Supplementary Notes

Wisdom of crowds for robust gene network inference

Daniel Marbach[♣,1,2], James Costello[♣,3], Robert Küffner[♣,4], Nicci M Vega[3], Robert Prill[6], Diogo M Camacho[3], Kyle R Allison[3], the DREAM5 Consortium, Manolis Kellis[1,2], James J Collins[3,5], and Gustavo Stolovitzky[6]

The DREAM5 Consortium: Andrej Aderhold[25,27], Richard Bonneau[30,31,33], Yukun Chen[9], Francesca Cordero[12,13], Martin Crane[34], Frank Dondelinger[25,26], Mathias Drton[8], Roberto Esposito[12], Rina Foygel[8], Alberto de la Fuente[14], Jan Gertheiss[7], Pierre Geurts[21,22], Alex Greenfield[31], Marco Grzegorczyk[29], Anne-Claire Haury[17,18,19], Benjamin Holmes[1,2], Torsten Hothorn[7], Dirk Husmeier[25], Vân Anh Huynh-Thu[21,22], Alexandre Irrthum[21,22], Guy Karlebach[35], Sophie Lèbre[28], Vincenzo De Leo[14,15], Aviv Madar[30], Subramani Mani[9], Fantine Mordelet[17,18,19,20], Harry Ostrer[32], Zhengyu Ouyang[16], Ravi Pandya[11], Tobias Petri[4], Andrea Pinna[14], Christopher S. Poultney[30], Serena Rezny[8], Heather J. Ruskin[34], Yvan Saeys[23,24], Ron Shamir[35], Alina Sîrbu[34], Mingzhou Song[16], Nicola Soranzo[14], Alexander Statnikov[10], Paola Vera-Licona[17,18,19], Jean-Philippe Vert[17,18,19], Alessia Visconti[12], Haizhou Wang[16], Louis Wehenkel[21,22], Lukas Windhager[4], Yang Zhang[16], and Ralf Zimmer[4]

[1] Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA
[2] Broad Institute of MIT and Harvard, Cambridge, MA, USA
[3] Howard Hughes Medical Institute, Center for BioDynamics, and Dept. of Bioengineering, Boston University, Boston, MA, USA
[4] Ludwig-Maximilians University, Department of Informatics, Amalienstr. 17, 80333 Munich, Germany
[5] Wyss Institute for Biologically Inspired Engineering, Harvard University, Boston, MA, USA
[6] IBM T. J. Watson Research Center, Yorktown Heights, New York, NY, USA
[7] Ludwig-Maximilians University, Department of Statistics, Ludwigstr. 33, 80539 Munich, Germany
[8] University of Chicago, Department of Statistics, Chicago, IL, USA
[9] Department of Biomedical Informatics, Vanderbilt University, Nashville, TN, USA
[10] Center for Health Informatics and Bioinformatics, New York University, New York, NY, USA
[11] Microsoft Research, 1 Microsoft Way, Redmond, WA, USA
[12] Department of Computer Science, Corso Svizzera 185, 10149, Torino, Italy
[13] Department of Clinical and Biological Sciences, Regione Gonzole 10, Orbassano, Italy
[14] CRS4 Bioinformatica, Parco Tecnologico della Sardegna, Edificio 3, Loc. Piscina Manna, 09010 Pula (CA), Italy
[15] Linkalab, Complex Systems Computational Laboratory, 09100 Cagliari (CA), Italy
[16] Department of Computer Science, New Mexico State University, Las Cruces, NM, USA
[17] Mines ParisTech, CBIO, 35 rue Saint-Honoré, Fontainebleau, F-77300, France
[18] Institut Curie, Paris, F-75248, France
[19] INSERM, U900, Paris, F-75248, France
[20] CREST, INSEE, Malakoff, F-92240, France
[21] Department of Electrical Engineering and Computer Science, Systems and Modeling, University of Liège, Belgium
[22] GIGA-Research, Bioinformatics and Modeling, University of Liège, Belgium
[23] Department of Plant Systems Biology, VIB, Gent, Belgium
[24] Department of Plant Biotechnology and Bioinformatics, Ghent University, Gent, Belgium
[25] Biomathematics and Statistics Scotland, Edinburgh & Aberdeen, UK
[26] School of Informatics, University of Edinburgh, UK
[27] School of Biology, University of St Andrews, UK
[28] LSIIT, UMR UdS-CNRS 7005, Illkirch - Université de Strasbourg, France
[29] Department of Statistics, TU Dortmund University, Germany
[30] Department of Biology, Center for Genomics & Systems Biology, New York University, New York, NY, USA
[31] Computational Biology Program, New York University Sackler School of Medicine, New York, NY, USA
[32] Human Genetics Program, Department of Pediatrics, New York University Langone Medical Center, New York, NY, USA
[33] Computer Science Department, Courant institute of Mathematical Sciences, New York University, New York, NY, USA
[34] Centre for Scientific Computing and Complex Systems Modelling, School of Computing, Dublin City University, Ireland
[35] The Blavatnik School of Computer Science, Tel Aviv University, Israel
♣These authors contributed equally

Supplementary Notes

Figures

Tables

Websites

DREAM website (wiki.c2b2.columbia.edu/dream/index.php/D5c4) Current information on the DREAM5 network inference challenge, including anonymized data, challenge description, and evaluation scripts.

M3D website (m3d.bu.edu/dream) The deanonymized expression compendia are made publicly available from the Many Microbe Microarrays Database (M3D).

GP-DREAM web platform (dream.broadinstitute.org) The GP-DREAM platform offers a toolkit of network inference and consensus methods.

Overview of the supplementary notes

Supplementary Note 1, *Challenge description.*
This chapter briefly introduces the general outline of the challenge including the provided datasets, participation requirements and evaluation criteria. These aspects are described in more detail in the subsequent three sections.

Supplementary Note 2, *Gene expression compendia.*
The four gene expression compendia provided to the participants are described here. This section also describes how the set of valid transcription factors was obtained. The set of transcription factors was also available to the participants.

Supplementary Note 3, *Gold standards.*
This section explains the compilation of the gene regulatrory networks, *i.e.* the gold standards used for the evaluation of the challenge participants. In case of *S. cerevisiae*, we describe the compilation of two alternative gold standards used to evaluate the poor performance for *S. cerevisiae*. Note that this and the previous section expand on the *Methods: Expression data and gold standards* section of the main manuscript.

Supplementary Note 4, *Assessment of network inference methods.*
This section describes the metrics used for the assessment of methods, *i.e.* the AUROC, AUPR as well as the overall score. It further details our PCA and Network motif analyses including additional results and figures. It thus expands on three *Methods* sections of the main manuscript, namely *Performance metrics*, *Clustering of inference approaches by principal component analysis (PCA)* and *Network motif analysis.*

Supplementary Note 5, *Data information content.*
This section provides the results and figures for two aspects that are mentioned briefly in the *Discussion* section of the main manuscript, namely the performance differences between the three organisms as well as which kinds of microarray experiments provide the most valuable information for network inference. In particular, the low performance of network inference in *S. cerevisiae* is analyzed and possible reasons are discussed with the corresponding references.

