efficiency of translation and elongation. *Mol Syst Biol* 2014;10:770.

Reeve, B.*, et al.* Predicting translation initiation rates for designing synthetic biology. *Front Bioeng Biotechnol* 2014;2:1.

Rodrigo, G. and Jaramillo, A. RiboMaker: computational design of conformation-based riboregulation. *Bioinformatics* 2014;30(17):2508-2510.

Salis, H.M. The ribosome binding site calculator. *Methods Enzymol* 2011;498:19-42.

Salis, H.M., Mirsky, E.A. and Voigt, C.A. Automated design of synthetic ribosome binding sites to control protein expression. *Nat Biotechnol* 2009;27(10):946-950.

Seo, S.W.*, et al.* Predictive combinatorial design of mRNA translation initiation regions for systematic optimization of gene expression levels. *Sci Rep* 2014;4:4515.

Shine, J. and Dalgarno, L. The 3'-terminal sequence of Escherichia coli 16S ribosomal RNA: complementarity to nonsense triplets and ribosome binding sites. *Proc Natl Acad Sci U S A* 1974;71(4):1342-1346.

Shultzaberger, R.K.*, et al.* Anatomy of Escherichia coli ribosome binding sites. *J Mol Biol* 2001;313(1):215-228.

Toone, E. and de Winde, H. Energy biotechnology in 2013: advanced technology development for breakthroughs in fuels and chemicals production. *Curr Opin Biotechnol* 2013;24(3):367-368.

Tuller, T.*, et al.* An evolutionarily conserved mechanism for controlling the efficiency of protein translation. *Cell* 2010a;141(2):344-354.

Tuller, T.*, et al.* Translation efficiency is determined by both codon bias and folding energy. *Proc Natl Acad Sci U S A* 2010b;107(8):3645-3650.

3.3 Appendix: A note on a large-scale data set related to translational efficiency

Motivation

Some of the results of section 3.2 were based on a previously published data set that contains estimates of the translational efficiencies of certain constructs [98]. Translational efficiency can be quantified as the number of proteins that can be produced from one mRNA molecule. In the above-mentioned study, translational efficiency was estimated indirectly with a novel high-throughput method rather than a direct measurement.

In short, the constructs (figure 15) in this data set consisted of a promoter, a ribosome binding site, a codon variant sequence, a green fluorescent protein (GFP) gene sequence and an independently expressed RFP at a different location on the construct. The promoters were either strong ("high") or weak ("low"). The ribosomal binding site (RBS) was either weak, medium or strong or taken from one of 137 essential *E. coli* genes. The length of the RBS was between 18 and 21 nucleotides. The codon variants consisted of 12 synonymous variants of the first 11 codons (33 nucleotides) of the before-mentioned 137 essential *E. coli* genes, resulting in a total of 137×13 (including the WT variant) codon variant sequences. The ratio between GFP and RFP fluorescence was used to sort cells into 11 bins using flow cytometry, and read counts (RNA scores) of constructs in each bin were obtained with RNA sequencing. Based on the number of read counts in each bin, the GFP expression could be estimated (protein score). However, the information about the cell's exact GFP/RFP fluorescence ratios in each bin was lost and had to be estimated using average fluorescence intensities from the other bins.

To test if the data was properly normalized, the translational efficiency scores can be compared to the RNA scores. With proper normalization, the translational efficiencies should be independent of the RNA scores, and prediction of translational efficiency should not improve when RNA scores are incorporated as features.

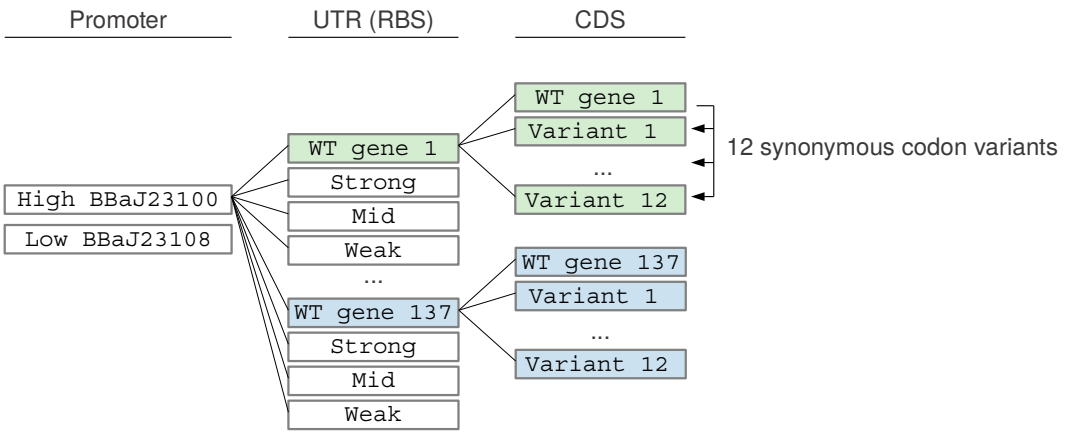


Figure 15: Composition of the constructs used by a large-scale study that investigated N-terminal codon bias[98].

Methods

Only data corresponding to the "high" promoter was used in the following analyses.

A model was trained to discriminate between sequences with translational efficiencies above or below a certain threshold (binary classification problem). The model consisted of a linear map that mapped binary input features to class-conditional probabilities. The translational efficiencies threshold was defined as the median translational efficiency of all constructs. A total of 61 binary input features were created, corresponding to features designated to RNA score, RBS and CDS sequences (figure 16). The first 51 input features corresponded to the last 18 nucleotides of the RBS and all nucleotides of the CDS, respectively. All nucleotides were converted into one-hot representation. The remaining 10 input features corresponding to the RNA score were created as follows: all training data set RNA scores were used to define 10 consecutive intervals such that approximately 10 % of all RNA scores fell into each interval. Thus, the intervals corresponded to the consecutive 10 % percentiles of the RNA score distribution. The interval that a particular construct was assigned to then defined which of the first 10 input features was set to "on".

Test data sets were generated by randomly choosing 137 examples (without replacement) from the set of all examples corresponding to the "high" promoter. All remaining examples (corresponding to the "high" promoter) were assigned to the training sets. As the sequence constructs corresponded to combinations of UTR and CDS sequences, some partial constructs occured in both training and test data sets, resulting in data sets that were not independent. To account for this, one special test set was generated with examples containing only WT RBS and WT CDS sequences, and the corresponding training set containing no WT sequences. A total of 30 training and test sets were generated. Networks were trained by minimizing the cross-entropy between actual and predicted targets at a learning rate of 0.001 and batch site of 128 for 3000 steps.

To assess an effect of RNA scores on predictive performance, two setups for network training were investigated: one in which the one-hot coding of RNA scores ($\log_{10}$) was randomly shuffled, and one in which it was left intact. This way, the number of parameters was kept the same in both setups. The final test accuracies (excluding the "WT only" test set) were statistically tested for difference using a Wilcoxon-Mann-Whitney test.

To investigate a correlation between translational efficiency estimates and RNA scores, the coefficient of correlation was calculated for the base 10 logarithms of both quantities.

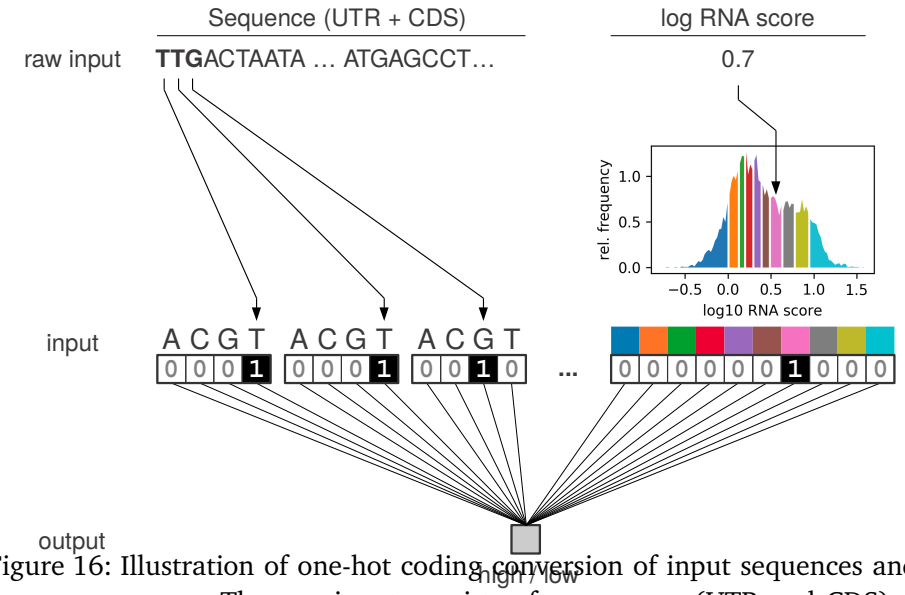


Figure 16: Illustration of one-hot coding conversion of input sequences and discretization of RNA scores. The raw input consists of a sequence (UTR and CDS) and the measured RNA score. The RNA score is discretized into one-hot coding according to the percentiles of the overall RNA score distribution in the training data set.