Supplementary Note 6, *Integration of predictions.*
Here, we discuss the details on our ensemble approach integrating 35 network inference approaches to obtain consensus predictions. This section also expands on Box 1 from the main manuscript and describes the simulation approach that generated the theoretical distributions shown in the box.

Supplementary Note 7, *E. coli and S. aureus community networks.*
This section describes the details of how the community networks were constructed along with further analysis of the *S. aureus* network using the RegPrecise database. Module detection and GO term enrichment methods are also fully explained.

Supplementary Note 8, *Experimental validation.*
Experiments were designed to test transcription factor to target gene regulation. The selection criteria for transcription factors along with the full experimental details are covered.

Supplementary Note 9, *Methodological insights.*
This section extends the *Discussion* section of the main manuscript and describes in detail the method specific advantages and disadvantages that we identified. We analyze and discuss the reasons of the network motif specific performance of certain methods and method classes. We also discuss performance improvements due to method specific enhancements as well as performance degradations due to method specific omissions.

Supplementary Note 10, *Network inference methods.*
This section contains the detailed descriptions of 15 network inference methods that were supplied by participants of the challenge. Further, six off-the-shelf methods (marked by an asterisk in the title) are described that were used to compare the performance of participants against state-of-the-art approaches.

1 DREAM5 network inference challenge description

The DREAM5 network inference challenge solicited predictions of genome-scale transcriptional regulatory networks for expression compendia in *E. coli*, *S. cerevisiae*, *S. aureus*, and an *in silico* compendium. Each compendium is represented as an expression matrix of g genes by c chip measurements (**Figure SS1**).

In addition to the gene expression data (see below), a number of descriptive features were supplied for each microarray experiment (e.g., temporal information if the experiment is part of a time series, or the deleted gene if it is a gene deletion experiment). Participants were further given a list of candidate transcription factors for each compendium. To fully anonymize the datasets, for the *E. coli*, *S. cerevisiae*, and *S. aureus* compendia, all genes were assigned an arbitrary gene identifier (e.g. $G_1, G_2, \ldots, G_N$, where there are N genes in a given compendium). Additionally, a set of decoy genes, representing roughly 5% of the compendium, were introduced by randomly selecting gene expression values from the compendium, itself. Gene expression profiles for the *in silico* network were derived from GeneNetWeaver.[5,6]

In order to participate in the challenge, predictions for all four compendia were required. For each network, a list of directed, unsigned edges had to be submitted ordered according to the participant's confidence scores.

The set of submitted predictions were compiled into the prediction matrix (**Figure SS1**) that contains 29 entries from the participating teams, 6 entries from publicly available methods, and an entry from integrating the 29 methods into an additional set of integrated community predictions. In the prediction matrix, each method is represented as an ordered list of predicted gene regulatory interactions, ranked for prediction confidence.

Organism specific gold standards containing the known transcription factor to target gene (transcription factor-target gene) interactions (= true positives) were compiled for assessing the participating approaches. For the evaluation, we considered all transcription factor-target gene pairs that are not part of the gold standards as negatives, although, as the gold standards are based on incomplete knowledge, they might contain yet unknown true interactions.

Expression compendia and gold standards are briefly described in the following sections. The full description of the challenge is available on the challenge website[a].

[a] http://wiki.c2b2.columbia.edu/dream/index.php/D5c4

2 Gene expression compendia

Expression compendia, gene lists, and transcription factor lists for all considered datasets are supplied in **Supplementary Data 1**. The data is also freely available from the DREAM website[b] and the Many Microbe Microarrays Database (M3D)[c].

Staphylococcus aureus. A compendium of microarray data was compiled for *S. aureus*, where all chips are the same Affymetrix platform, the *S. aureus* Genome Array[d]. Chips were downloaded from Gene Expression Omnibus (GEO) (Platform ID: GPL1339), a publicly available web repository hosted by the National Center for Biotechnology Information (NCBI)[e]. In total, 160 chips with available raw data Affymetrix files (.CEL files) were compiled.

Microarray normalization was done using Robust Multi-chip Averaging (RMA)[9] through the software RMAExpress[f]. All 160 chips were uploaded into RMAExpress and normalization was done as one batch. All arrays were background adjusted, quantile normalized, and probesets were summarized using median polish. Normalized data was exported as log-transformed expression values. Mapping of Affymetrix probeset ids to gene ids was done using the library files made available from Affymetrix. Control probesets and probesets that did not map unambiguously to one gene were removed, specifically probeset ids ending in _x, _s, _i were removed. Lastly, if multiple probesets mapped to a single gene, then expression values were averaged within each chip. Completion of these steps resulted in a total of 2,677 genes over the 160 microarrays.

In addition to the gene × condition matrix, DREAM5 participants were also supplied with a list of known or putative transcription factors. Transcription factors were identified based on their Gene Ontology (GO) annotation. We selected genes annotated with a *biological process* related to transcription, namely `GO:0009299;mRNA transcription` or `GO:0006351;transcription, DNA dependent`. Additionally, we also required that a gene be annotated with a *molecular function* of `GO:0003677;DNA binding` or any child terms. A total of 90 genes were designated as potential transcription factors.

Escherichia coli. A compendium of microarray data was compiled for *E. coli*, where all chips are the same Affymetrix platform, the *E. coli* Antisense Genome Array[g]. Chips were downloaded from GEO (Platform ID: GPL199). In total, 805 chips with available raw data Affymetrix files (.CEL files) were compiled.

[b] http://wiki.c2b2.columbia.edu/dream
[c] http://m3d.bu.edu/dream/
[d] http://www.affymetrix.com
[e] http://www.ncbi.nlm.nih.gov/geo
[f] http://rmaexpress.bmbolstad.com
[g] http://www.affymetrix.com

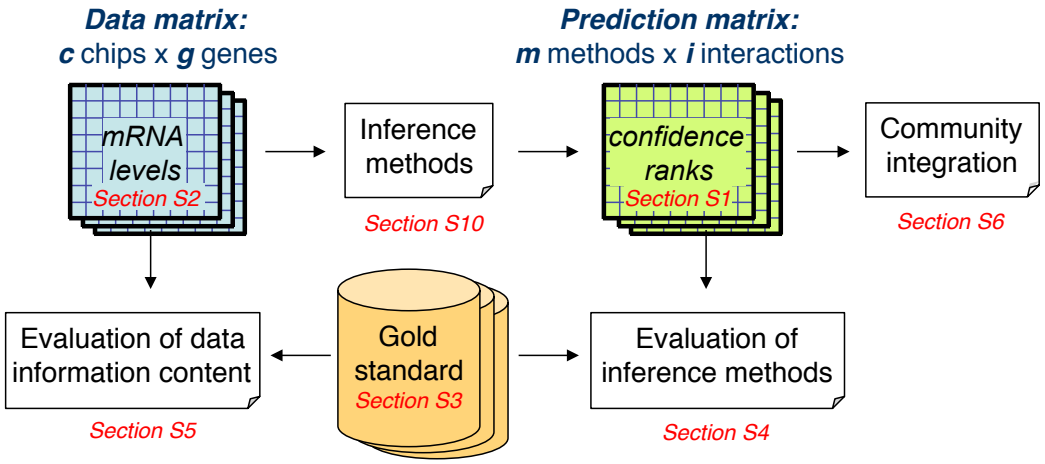

Figure S1: Evaluation of datasets and inference approaches
DREAM5 participants were solicited to infer gene regulatory interactions (prediction matrix, green) from three different expression compendia (data matrix, blue). An additional set of predictions was derived by integrating all teams into community predictions. Analysis steps discussed in the present paper differ according to which of the two matrices are used and whether they depend on the gold standard of known gene regulatory interactions.