Results

Incorporation of RNA scores significantly improved predictive performance on the test data sets by approximately 4 percent ($p < 10^{-5}$, Wilcoxon-Mann-Whitney test based on 30 independent trainings). Training with RNA scores also improved predictions of test sequences that were completely different (WT RBS and WT CDS) from the training sequences. The logarithms of RNA scores and translational efficiency estimates were correlated ($\rho = 0.65$, figure 17).

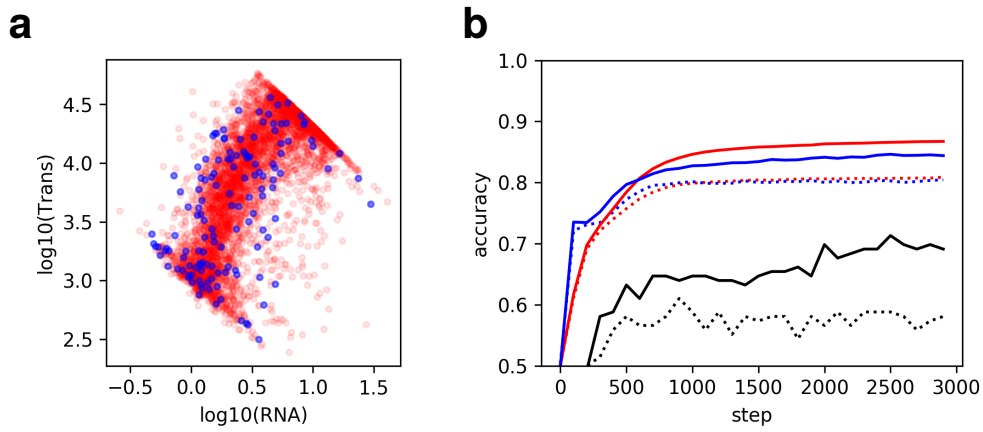


Figure 17: **a** Scatter plot of RNA scores ($\log_{10}$) and translational efficiencies ($\log_{10}$). The quantities are correlated with $\rho = 0.65$. Blue dots indicate constructs in which both UTR and CDS corresponded to wild type sequences, red dots indicate constructs in which either UTR and/or CDS were designed by the researchers.
b Training (line) and test (dotted) accuracies for classifying sequences into having either high or low translational efficiency. Red: both sequence and RNA scores were used as input features; blue: sequences and shuffled RNA scores were used as features; black: only sequences and RNA scores were used that corresponded to constructs in which both UTR and RBS were wild type sequences.

Discussion

The results suggest that the translational efficiency estimates were not properly normalized to the RNA scores. Otherwise, incorporation of RNA scores would not have improved predictive performance. Improper normalization could result from the discretization step of the high-throughput method with which the data set was generated (assigning constructs to bins based on GFP/RFP fluorescence ratio).

3.4 Scanning bacterial genes for determinants of translational efficiency

Motivation

To date, the genomes of more than 100,000 bacteria are publicly available from NCBI, but data regarding gene expression is missing for most of these genomes. In order to learn determinants for translational efficiency from these genomes with supervised learning, it is necessary to use other information about how strongly certain genes are expressed.

Ribosomal genes are among the most strongly expressed genes[96], and it seems reasonable to assume that ribosomal genes are generally enriched in sequence features resulting in high translational efficiency. Consequently, it should be possible to learn these sequence features using machine learning. The existence of sequence information is evident from the nucleotide frequencies of genes (figures 18 and 19). For example, there are codons that are preferentially used at the start of genes. The key difficulty is to learn only the sequence features relevant for translational efficiency and not other features unique to ribosomal sequences. If the sequence variation in ribosomal sequences were too small, it the machine learning algorithm would simply memorize ribosomal sequences. A prior investigation of the sequence similarity would be necessary, however this is a daunting task in itself given that there are more than 100,000 bacterial genomes. Control for overfitting would hence need to be achieved in a different way. Given the knowledge that ribosomal genes are more different from each other than to their homologs in other species[116], it should be possible to control for overfitting by training on one set of homologs and testing on the other.

Additional prior knowledge about translational efficiency comes from the fact that the copy number of ribosomal RNAs is positively correlated with a bacterium's growth rate [117]. This suggests that transcription rates become a limiting factor at high growth rates. It seems plausible that organisms with high growth rates should also possess genes with optimized translational efficiencies.

Since a positive correlation between mRNA folding energy and translational efficiency has been observed for artificial sequences [98], the hypothesis that ribosomal sequences have higher folding energy than other genes also seems plausible.

In the following, it is investigated if ribosomal genes can be used to learn translational efficiency. In addition to that, it is investigated if ribosomal genes have higher folding energies (fold less strongly) than other genes.

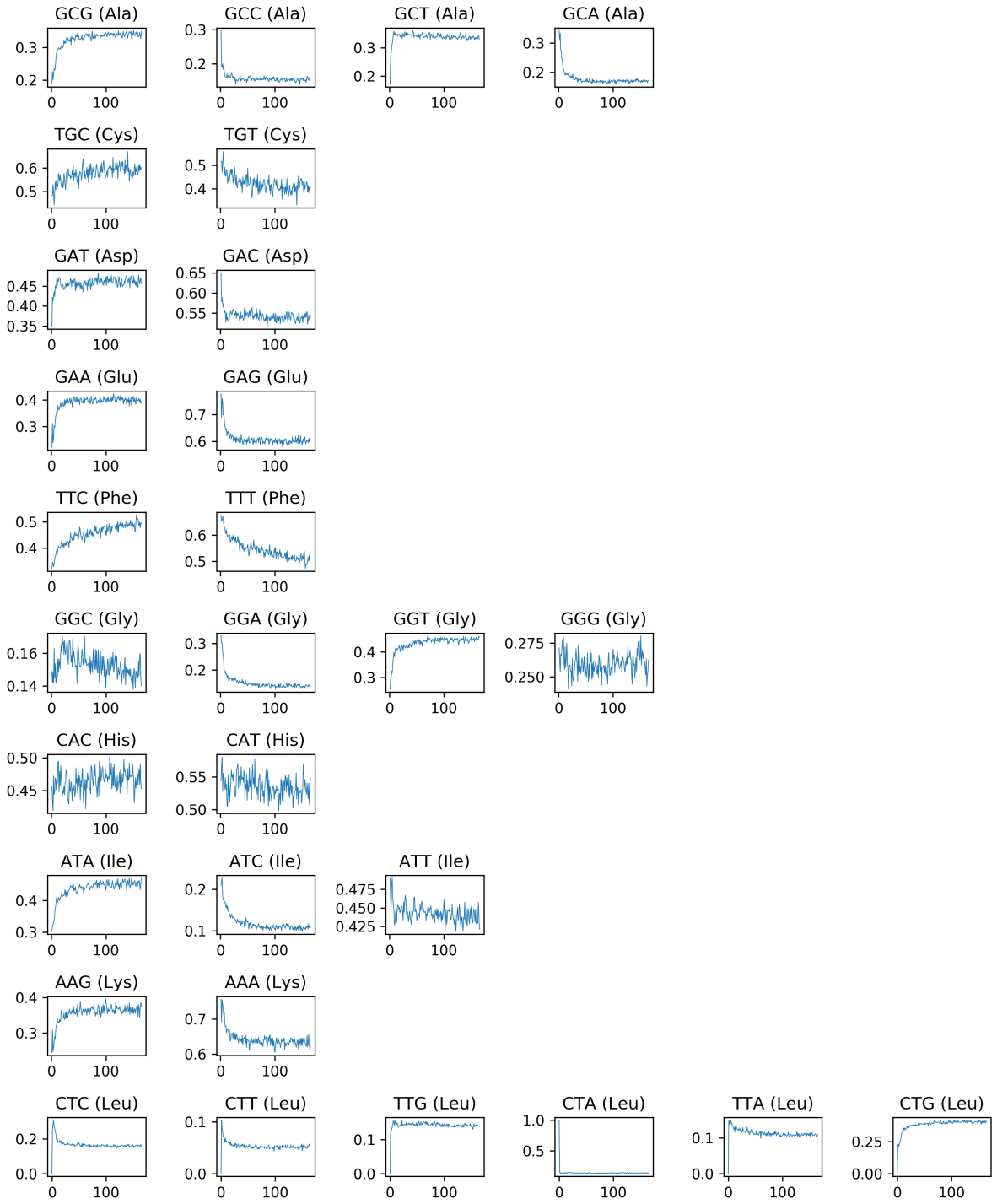


Figure 18: Relative codon frequencies after and including the start codon (position 0) of all genes from 102,183 bacterial genomes. The x-axes correspond to the codon number after the start codon position whereas the y-axes show the relative codon frequency (that is, the curves of all plots in a row add up to a line with value 1 everywhere). The figure continues on the following page.

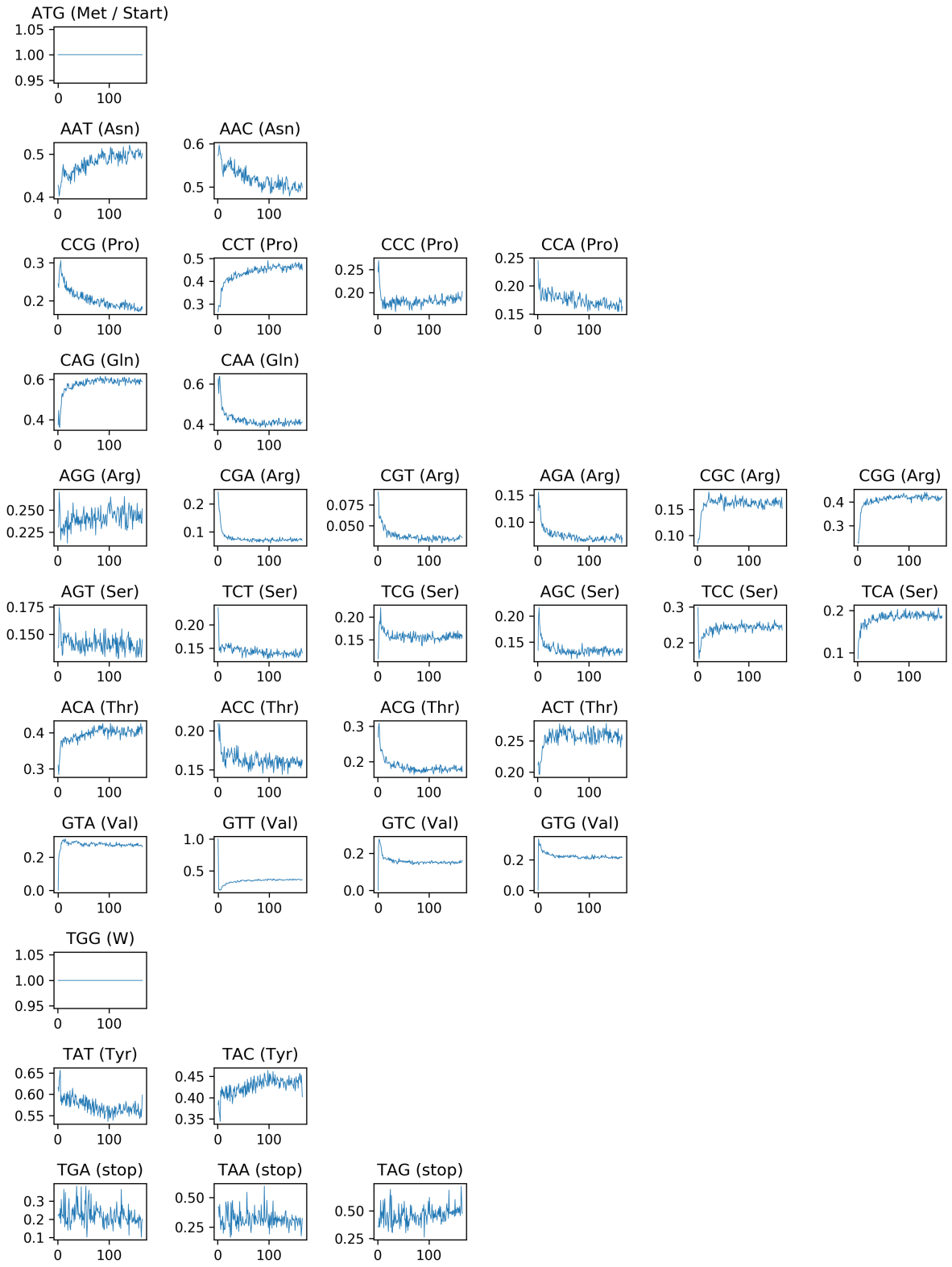


Figure continued from previous page.

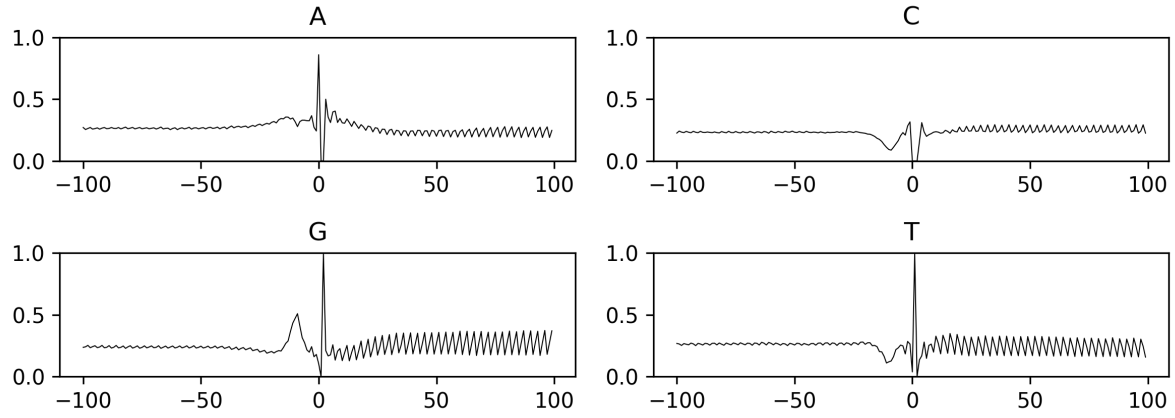


Figure 19: Relative nucleotide frequencies around the start codon (position 0) of all genes from 102,183 bacterial genomes. The x-axes correspond to the nucleotide position whereas the y-axes show the relative nucleotide frequency.

Methods

All available bacterial genomes were downloaded from NCBI, and sequences corresponding to a region 18 nucleotides before (untranslated region, UTR) and 33 nucleotides from the start codon (coding sequence, CDS) were extracted, resulting in a total of 280,071,706 sequences from 102,183 genomes. Training and test data sets were generated based on whether genes coded for ribosomal or other proteins and whether they coded for the small (S) or large (L) ribosomal subunit. Only sequences of species with ribosomal RNA gene copy numbers greater than 10 were used. The coding sequences were translated to canonical amino acid sequences (1-of-20 coding) and also separately converted to codon sequences (1-of-64 coding).

A linear model was trained to map amino acid sequences to the respective codons. The UTRs and ribosomal gene codon frequencies of the respective species were supplied as additional input features. Networks were trained at a learning rate 0.001 and batch size 128 for 10,000 mini-batches.

To determine whether ribosomal genes had higher folding energies than other genes, separate sets of 10,000 ribosomal and non-ribosomal genes were randomly selected from all genomes, and folding energies were estimated at $30°C$ using RNAfold [92]. This was done separately for species with high (> 10) and low (≤ 10) rRNA copy numbers.