Microarray normalization was done as described for *S. aureus* above. Completion of microarray normalization and filtering resulted in a total of 4,297 genes over the 805 microarrays.

The list of potential transcription factors was obtained from two sources. First, transcription factors defined by RegulonDB were used.[31] Second, transcription factors were identified using GO terms as described for *S. aureus* above. A total of 296 genes were designated as potential transcription factors.

Saccharomyces cerevisiae. A compendium of microarray data was compiled for *S. cerevisiae*, where all chips are the same Affymetrix platform, the Affymetrix Yeast Genome S98 Array[h]. Chips were downloaded from GEO (Platform ID: GPL). In total, 536 chips with available raw data Affymetrix files (.CEL files) were compiled.

Microarray normalization was done as described for *S. aureus* above. Completion of microarray normalization and filtering resulted in a total of 5,667 genes over the 536 microarrays.

We used the list of potential transcription factors defined by Zhu et al.[92] We further added transcription factors identified using GO terms as described above. A total of 183 genes were designated as potential transcription factors.

We also evaluated the performance of some algorithms on an independent yeast dataset of 904 chip measurements from the M3D database.[27] This dataset was used for internal comparison and was not provided to the participants of the DREAM5 challenge. The preparation and preprocessing of this dataset was performed in the same way as the datasets of the challenge.

[h] http://www.affymetrix.com

3 Gold standards

In this section, we describe how the gold standards for *E. coli* and *S. cerevisiae* were compiled. *S. aureus* was not used for benchmark evaluation and no gold standard was constructed. For the *in silico* benchmark, the true network structure is known and was used as gold standard. All gold standards are supplied in **Supplementary Data 1**.

Note that in contrast to the *in silico* network, the gold standards for *E. coli* and *S. cerevisiae* are obviously not perfect. For *S. cerevisiae*, we tested a range of alternative gold standards (see below). Of these, we chose the most stringent gold standards for both organisms, which include only interactions with strong experimental support. Thus, most of the edges contained in the gold standards are likely to be true (*i.e.*, the gold standards are expected to have relatively few *false positives*). However, they contain only a subset of the true interactions (*i.e.*, they have many *false negatives*). Thus, predicted interactions that are not part of the gold standard should not be considered incorrect — they may also be newly discovered interactions that are currently missing in the gold standard, as our experimental validation of such novel interactions demonstrates (**Figure 4c** of the main text). Consequently, the reported *precision* and *false positive rate* (**Supplementary Note 4.1**) of network predictions should be considered with caution.

Escherichia coli.

The model organism *E. coli* has a well-studied transcriptional regulatory network, which makes it well-suited as a benchmark for network inference. Known transcriptional interactions are collected in the manually curated EcoCyc[42] and RegulonDB[31] databases (the two databases are synchronized). Each interaction is annotated with a set of evidence codes, which are classified as either strong or weak evidence[i]. The *E. coli* gold standard was constructed from RegulonDB Release 6.8. Only transcriptional interactions with at least one strong evidence were included (2,066 interactions).

Saccharomyces cerevisiae.

Several large-scale studies and databases have produced genome-wide regulatory networks for *S. cerevisiae*. We have confirmed that our results are consistent and reproducible across several alternative gold standards. In particular, we find that the performance of inference methods is low for *S. cerevisiae* (see *Discussion* of the main text and **Fig. SS8** & **Fig. SS9**) independently of the gold standard. In total, we have tested 16 gold standards derived from three sources.[1,39,50] In what follows, we describe the different gold standards and discuss the measured performance (AUPR and AUROC, see **Supplementary Note 4.1**) of inference methods on these gold standards (**Fig. SS2**).

The first set of gold standards was obtained from the study of MacIsaac *et al.*,[50] which is based on a re-analysis of ChIP-chip data for 203 transcription factors from Harbison *et al.*[33] Regulatory interactions were identified based on measured binding (ChIP) and/or presence of evolutionary conserved motifs of the transcription factor in the intergenic region upstream of target genes. By varying the thresholds required for binding and evolutionary conservation of motifs, different versions of the network were obtained. We considered all nine versions available from the author's website[j] (genes and transcription factors that are not part of our expression compendium were excluded). The gold standard based on the most stringent thresholds, which includes only interactions with strong evidence of binding and a strongly conserved motif, also leads to the "strongest signal" (highest AUROCs and biggest fold-improvements for the AUPRs[k]), *i.e.*, it "agrees best" with the inferred networks. In contrast, the gold standard based on the loosest thresholds, which requires no ChIP binding and no evolutionary conservation (only the presence of a motif), leads to the "weakest signal" — likely because it contains many false positives (motif instances that are not bound and/or not functional in vivo). Incidentally, our observations confirm the conclusion of MacIsaac *et al.* and others[43] that evolutionary conservation of motifs can be used as a signal to improve the quality of ChIP-based networks.

The second set of gold standards was derived from the study of Hu, Killion & Iyer,[39] which conducted a comprehensive mRNA profiling experiment of 269 yeast transcription factor deletion mutants. Genes were considered a target of a given transcription factor if they exhibited a fold-change above a certain threshold. Independently of the considered threshold, the overlap of the inferred networks and this gold standard is not better than expected by chance (AUPR and AUROC values are similar to those expected for random predictions, **Fig. SS2**).

Finally we evaluated a gold standard from the curated YEASTRACT database,[1] which compiles direct and indirect evidence for yeast gene regulatory interactions from more than 1,200 publications. While direct evidence (28,336 interactions, as of version 1.1503, Jun 26, 2010) was predominantly derived from ChIP experiments, indirect evidence (21,847 interactions) was gathered from expression measurements of transcription factor deletion or overexpression mutants. In the present document we will refer to the YEASTRACT gold standard as the intersection of interactions with both direct and indirect evidence (2,528 interactions, after filtering out transcription factors and genes not part of our expression compendium). The YEASTRACT gold standard leads to slightly better AUPR and AUROC values than the high-confidence network of MacIsaac *et al*, however, the better agreement between the inferred networks and YEASTRACT may be because YEASTRACT includes expression data as evidence and is thus not completely independent from the inferred networks, which are also expression-based. In contrast, the MacIsaac gold standard is based on orthogonal datasets (ChIP and conserved motifs) and does not incorporate expression-based evidence.