Results

When training and test data sets contained only ribosomal genes, high accuracies were achieved (table 1). The same was true when training and test data sets contained only genes corresponding to the same ribosomal subunits. However, when the data sets contained sequences from different ribosomal subunits, the test accuracy dropped substantially, indicating overfitting. Using non-ribosomal instead of ribosomal sequences for training resulted in lower but comparable accuracies on both training and test data sets, however accuracies were lower when ribosomal sequences were used for the test data.

The distributions of estimated folding energies of non-ribosomal genes differed between species with low or high rRNA copy numbers in their skewness: bacteria with low rRNA copy number were slightly more likely to have mRNAs with low folding energy (figure 20). Presumably due to a high degree of similarity between ribosomal genes, the histograms for the ribosomal genes resulted in partially highly similar folding energies; this made it necessary to use larger bin sizes than expected by the number of sequences.

Table 1: Accuracies with which the the first 11 codons of mRNAs could be predicted. Predictions were made based on the respective translated sequences and the 18 nucleotides of the preceding untranslated region. Letters S and L indicate small and large ribosomal subunits, respectively.

training data	test data	training accuracy	test accuracy
ribosomal, S+L	ribosomal, S+L	0.98	0.96
ribosomal, S	ribosomal, S	0.98	0.98
ribosomal, L	ribosomal, S	0.98	**0.33**
non-ribosomal	ribosomal, S	0.65	0.56
non-ribosomal	non-ribosomal	0.66	0.65

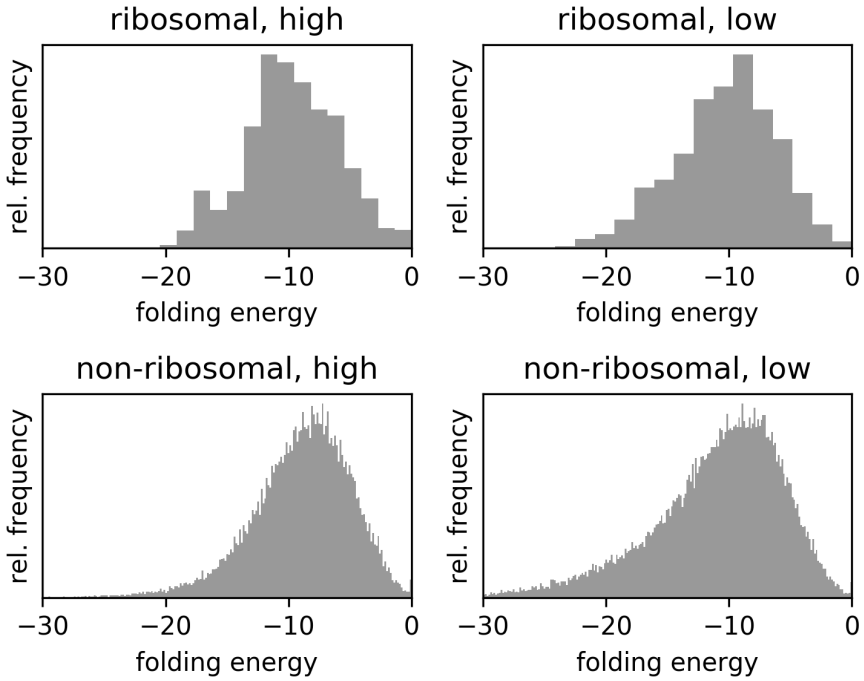


Figure 20: Histograms of predicted mRNA folding energies for bacteria with high (> 10) and low (≤ 10) rRNA copy number, for both ribosomal and non-ribosomal genes.

Discussion

The results indicate that there is insufficient sequence variation among the ribosomal genes to avoid overfitting. This suggests that, despite the availability of more than 100,000 sequenced bacterial genomes, supervised learning based on ribosomal sequence features is not sufficient to disentangle unique ribosomal features from those that are generally more relevant for high translational efficiency.

A likely reason for this is insufficient sequence variability among the ribosomal genes, resulting in a lower effective amount of data. That is, the data is not uniformly sampled from feature space

but rather corresponds to a few clusters with high degree of similarity. Ribosomal genes are highly conserved [116]. Given that ribosomal genes are fundamental components of the cellular machinery and that they are highly expressed, they are maintained by strong selective pressure. Increasing the effective amount of data could be accomplished by using sequences that are homologs of known highly expressed genes. Although there is evidence for a correlation between gene expression and conservation [118], it remains to be elucidated whether transfer of this prior knowledge can improve predictions.

An alternative to a supervised approach might be to use unsupervised methods such as autoencoders to compress sequence information into low-dimensional latent features, which could then be correlated with ribosomal genes. The advantage would be that no hand-crafted classes were required, and other highly translationally efficient genes would be automatically included. However, it remains to be shown whether translational efficiency could be distilled as a main feature relevant for effective compression of sequence information, or if latent features would segregate according to, for example, gene families.

4 Assessment of a biologically inspired neural network for one-shot RNA secondary structure prediction

Summary of the obtained results

RNA folding can be considered as a combinatorial problem which is predominantly addressed by dynamic programming. Machine learning techniques that learn the physical principles underlying RNA folding from data are a promising approach for improving predictive performance. Since RNA folding is the result of intramolecular hybridization of complementary sequence regions, learning RNA secondary structure requires the ability to identify spatially correlated data features.

As a first step towards a model that is capable of this task, a special neural network architecture was designed that aimed at enumerating intramolecular hybridization states, and then choosing the most likely hybridization state. This architecture displayed improved predictive accuracy compared to a naive fully connected network mapping sequence to secondary structure. However, the investigated network achieved only 80 % prediction accuracy compared to the current state-of-the-art method for RNA secondary structure prediction, and networks based on attention mechanisms were proposed as an alternative as they likely scale better with sequence length and were shown to be capable of solving combinatorial problems.

Motivation

As various aspects of the mechanisms governing RNA folding have been revealed, it is reasonable to assume that utilization of this knowledge can support supervised learning of RNA folding. In order to model the most basic mechanisms of RNA folding, a neural network should be capable of finding intramolecular regions that hybridize, and it should learn the influence of directly adjacent nucleotides on hybridization. Although large data sets related to RNA folding are yet unavailable, synthetic data can be used for early investigation of a model for RNA folding.

Methods

Two neural network structures were trained to map nucleic acid sequences to their corresponding dot-bracket structures. Data was created by generating random nucleic acid sequences with a length of 33 nucleotides and predicting the respective dot-bracket structures with RNAfold[92]. Separate training and test data sets were created, with training and test sets containing 1,000,000 and 100,000 examples, respectively. Although it is possible that some sequences occurred in both training and test data sets, the probability for this was low ($\frac{1}{4^{33}} \cdot 10^5 \cdot 10^6 \approx 10^{-8}$), enabling proper monitoring of overfitting.

The two networks were a 2-layer neural network with fully connected layers and a specially designed neural network. Both networks made predictions in one forward-pass (one-shot). The networks used one-hot coded nucleotide sequences and dot-bracket structures as inputs and targets, respectively. The goal was to predict the mutually exclusive symbols at each position in the dot-bracket structure, for which the multiclass cross-entropy[31] between predictions and targets was minimized. The 2-layer neural network mapped the input to the output via two consecutive hidden layers of equal size, and the softplus function was used as an activation function. The specially designed network resembled the 2-layer neural network, but additionally had a special layer architecture between the input and the first hidden layer (figure 21). This special architecture enabled enumerating hybridization of the sequence with itself. It also allowed learning weights for dinucleotide binding events to account for effects of directly adjacent nucleotides. The 2 fully connected layers were chosen with the intention to enable approximate computation of mutually exclusive hybridization states.

The network architectures were investigated for different sizes of the hidden layers but comparable number of parameters. Prediction accuracies were calculated as the number of correctly predicted dot-bracket symbols per structure. As the dot-bracket representation is mutually exclusive at each position, random guessing of the dot-bracket symbols would result in an accuracy of $\frac{1}{3} \approx 0.33$. To obtain a better reference for prediction accuracies, an average dot bracket structure was calculated by calculating the position-dependent frequencies of dots and brackets and then choosing the most likely symbols. This average structure was then compared to all dot bracket structures in the test

data set to obtain a more informative baseline prediction accuracy.

As RNA sequences that do not fold strongly tend to occupy more folding states at comparable frequencies, corresponding structures might be difficult to predict. To test this hypothesis, the accuracies of 10,000 predicted structures were compared to the respective folding energies. The hypothesis of a significant correlation between folding energy and prediction accuracy was tested via bootstrapping with 10,000 bootstrap samples.

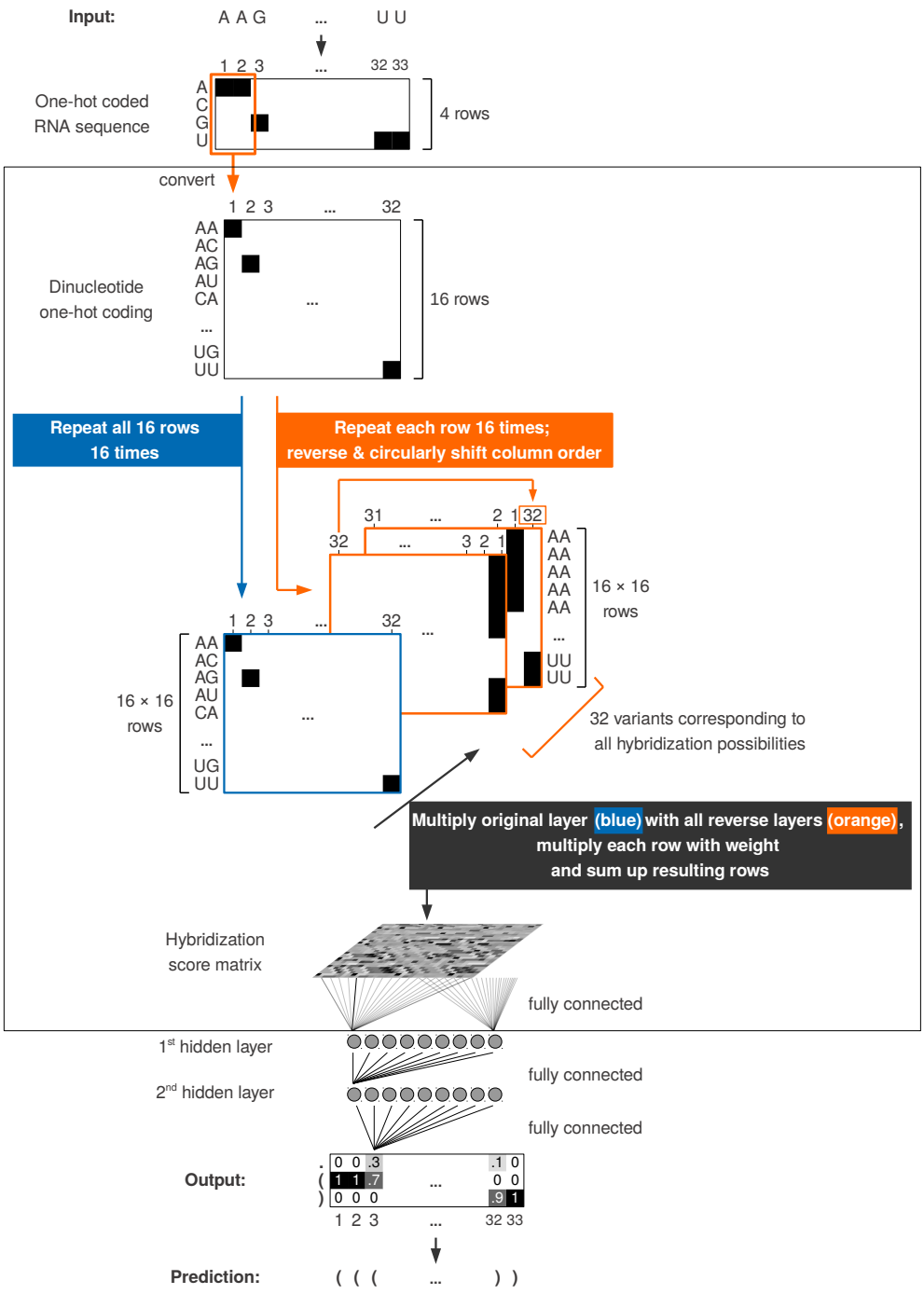


Figure 21: Illustration of a neural network for RNA secondary structure prediction. The network takes RNA sequences of fixed length (33 nucleotides here) as input and predicts corresponding dot-bracket structures. It works as follows. First, sequences are converted to one-hot coding, which is then converted to a dinucleotide one-hot coding that is one element shorter than the original sequence (there are 32 overlapping dinucleotides). Then, the resulting dinucleotide sequence is compared to its reverse sequence such that all possible hybridizations and dinucleotide combinations are enumerated. All enumerated hybridization states are converted into a score matrix, which then serves as input to the following fully connected network with 2 hidden layers. Finally, the hidden nodes are mapped to yield dot-bracket symbol probabilities, which are then turned into the final predictions.

Results

The accuracies on training and test data sets for both the 2-layer neural network and the special network architecture are shown in table 2 and table 3, respectively. When an average dot bracket structure (figure 22) was used to predict all dot bracket structures in the test set, the resulting prediction accuracy was 0.58. This average dot bracket structure entirely consisted of dots. All accuracies obtained with either the fully connected or specially designed network were higher than this value. Despite a comparable number of parameters, the specially designed architecture performed substantially better on both data sets than the 2-layer neural network. Example outputs from the best performing model are given in table 4, with accuracies next to the predictions. It can be observed that general patterns such as hairpins and adjacent hairpins were predicted. However, the number of opening brackets frequently did not match the number of closing brackets and vice versa. No correlation was found between structure prediction accuracies and the respective folding energies (figure 23).

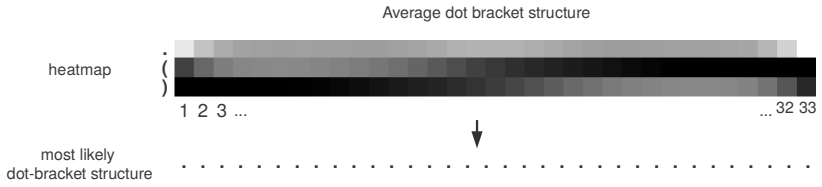


Figure 22: Heatmap of the average position-dependent symbol frequencies and most likely symbols of dot-bracket structures derived from random sequences of 33 nucleotides length. Bright regions in the heatmap correspond to high symbol frequencies. The dot-bracket structure consisting of the most likely symbols has, on average, 58 % identity with the dot-bracket structures of random sequences.

Table 2: Performance of a neural network network that enumerates hybridization states.

sizes of hidden layer 1 and 2	number of parameters	training accuracy	test accuracy
10, 10	11705	0.69	0.69
300, 300	265355	0.78	0.77
500, 500	812855	0.83	0.80
1000, 1000	2125355	0.84	0.81

Table 3: Performance of a 2-layer neural network with fully connected layers.

sizes of hidden layer 1 and 2	number of parameters	training accuracy	test accuracy
40, 40	11019	0.67	0.66
400, 400	273299	0.69	0.67
800, 800	826499	0.69	0.67

Table 4: Samples from the test data set with predictions from the best performing network and corresponding prediction accuracies (percent identity between original and predicted dot-bracket structures)

example		sequence	accuracy
1	RNA sequence:	CAUUAGAGACCGUGGGUUAUGCAGAGUCCGAUA	
	original:	.....(.(((..((.......))..))))....	
	predicted:	......(((((.((......)))..))))....	87 %
2	RNA sequence:	CCAGCAACCGCCACAUCAAUGGUCUGAGCGUCC	
	original:	...((....((((......))))....))....	
	predicted:	...((....((((......))))....))....	100 %
3	RNA sequence:	CGUUUAGGCCAAUAUAUUUAGACAACGAUUAUU	
	original:	.(((((((........))))))).........	
	predicted:	.(((((...........))))).........	90 %
4	RNA sequence:	CUGCGCGGCUGGGACCUCCAUUAGUUGAGAUCA	
	original:	...(.(((((((........)))))))).)....	
	predicted:	.....(((((((....))...))))).)....	81 %
5	RNA sequence:	AGAGCGGUAGAAGAUAGCAGUUGUUACCAUGCU	
	original:	..(((((((..(((....))).)))))..)))	
	predicted:	...(((((((..((.....)).)))))..)))	87 %
6	RNA sequence:	CCAGAGUAGUUCCACCCUUUAUUUAGUGUGCCG	
	original:	...........(((.((......)).)))...	
	predicted:	...........(((.........)).))...	84 %
7	RNA sequence:	GGUAGACAAGUUGUUACAAUCAGUCGAGAGAUG	
	original:	....(((..((((...)))).))).).....	
	predicted:	.((((((...(((...)))).))).).....	87 %
8	RNA sequence:	CGUCUUCAAAUCAUGGACUUAGUUCUGAAAUUA	
	original:	.........(((.((((...))))))).....	
	predicted:	............((((....)))).......	81 %
9	RNA sequence:	CUAAUUGGCCUUUGACGCAACUUGUGAUGCAAA	
	original:	..............(((.(....).)))...	
	predicted:	..........(...((((....)))..)))).	75 %
10	RNA sequence:	CCGCUUAAUGCGCGGAGGGGCUUCCGUAUCCUG	
	original:	((((.......))))(((((....)).)))).	
	predicted:	((((......))))).(((......)...)...	75 %