Based on the above observations, we used the network based on the most stringent thresholds from MacIsaac *et al.* as gold standard for all method related assessments and all other analyses, unless noted otherwise.

[i] http://regulondb.ccg.unam.mx/evidenceClassification.jsp

[j] http://fraenkel.mit.edu/improved_map

[k] The AUPR of network predictions depends on the *connectivity* of the gold standard, *i.e.*, the ratio of positives (present edges) to negatives (absent edges). The more densely connected the gold standard, the easier it is to correctly "guess" true edges and obtain a high AUPR. For example, the same predictions have an AUPR of ∼32% on the densely connected (low confidence) gold standard of the 1st row, and ∼2% on the loosely connected (high confidence) gold standard of the 9th row in **Fig. SS2**. Thus, the absolute value of the AUPR should be considered with caution and always be compared to the expected AUPR of a random prediction, for instance.

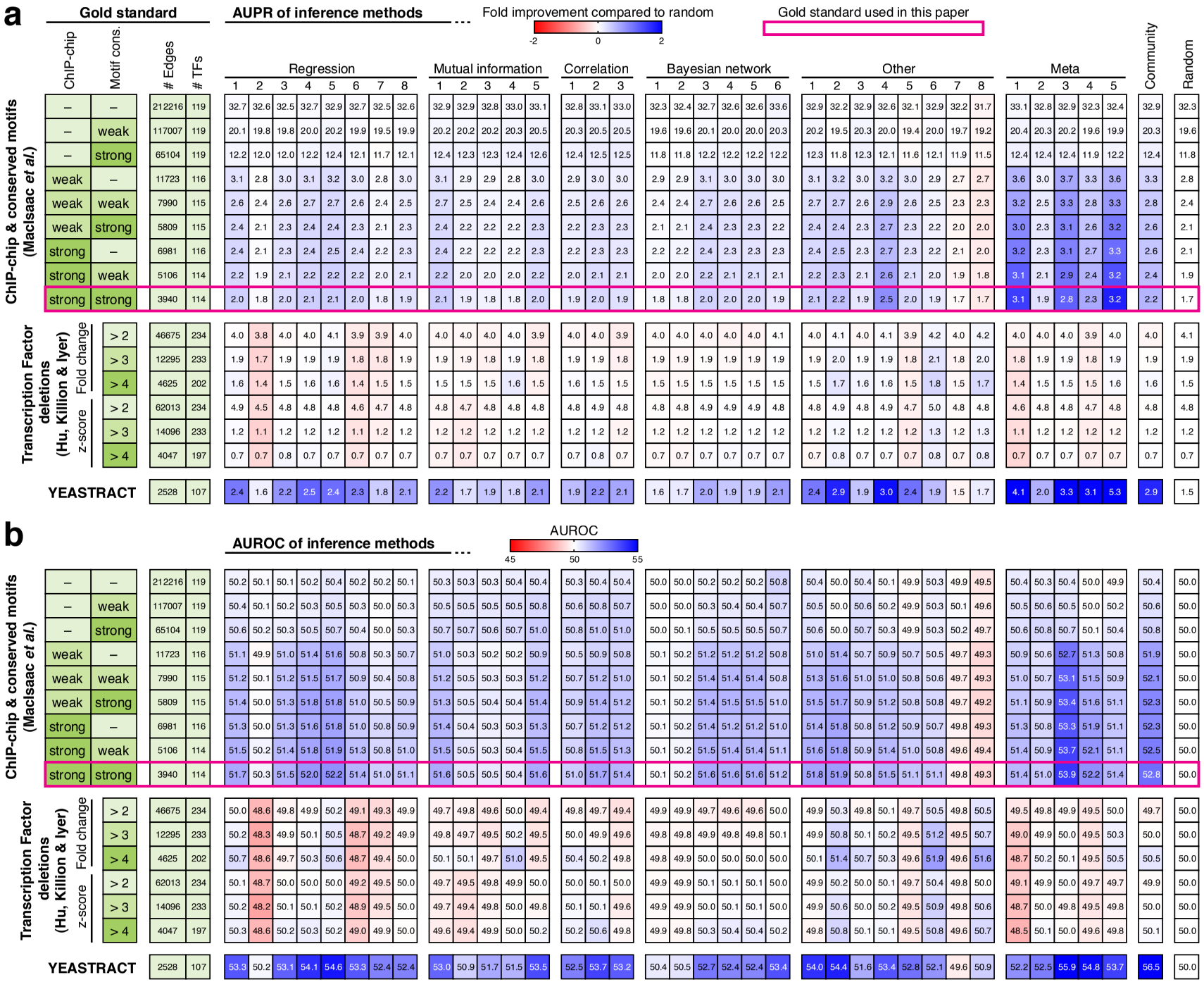

Figure S2: AUPR and AUROC values using different gold standards for *S. cerevisiae*

4 Assessment of network inference methods

4.1 Performance assessment

We used a standard approach established in previous editions of the challenge to evaluate network predictions.[55,69,81] Briefly, we evaluate network predictions as a binary classification task (edges are predicted to be present or absent) and use standard performance metrics from machine learning, specifically precision vs. recall (PR) and receiver operating characteristic (ROC) curves.[23] We further evaluate predictions statistically and compute an *overall score* summarizing the performance across networks. Scoring metrics are described below. Evaluation results are reported in:

- **Figure SS3**: PR and ROC curves;
- **Figure SS4**: Comparison of the ranking across compendia;
- **Supplementary Data 2**: Table with the area under the curves and scores;
- **Supplementary Data 1**: Evaluation scripts.

PR and ROC curves.

To score the ranked lists of interactions (the prediction format is described in **Supplementary Note 1**) against a binary gold standard, performance was assessed by the area under the ROC curve (AUROC, true positive rate vs. false positive rate) and the area under the precision vs. recall curve (AUPR). Expressions for *true positive rate* (TPR), *false positive rate* (FPR), *precision* and *recall* as a function of the cutoff (k) in the edge list are as follows:[69,81]

$$\text{recall}(k) = \frac{\text{TP}(k)}{\text{P}},$$

where $\text{TP}(k)$ is the number of true positives in the top k predictions in the edge list, and P is the number of positives in the gold standard.

$$\text{precision}(k) = \frac{\text{TP}(k)}{\text{TP}(k) + \text{FP}(k)} = \frac{\text{TP}(k)}{k},$$

where $\text{FP}(k)$ is the number of false positives at cutoff k in the edge list. The *true positive rate* is equivalent to recall and is defined as:

$$\text{TPR}(k) = \frac{\text{TP}(k)}{\text{P}}.$$

The *false positive rate* is the fraction of negatives that are incorrectly predicted at cutoff k

$$\text{FPR}(k) = \frac{\text{FP}(k)}{\text{N}},$$

where N is the number of negatives in the gold standard.

Note that the length of the prediction lists was limited to at most 100,000 edges (some teams also submitted shorter lists). Edges not included in the list are thus effectively predicted to be absent. We extended the PR and ROC curves in an analytical way after the end of the list by assuming a random ordering of the remaining edges, as described.[81] Note that AUROC values can vary by up to 7.5 percentage points when considering truncated vs. full lists of predictions (the AUPR values, on the other hand, don't vary significantly — the reason is discussed in the legend of **Figure SS3**).

Note that predictions for transcription factors and genes that are not part of the gold standard, i.e., for which no experimentally supported interactions exist, were ignored in this evaluation.

Empirical p-values and overall scores.

AUROC and AUPR values were separately transformed into p-values by simulating a null distribution for a large number (25,000) of random networks. We fit the histogram of the randomly obtained AUROC and AUPR values using stretched exponentials as previously described[81] to extrapolate the distribution to values beyond the immediate range of the histogram. Note that p-values obtained this way depend on how "random" is defined. We chose to construct random edge lists by sampling edges from the submitted edge lists of the participants. By design, an "average" performance should have a p-value of approximately 0.5 for the two metrics. We call this procedure "grading on a curve" since the p-values are designed to identify the best and worst relative performance but have no absolute interpretation.

Overall scores are derived from the AUPR or the AUROC for the three different networks by calculating the geometric mean of the network specific p-values. We report the negative $\log_{10}$ of this value as a score, or equivalently,

$$\text{ROC score} = \frac{1}{3}\sum_{i=1}^{3} -log_{10}\, p_{\text{ROC}i}$$

$$\text{PR score} = \frac{1}{3}\sum_{i=1}^{3} -log_{10}\, p_{\text{PR}i}.$$

Finally, an overall score is obtained as the mean of the AUROC and AUPR derived scores. A high score corresponds thus to low (significant) p-values.

4.2 Clustering of methods by PCA

To depict similarities and differences between inference approaches via Principal Component Analysis (PCA),

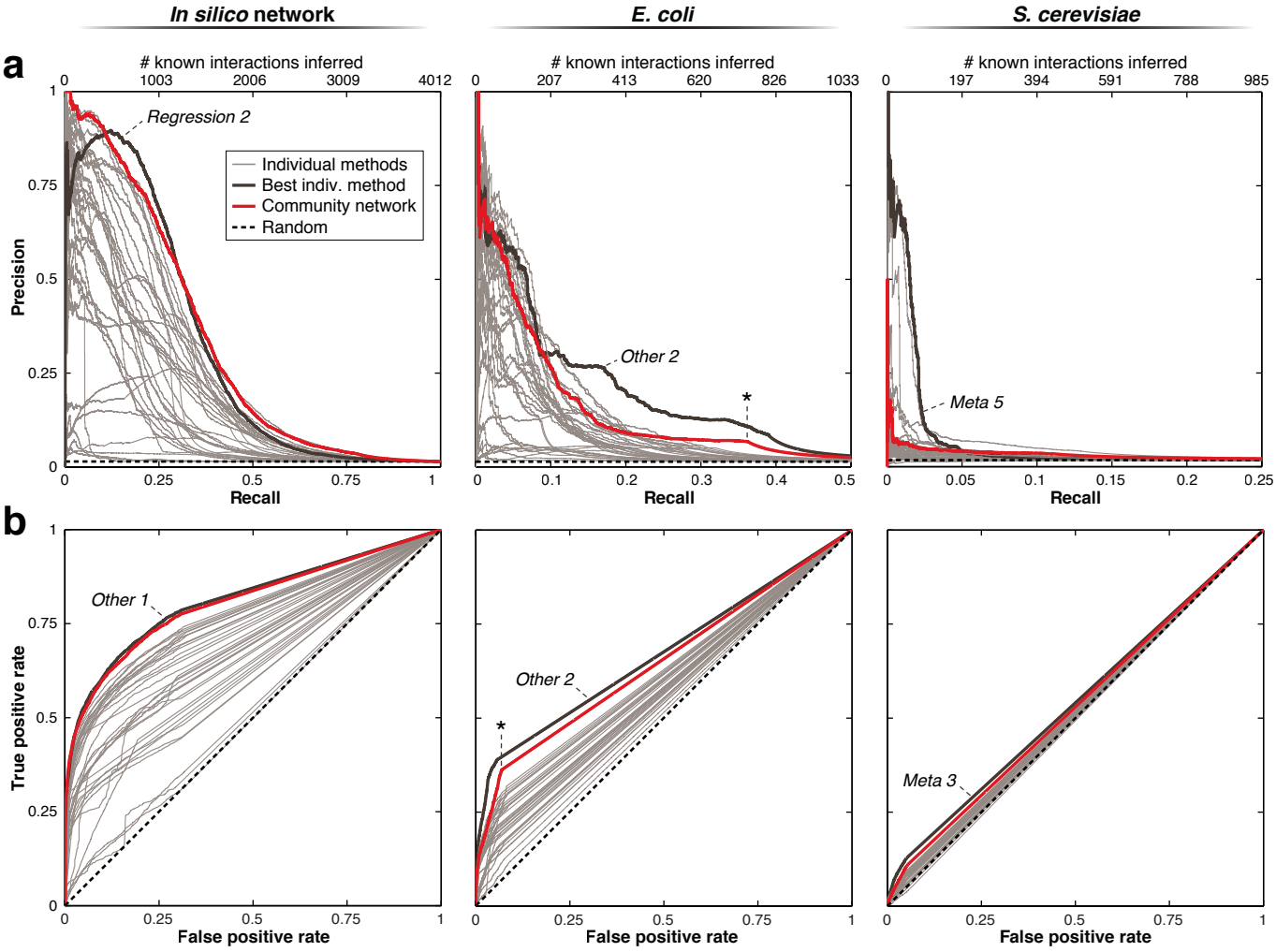

Figure S3: PR and ROC curves for individual methods and community predictions

PR curves are shown in panel **a** (note that the *recall*-axis has a different scale for each of the three networks), ROC curves are shown in panel **b**. The performance of individual methods varies strongly across networks and a different inference method performs best for each network. The integrated community predictions, on the other hand, are more robust: the community ranks 1st, 3rd and 7th on the three networks according to the area under the PR curve, and it ranks 2nd on each network according to the area under the ROC curve (cf. **Fig. SS4**). Since the prediction lists were limited to 100k edges, the curves were extended analytically beyond this point (assuming random ordering of the remaining edges). For example, this point has been marked for the community prediction of *E. coli* (*). Whereas the analytical extension accounts only for a small part of the PR curve, it accounts for most of the ROC curve. Indeed, the PR curves are more informative than the ROC curves for visualizing the performance at the top of the prediction lists — for example, the high precision of the most confident predictions of *Meta 5* for *S. cerevisiae* is not apparent in the ROC curve (see Ref. 23 for an excellent discussion of the relation between PR and ROC curves).