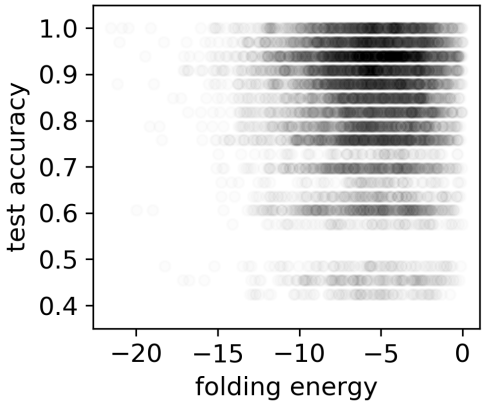


Figure 23: Plot of folding energies corresponding to MFE structures predicted by RNAfold and secondary structure prediction accuracies of the best performing model for random RNA sequences. No significant correlation ρ between both quantities could be observed ($\rho = -0.006$, $p = 0.26$ for $H_0 : \rho > 0$). The accuracy values are discrete, which is a result of the limited length (33 nucleotides) of the investigated sequences. Dots were plotted with high transparency to allow visualization of the data distribution (darker areas correspond to multiple overlapping dots).

Discussion

These results indicate that a specially designed network architecture can find the relevant sequence information for predicting RNA folding more easily than a naive neural network with comparable number of parameters. This suggests that hard-coded prior knowledge about the underlying problem was the cause for the improved performance. However, since the highest achieved accuracy was well well below 100 % and accuracies did not increase with increasing number of parameters, it remains unclear if all possible information was utilized to achieve the maximum possible accuracy or if the network structure was the limiting factor. Answering this question would require investigation of different neural networks for RNA secondary structure prediction. Both networks contained two hidden layers whose size depended on the sequence length, implying that this architecture would not scale well with longer sequences. Furthermore, the fact that the number of opening brackets often did not match the number of closing brackets indicates insufficient ability to solve the underlying combinatorial problem. Overall, this suggests to use approaches from neural machine translation that can translate between sequences (in this case, nucleotide to dot-bracket sequences). Attention-based recurrent network architectures seem to be especially promising candidates as they enable solving combinatorial tasks on sequences with variable lengths[49]. The fact that folding energies did not correlate with prediction accuracies indicates that, in order to generate a non-synthetic training data set with real RNA structures, sequences do not need to come from a particular folding energy distribution to accurately learn RNA folding.

Conclusion

The relationship between data structures and the ability of data-efficient machine learning was investigated based on four fundamental biological problems: gene regulatory networks, biological motifs, translational efficiency and RNA folding.

Utilization of prior knowledge about the data or the underlying processes improved learning in all cases. Inference of gene regulatory networks improved substantially by incorporating a biologically informed noise cutoff, a node clustering method, and a noise-unbiased regression method. Biological sequence motifs can now be learned efficiently with convolutional neural networks from few training examples due to an improved network optimization method that helps to escape local optima. Prediction of translational efficiency was improved by incorporating knowledge about inter- and intramolecular nucleic acid hybridization, and evidence was presented that selective pressure on highly expressed genes prevents learning translational efficiency from sequences alone. Finally, it was found that providing a neural network with the ability to select from intramolecular hybridization states seems to be crucial for predicting RNA secondary structure more accurately.

Overall, it was demonstrated that models tailored to the underlying problem are crucial for data-efficient machine learning. The results also highlight the importance of prior and transferable knowledge for model selection or design, suggesting that vast quantities of data is not the universal answer to addressing biological problems with machine learning and, vice versa, out-of-the-box machine learning is not the universal answer in the post-genomic era.

Acknowledgements

Markus. It has been an awesome journey through the field of machine learning with you. There has not been a single day on which I regretted choosing you as my supervisor. If I ever become a supervisor or boss myself, I am going to apply the skills that I learned from you. Thank you.

Sarah. Thank you for cheering me up whenever my code did not work, for telling me to stop working and to go to sleep and for the inspiring discussions that helped me to stay creative.

Mathias. Thank you for all the helpful discussions and insights into science.

Linlin. Thank you for making me laugh on so many occasions. Oh, and of course thank you for the discussions.

Laura. Thank you for being Sarah's supervisor and enabling her to get her PhD.

Lukas and Jenja. Thank you for your criticism, homes.

Dr. Walther Enßlin and Bernhard Osterwind. Thank you for getting me into science.

Mom & Patricia. Thank you for food, housing, and letting me excite you about biology.

Thank you Nima, Nima, Nadia, Armin, Sabrina, Rahil, Atefeh, Lisa, Fiona, Petra and Jürgen for being awesome colleagues.

Bibliography

1. Schrödinger, E. *What is Life? The Physical Aspect of the Living Cell* (Cambridge University Press, 1944).
2. Boltzmann, L. *The Second Law of Thermodynamics* (ed McGuinness, B.) (Reidel Publishing Company, 1974).
3. Darwin, C. *On The Origin of Species by Means of Natural Selection, or Preservation of Favoured Races in the Struggle for Life* (John Murray, London, 1859).
4. Tishkoff, S. A. *et al.* Convergent adaptation of human lactase persistence in Africa and Europe. *Nat. Genet.* **39,** 31–40 (2007).
5. Gerbault, P. *et al.* Evolution of lactase persistence: an example of human niche construction. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.* **366,** 863–877 (2011).
6. Hall, M. N., Gabay, J., Débarbouillé, M. & Schwartz, M. A role for mRNA secondary structure in the control of translation initiation. *Nature* **295,** 616 (1982).
7. Chen, C. *et al.* Dynamics of translation by single ribosomes through mRNA secondary structures. *Nat. Struct. Mol. Biol.* **20,** 582–588 (2013).
8. Wen, J. D. *et al.* Following translation by single ribosomes one codon at a time. *Nature* **452,** 598–603 (2008).
9. Elena, S. F. & Lenski, R. E. Evolution experiments with microorganisms: the dynamics and genetic bases of adaptation. *Nature Reviews Genetics* **4.** Review Article, 457 (2003).
10. Cressler, C. E., McLEOD, D. V., Rozins, C., VAN DEN Hoogen, J. & Day, T. The adaptive evolution of virulence: a review of theoretical predictions and empirical tests. *Parasitology* **143,** 915–930 (June 2016).
11. Bleuven, C. & Landry, C. R. Molecular and cellular bases of adaptation to a changing environment in microorganisms. *Proceedings of the Royal Society of London B: Biological Sciences* **283** (2016).
12. Jacob, F. & Monod, J. Genetic regulatory mechanisms in the synthesis of proteins. *J. Mol. Biol.* **3,** 318–356 (1961).
13. Dekel, E. & Alon, U. Optimality and evolutionary tuning of the expression level of a protein. *Nature* **436,** 588–592 (2005).
14. Bennett, M. R. *et al.* Metabolic gene regulation in a dynamically changing environment. *Nature* **454,** 1119 (2008).
15. Shen-Orr, S. S., Milo, R., Mangan, S. & Alon, U. Network motifs in the transcriptional regulation network of Escherichia coli. *Nature Genetics* **31,** 64 (2002).
16. Sadava, D. E., Hillis, D. M., Heller, H. C. & Hacker, S. D. *Life: The science of biology* 11th ed. (Sinauer Associates, Sunderland, MA, 2017).

17. Alberts, B. *et al. Molecular Biology of the Cell* 4th (Garland, 2002).

18. Gluck, M. A., Mercado, E. & Myers, C. E. *Learning and Memory: From Brain to Behavior* 2nd (Worth Publishers, 2013).

19. Samuel, A. L. Some Studies in Machine Learning Using the Game of Checkers. *IBM J. Res. Dev.* **3,** 210–229 (July 1959).

20. Kuczaj, S. A. The acquisition of regular and irregular past tense forms. *Journal of Verbal Learning and Verbal Behavior* **16,** 589 –600 (1977).

21. Xie, K. *et al.* Brain Computation Is Organized via Power-of-Two-Based Permutation Logic. *Frontiers in Systems Neuroscience* **10,** 95 (2016).

22. Thoreson, W. B. & Mangel, S. C. Lateral interactions in the outer retina. *Prog Retin Eye Res* **31,** 407–441 (2012).

23. Lecun, Y. in *Connectionism in perspective* (eds Pfeifer, R., Schreter, Z., Fogelman, F. & Steels, L.) (Elsevier, 1989).

24. He, K., Zhang, X., Ren, S. & Sun, J. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification* in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* (IEEE Computer Society, Washington, DC, USA, 2015), 1026–1034.

25. Kaiser, L. *et al.* One Model To Learn Them All. *CoRR* **abs/1706.05137** (2017).

26. Graves, A. *et al.* Hybrid computing using a neural network with dynamic external memory. *Nature* **538,** 471–476 (Oct. 2016).

27. Pratt, L. Y. in *Advances in Neural Information Processing Systems 5* (eds Hanson, S. J., Cowan, J. D. & Giles, C. L.) 204–211 (Morgan-Kaufmann, 1993).

28. Schmidhuber, J. Goedel Machines: Self-Referential Universal Problem Solvers Making Provably Optimal Self-Improvements. *CoRR* **cs.LO/0309048** (2003).

29. Cordovado, S. K. *et al.* CFTR mutation analysis and haplotype associations in CF patients. *Mol. Genet. Metab.* **105,** 249–254 (2012).

30. Xiong, H. Y. *et al.* The human splicing code reveals new insights into the genetic determinants of disease. *Science* **347** (2015).

31. Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006).

32. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. in (eds Rumelhart, D. E., McClelland, J. L. & PDP Research Group, C.) 318–362 (MIT Press, Cambridge, MA, USA, 1986).

33. Baldi, P. & Hornik, K. Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima. *Neural Netw.* **2,** 53–58 (Jan. 1989).

34. Kingma, D. P. & Welling, M. Auto-Encoding Variational Bayes. *CoRR* **abs/1312.6114** (2013).

35. Alipanahi, B., Delong, A., Weirauch, M. T. & Frey, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* **33,** 831–838 (2015).

36. Hinton, G. E., Osindero, S. & Teh, Y. W. A fast learning algorithm for deep belief nets. *Neural Comput* **18,** 1527–1554 (2006).

37. Simonyan, K. & Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* **abs/1409.1556** (2014).

38. He, K., Zhang, X., Ren, S. & Sun, J. *Deep Residual Learning for Image Recognition* in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016* (2016), 770–778.

39. Ruder, S. An overview of gradient descent optimization algorithms. *CoRR* **abs/1609.04747** (2016).

40. Abadi, M. *et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems* Software available from tensorflow.org. 2015.

41. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput* **9,** 1735–1780 (1997).

42. Ioffe, S. & Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR* **abs/1502.03167** (2015).

43. Tibshirani, R. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B* **58,** 267–288 (1994).

44. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* **15,** 1929–1958 (2014).

45. Welling, M. & Teh, Y. W. *Bayesian Learning via Stochastic Gradient Langevin Dynamics.* in *ICML* (eds Getoor, L. & Scheffer, T.) (Omnipress, 2011), 681–688.

46. Bengio, Y., Courville, A. C. & Vincent, P. Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives. *CoRR* **abs/1206.5538** (2012).

47. Alon, U. *An Introduction to Systems Biology: Design Principles of Biological Circuits* 1st ed. (Chapman and Hall/CRC, 2006).

48. Lipton, Z. C., Berkowitz, J. & Elkan, C. A Critical Review of Recurrent Neural Networks for Sequence Learning. *CoRR* **abs/1506.00019** (2015).

49. Bahdanau, D., Cho, K. & Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* **abs/1409.0473** (2014).

50. Vinyals, O., Bengio, S. & Kudlur, M. *Order matters: Sequence to sequence for sets* in *International Conference on Learning Representations (ICLR)* (2016).

51. George, J. *et al.* Comprehensive genomic profiles of small cell lung cancer. *Nature* **524,** 47–53 (2015).

52. Vogelstein, B. *et al.* Cancer genome landscapes. *Science* **339,** 1546–1558 (2013).

53. Scott, A. M., Allison, J. P. & Wolchok, J. D. Monoclonal antibodies in cancer therapy. *Cancer Immun.* **12,** 14 (2012).

54. Giacca, M. & Zacchigna, S. VEGF gene therapy: therapeutic angiogenesis in the clinic and beyond. *Gene Ther.* **19,** 622–629 (2012).

55. Marbach, D. *et al.* Wisdom of crowds for robust gene network inference. *Nat. Methods* **9,** 796–804 (2012).

56. Kemmeren, P. *et al.* Large-scale genetic perturbations reveal regulatory networks and an abundance of gene-specific repressors. *Cell* **157,** 740–752 (2014).

57. Park, Y. N., Masison, D., Eisenberg, E. & Greene, L. E. Application of the FLP/FRT system for conditional gene deletion in yeast Saccharomyces cerevisiae. *Yeast* **28,** 673–681 (2011).

58. Fraser, C. M. *et al.* The minimal gene complement of Mycoplasma genitalium. *Science* **270,** 397–403 (1995).

59. Wegrzyn, J. L. *et al.* Unique features of the loblolly pine (Pinus taeda L.) megagenome revealed through sequence annotation. *Genetics* **196,** 891–909 (2014).

60. *Kyoto Encyclopedia of Genes and Genomes, information about the human genome* http://www.genome.jp/kegg-bin/show_organism?org=hsa. Accessed: 2018-04-18.

61. Tu, Y., Stolovitzky, G. & Klein, U. Quantitative noise analysis for gene expression microarray experiments. *Proceedings of the National Academy of Sciences* **99,** 14031–14036 (2002).

62. Shi, L. *et al.* The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. *Nat. Biotechnol.* **24,** 1151–1161 (2006).

63. Li, J. J. & Biggin, M. D. Gene expression. Statistics requantitates the central dogma. *Science* **347,** 1066–1067 (2015).

64. Elowitz, M. B., Levine, A. J., Siggia, E. D. & Swain, P. S. Stochastic Gene Expression in a Single Cell. *Science* **297,** 1183–1186 (2002).

65. Kærn, M., Elston, T. C., Blake, W. J. & Collins, J. J. Stochasticity in gene expression: from theories to phenotypes. *Nature Reviews Genetics* **6.** Review Article, 451 (2005).

66. Li, J. *et al.* Exploiting the determinants of stochastic gene expression in Saccharomyces cerevisiae for genome-wide prediction of expression noise. *Proceedings of the National Academy of Sciences* **107,** 10472–10477 (2010).

67. Leek, J. T. *et al.* Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews Genetics* **11.** Perspective, 733 (2010).

68. MacQueen, J. *Some methods for classification and analysis of multivariate observations* in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (University of California Press, Berkeley, Calif., 1967), 281–297.

69. Jambhekar, A. & Derisi, J. L. Cis-acting determinants of asymmetric, cytoplasmic RNA transport. *RNA* **13,** 625–642 (2007).

70. Edelmann, F. T. *et al.* Molecular architecture and dynamics of ASH1 mRNA recognition by its mRNA-transport complex. *Nat. Struct. Mol. Biol.* **24,** 152–161 (Feb. 2017).

71. Illergard, K., Ardell, D. H. & Elofsson, A. Structure is three to ten times more conserved than sequence–a study of structural response in protein cores. *Proteins* **77,** 499–508 (2009).

72. Fischer, M. Census and evaluation of p53 target genes. *Oncogene* **36,** 3943–3956 (2017).

73. Vollmeister, E. *et al.* Tandem KH domains of Khd4 recognize AUACCC and are essential for regulation of morphology as well as pathogenicity in Ustilago maydis. *RNA* **15,** 2206–2218 (2009).