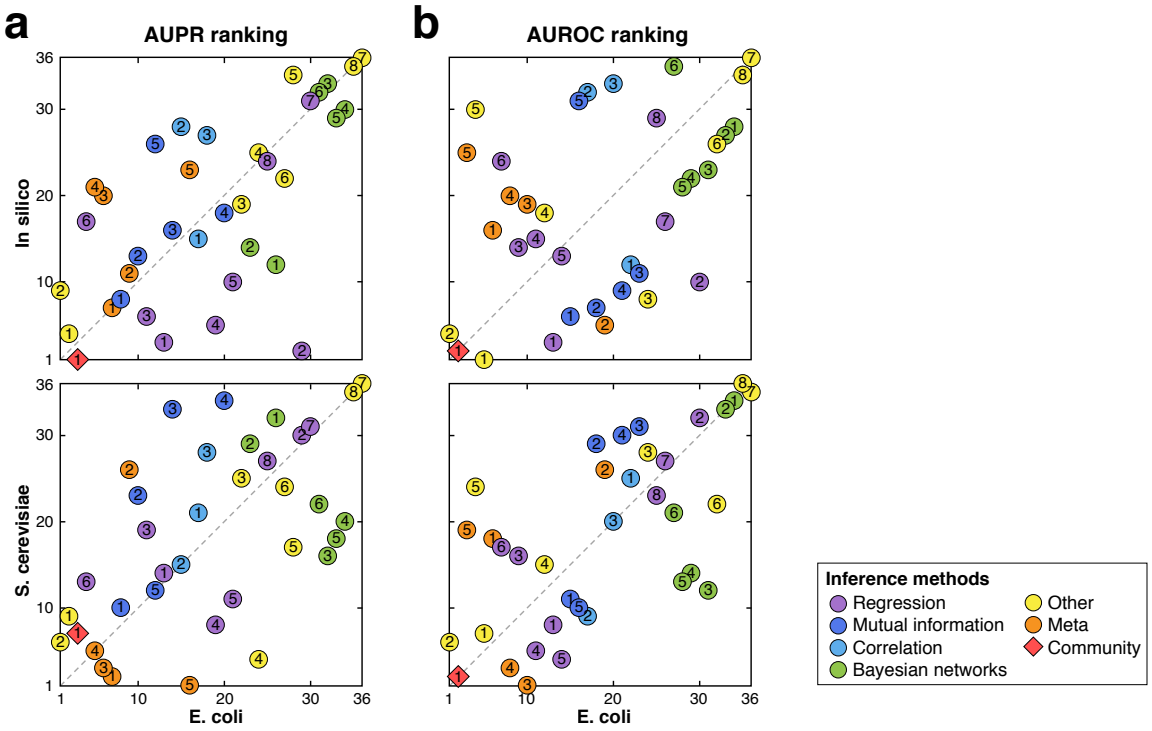

Figure S4: Comparison of the method rankings across compendia

The scatter plots compare the performance of inference methods across the three microarray compendia. Shown are the ranks of each method based on the AUPR (**a**) and AUROC (**b**). The ranking correlates poorly across compendia, *i.e.*, the performance of individual methods varies strongly. The performance of the community, on the other hand, is robust (bottom left corner of the plots).

we extracted for each inference method a feature vector from the prediction matrix (**Figure SS1**). These feature vectors consist of ranks assigned by the given inference method, where small ranks correspond to a high confidence in the corresponding predicted interactions. We restricted the prediction matrix to interactions predicted by at least three inference methods, yielding 292,654 interactions for the *E. coli* network, for instance. The prediction matrix contains in row j at position i the rank assigned by method j to the interaction i (or the maximum rank if interaction i was not included in the prediction list of method j). This represents each of the inference methods as a point in a 292,654-dimensional space.

Prior to visualization, we applied PCA to reduce the dimensionality of the data. First, we performed PCA for each of the four networks individually. An additional PCA was performed by constructing a combined prediction matrix by concatenation of the *in silico*, *E. coli*, and *S. cerevisiae* prediction matrices. PCA was performed by singular value decomposition using SVDLIBC[1] and the default parameters defined therein.

For a discussion of PCA related results the reader is also referred to **Supplementary Note 9**, the main text and **Figure 2b**, which shows the PCA on the combined prediction matrix. The visualization of the compendium specific prediction matrices are shown in **Figure SS5** (the 2nd vs. 3rd PC is shown — the 1st PC accounts mainly for the overall performance and is thus less characteristic for the different inference approaches, see **Fig. SS6**). Note that the PCA analysis does not use the gold standards. In contrast to the network motif analysis described in the next section, it can thus be applied even if no gold standard is available, as in the case of *S. aureus* in the present assessment.

We find that results are consistent across the four compendia. The PCA reveals four clusters of methods that largely confirm our method categorization scheme from **Table 1** (main text). Clusters 1, 2, and 3 overlap strongly with the method categories *Regression*, *Bayesian networks*, and *Correlation/Mutual information*, respectively. Although the performance between approaches from the same category can vary strongly (e.g., *Regression 1* and *Regression 8* rank 3rd and 33rd, respectively), the similarity between their predictions was picked up clearly in the PCA.

Unrelated methods from the categories *Meta* and *Other* are not represented by the depicted principal components