74. Weirauch, M. T. *et al.* Evaluation of methods for modeling transcription factor sequence specificity. *Nat. Biotechnol.* **31,** 126–134 (2013).

75. Stormo, G. D., Schneider, T. D., Gold, L. & Ehrenfeucht, A. Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Res.* **10,** 2997–3011 (1982).

76. Dunham, I. *et al.* An integrated encyclopedia of DNA elements in the human genome. *Nature* **489,** 57–74 (2012).

77. Zeng, H., Edwards, M. D., Liu, G. & Gifford, D. K. Convolutional neural network architectures for predicting DNA-protein binding. *Bioinformatics* **32,** i121–i127 (June 2016).

78. Doudna, J. A. & Doherty, E. A. Emerging themes in RNA folding. *Fold Des* **2,** 65–70 (1997).

79. Yanofsky, C. Attenuation in the control of expression of bacterial operons. *Nature* **289,** 751–758 (1981).

80. Winkler, W. C., Nahvi, A., Roth, A., Collins, J. A. & Breaker, R. R. Control of gene expression by a natural metabolite-responsive ribozyme. *Nature* **428,** 281–286 (2004).

81. Altuvia, S., Kornitzer, D., Teff, D. & Oppenheim, A. B. Alternative mRNA structures of the cIII gene of bacteriophage lambda determine the rate of its translation initiation. *J. Mol. Biol.* **210,** 265–280 (1989).

82. Kruger, K. *et al.* Self-splicing RNA: autoexcision and autocyclization of the ribosomal RNA intervening sequence of Tetrahymena. *Cell* **31,** 147–157 (1982).

83. Nissen, P., Hansen, J., Ban, N., Moore, P. B. & Steitz, T. A. The structural basis of ribosome activity in peptide bond synthesis. *Science* **289,** 920–930 (2000).

84. Simmonds, P., Tuplin, A. & Evans, D. J. Detection of genome-scale ordered RNA structure (GORS) in genomes of positive-stranded RNA viruses: Implications for virus evolution and host persistence. *RNA* **10,** 1337–1351 (2004).

85. Mathews, D. H., Sabina, J., Zuker, M. & Turner, D. H. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.* **288,** 911–940 (1999).

86. Hofacker, I. L. *et al.* Fast Folding and Comparison of RNA Secondary Structures. *Chemical Monthly* **125,** 167–188 (1994).

87. Zuker, M. Calculating nucleic acid secondary structure. *Curr. Opin. Struct. Biol.* **10,** 303–310 (2000).

88. Boltzmann, L. Über die Beziehung zwischen dem zweiten Hauptsatze des mechanischen Wärmetheorie und der Wahrscheinlichkeitsrechnung, resp. den Sätzen Ãijber das Wärmegleichgewicht. *Sitzungberichte der Kaiserlichen Akademie der Wissenschaften, Mathematich-Naturwissen Classe, Abt II* **76** (1877).

89. Ding, Y. & Lawrence, C. E. A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Res.* **31,** 7280–7301 (2003).

90. Nussinov, R., Pieczenik, G., Griggs, J. R. & Kleitman, D. J. Algorithms for Loop Matchings. *SIAM Journal on Applied Mathematics* **35,** 68–82 (1978).

91. Zuker, M. & Stiegler, P. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res.* **9,** 133–148 (1981).

92. Lorenz, R. *et al.* ViennaRNA Package 2.0. *Algorithms Mol Biol* **6,** 26 (2011).

93. Farajian, M. A., Turchi, M., Negri, M., Bertoldi, N. & Federico, M. *Neural vs. Phrase-Based Machine Translation in a Multi-Domain Scenario* in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (Association for Computational Linguistics, Valencia, Spain, 2017), 280–284.

94. Vinyals, O., Fortunato, M. & Jaitly, N. in *Advances in Neural Information Processing Systems 28* (eds Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M. & Garnett, R.) 2692–2700 (Curran Associates, Inc., 2015).

95. Bellman, R. Dynamic Programming Treatment of the Travelling Salesman Problem. *J. ACM* **9,** 61–63 (Jan. 1962).

96. Ikemura, T. Codon usage and tRNA content in unicellular and multicellular organisms. *Mol. Biol. Evol.* **2,** 13–34 (1985).

97. Plotkin, J. B. & Kudla, G. Synonymous but not the same: the causes and consequences of codon bias. *Nat. Rev. Genet.* **12,** 32–42 (2011).

98. Goodman, D. B., Church, G. M. & Kosuri, S. Causes and effects of N-terminal codon bias in bacterial genes. *Science* **342,** 475–479 (2013).

99. Sharp, P. M. & Li, W. H. The codon Adaptation Index–a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res.* **15,** 1281–1295 (1987).

100. Dos Reis, M., Wernisch, L. & Savva, R. Unexpected correlations between gene expression and codon usage bias from microarray data for the whole Escherichia coli K-12 genome. *Nucleic Acids Res.* **31,** 6976–6985 (2003).

101. Tuller, T., Waldman, Y. Y., Kupiec, M. & Ruppin, E. Translation efficiency is determined by both codon bias and folding energy. *Proceedings of the National Academy of Sciences* **107,** 3645–3650 (2010).

102. Boel, G. *et al.* Codon influence on protein expression in E. coli correlates with mRNA levels. *Nature* **529,** 358–363 (2016).

103. Salis, H. M., Mirsky, E. A. & Voigt, C. A. Automated design of synthetic ribosome binding sites to control protein expression. *Nat. Biotechnol.* **27,** 946–950 (2009).

104. Seo, S. W. *et al.* Predictive design of mRNA translation initiation region to control prokaryotic translation efficiency. *Metab. Eng.* **15,** 67–74 (2013).

105. Na, D. & Lee, D. RBSDesigner: software for designing synthetic ribosome binding sites that yields a desired level of protein expression. *Bioinformatics* **26,** 2633–2634 (2010).

106. Rodrigo, G. & Jaramillo, A. RiboMaker: computational design of conformation-based riboregulation. *Bioinformatics* **30,** 2508–2510 (2014).

107. Fuglsang, A. Codon optimizer: a freeware tool for codon optimization. *Protein Expr. Purif.* **31,** 247–249 (2003).

108. Puigbo, P., Guzman, E., Romeu, A. & Garcia-Vallve, S. OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. *Nucleic Acids Res.* **35,** W126–131 (2007).

109. Trefethen, L. N. & Bau, D. *Numerical Linear Algebra* (SIAM, 1997).

110. Wishart, J. The generalized product moment distribution in samples from a normal multivariate population. *Biometrika* **20A,** 32–52 (1928).

111. Krishnaiah, P. & Chang, T. On the exact distributions of the extreme roots of the Wishart and MANOVA matrices. *Journal of Multivariate Analysis* **1,** 108 –117 (1971).

112. Abramowitz, M. & Stegune, I. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables* 9th printing (Dover Publications, New York, 1972).

113. Fisher, R. A. The sampling distribution of some statistics obtained from non-linear equations. *Annals of Eugenics* **9,** 238–249 (Aug. 1939).

114. Gutmann, M. U. & Hyvärinen, A. Noise-contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *J. Mach. Learn. Res.* **13,** 307–361 (Feb. 2012).

115. Altschul, S. F. & Erickson, B. W. Significance of nucleotide sequence alignments: a method for random sequence permutation that preserves dinucleotide and codon usage. *Mol. Biol. Evol.* **2,** 526–538 (1985).

116. Lecompte, O., Ripp, R., Thierry, J. C., Moras, D. & Poch, O. Comparative analysis of ribosomal proteins in complete genomes: an example of reductive evolution at the domain scale. *Nucleic Acids Res.* **30,** 5382–5390 (2002).

117. Roller, B. R., Stoddard, S. F. & Schmidt, T. M. Exploiting rRNA operon copy number to investigate bacterial reproductive strategies. *Nat Microbiol* **1,** 16160 (2016).

118. Drummond, D. A., Bloom, J. D., Adami, C., Wilke, C. O. & Arnold, F. H. Why highly expressed proteins evolve slowly. *Proc. Natl. Acad. Sci. U.S.A.* **102,** 14338–14343 (2005).