[1] tedlab.mit.edu/~dr/SVDLIBC

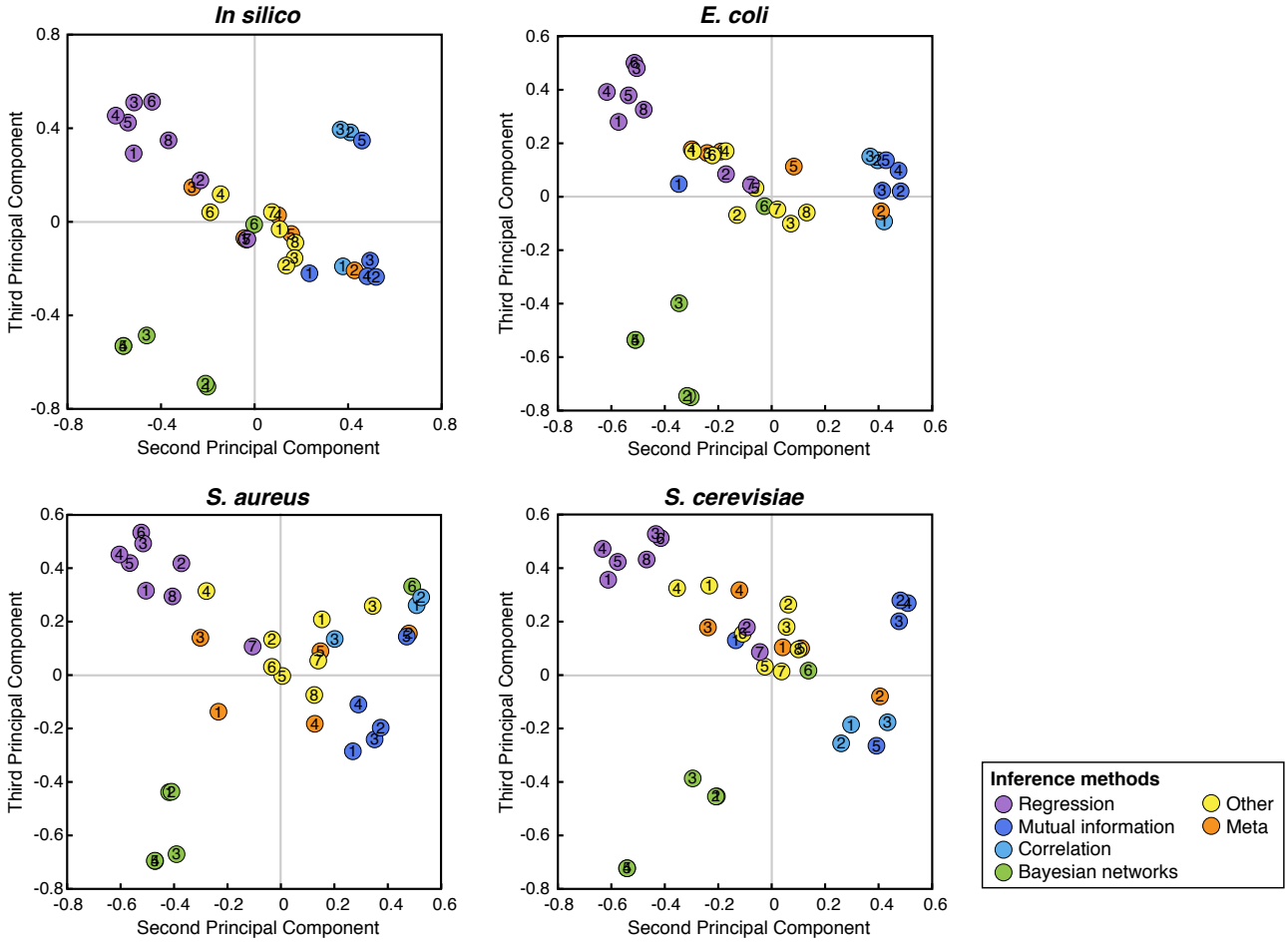

Figure S5: PCA for individual compendia

Methods that tend to predict similar interactions are clustered via PCA. Shown are the 2nd vs. 3rd principal components (the 1st PC is shown in **Fig. SS6**). Results for individual compendia (shown here) are largely consistent with the PCA across all compendia (shown in **Fig. 2b** of the main text).

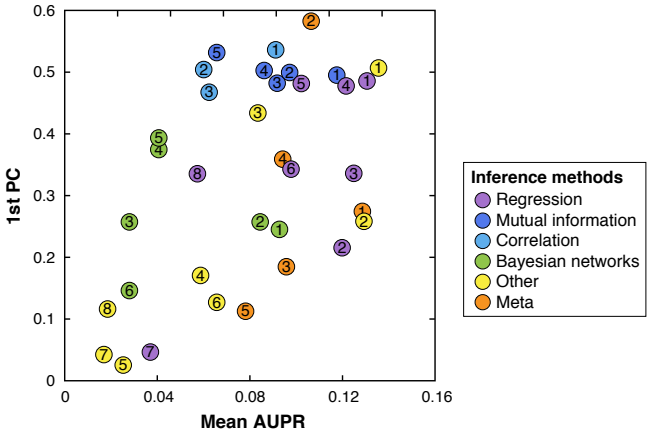

Figure S6: First principal component

The 1st PC correlates ($r = 0.54$) with overall performance (mean AUPR).

and are thus grouped near the origin in the PCA plot (cluster 4).

4.3 Network motif analysis

The goal of the network-motif analysis is to evaluate, for a given network inference method, whether an edge $A \to B$ that is part of a given motif (**Figure SS7**) is systematically predicted less (or more) reliably than expected.[55] We considered six different motifs. For each inference method, we determined the average rank r_m (*i.e.*, the prediction confidence) assigned to all edges of a given motif type m. We further determined the average rank $r_{\bar{m}}$ of all edges that are part of the complementary motif $\bar{m}$. The prediction bias is given by the difference $r_m - r_{\bar{m}}$. A positive bias means that this type of edge was ranked higher, *i.e.*, predicted more confidently, than other (complementary) edges. Conversely, a negative bias indicates a reduced

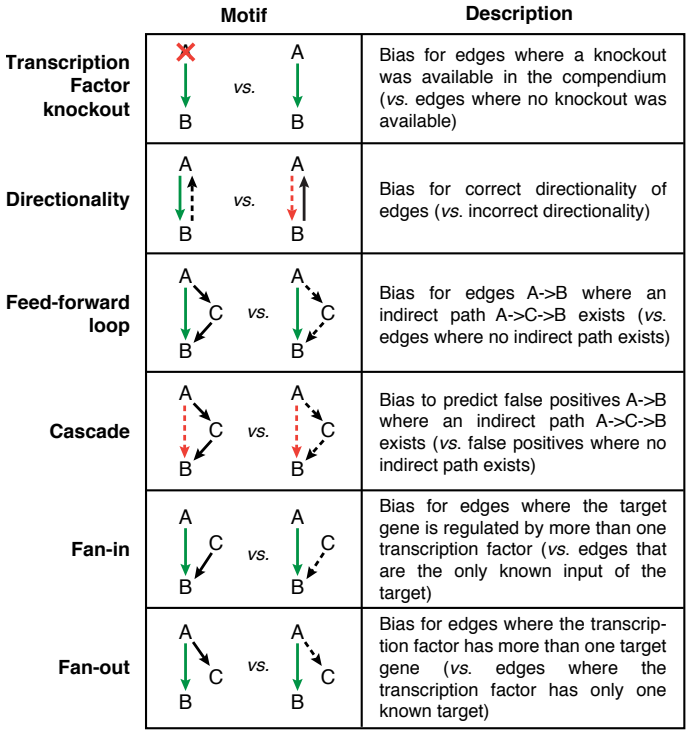

Figure S7: Definition of motif types used to analyze prediction biases
Network motifs were defined for the analysis of method specific prediction biases. The average rank (prediction confidence) assigned to edges $A \rightarrow B$ of a given motif type (*first column*) was compared to the average rank assigned to edges that are *not* part of this motif type (*i.e.*, that are part of the complementary motif type defined in the *second column*).

prediction confidence for this edge type. The six types of motifs and their complementary motifs are defined in **Figure SS7**.

Note that motif instances overlap in the network. Nevertheless, a given edge $A \rightarrow B$ is only counted once for a given motif regardless whether it is present in only one or several overlapping instances of that motif.

The relative advantages and limitations of different inference approaches to distinguish and exploit such local motifs are discussed in the main text (**Fig. 2c**) and in **Supplementary Note 9**.

5 Data information content

5.1 Estimation of inference difficulty

A fundamental assumption of expression-based network inference algorithms is that mRNA levels of regulators and their targets display some degree of mutual dependency. As a coarse estimate of inference difficulty, we thus analyzed to what extent regulatory interactions between transcription factors and their known targets are reflected by dependencies between their respective expression profiles. As measures of dependency, we selected mutual information and Pearson's correlation, as they are utilized for the detection of gene regulatory interactions by many of the commonly-used, and participant-submitted inference approaches. Pearson's correlation distinguishes positive from negative correlations but does not, in contrast to mutual information, enable the detection of non-linear dependencies. See **Supplementary Note 10.7** and **Supplementary Note 10.10** for a description of the calculations used for mutual information and Pearson's correlation, respectively.

The distribution of dependencies are shown in **Figure SS8**. Mutual information and Pearson's correlation were computed for each transcription factor-gene pair across all chip measurements from a given expression compendium. The set of possible gene pairs was partitioned into three subsets: *interacting* (*i.e.*, a transcription factor and one of its known target genes), *non-interacting*, and *co-regulated* pairs. The latter depict the dependencies between pairs of genes that are regulated by identical sets of transcription factors.

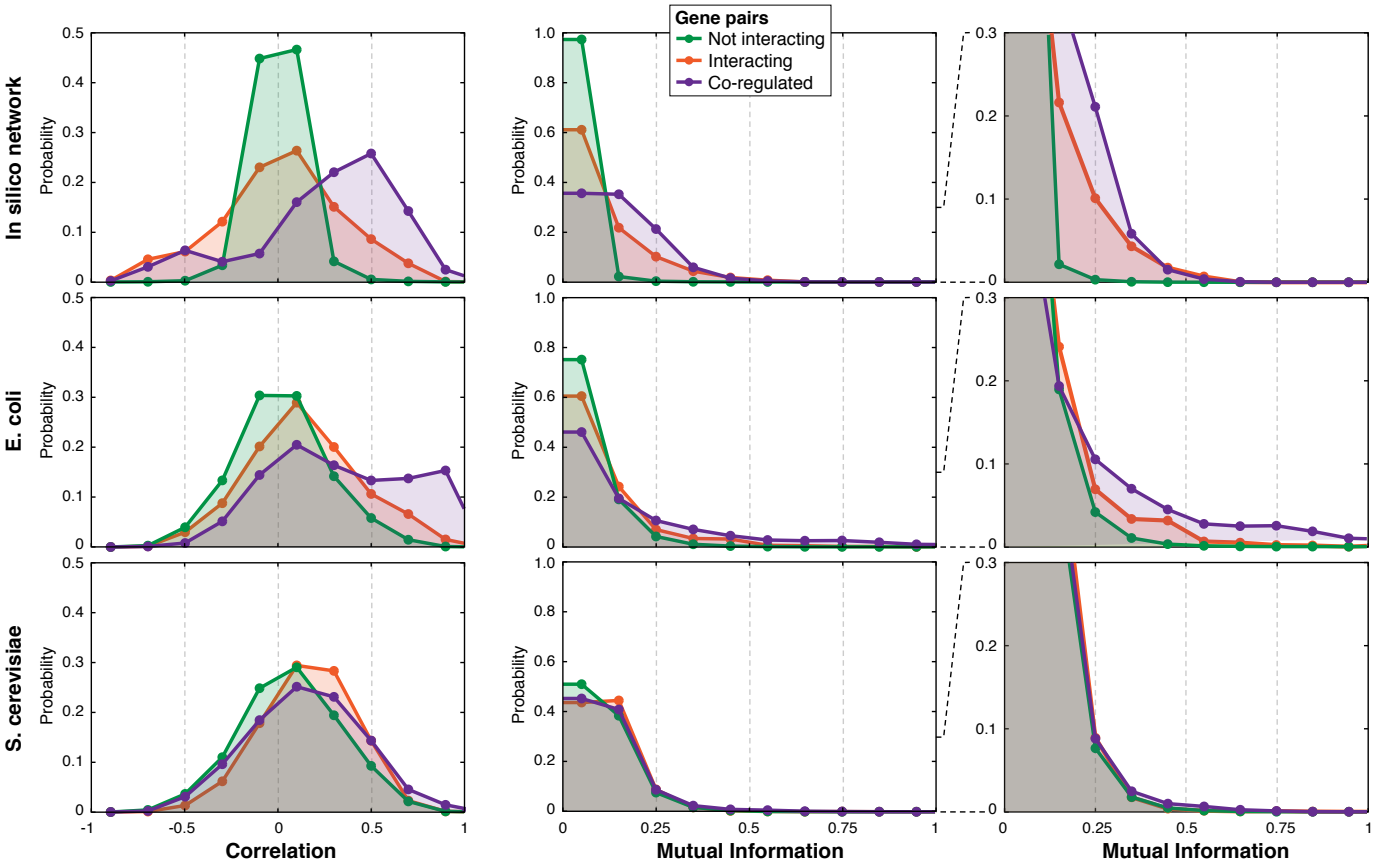

Figure S8: Mutual dependency of mRNA levels between transcription factors and target genes in different compendia

In both the *in silico* and *E. coli* compendia, mutual dependencies between expression profiles of transcription factors and their known target genes exceed the dependencies observed between non-interacting gene pairs. This is true regardless whether Pearson's correlation or mutual information is used to measure the dependencies. In *S. cerevisiae*, dependency distributions are almost identical, suggesting that it is much more difficult to detect transcription factor-gene interactions based on dependencies dervied from the compenium of gene expression data used in the DREAM5 network inference challenge.

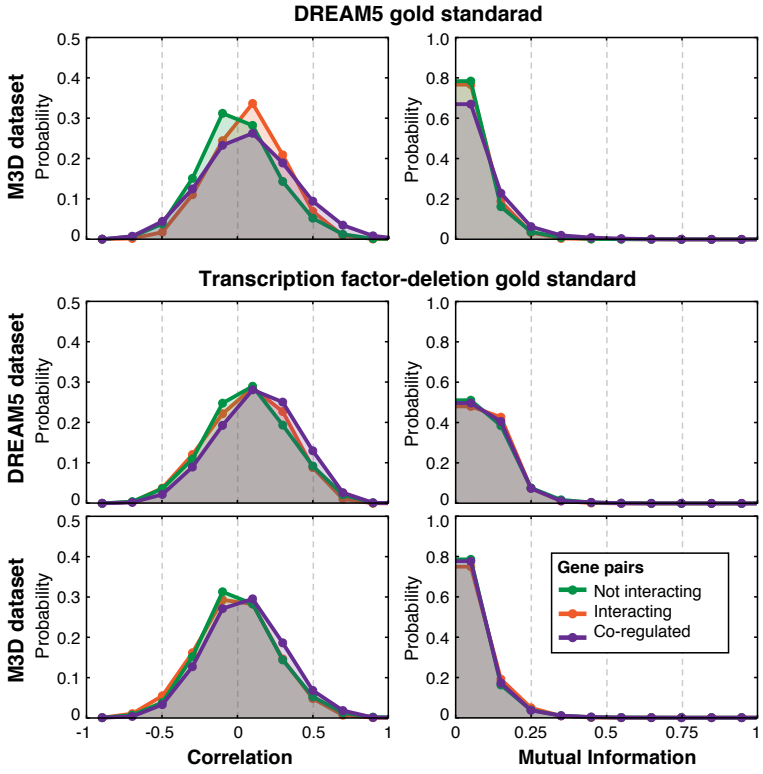

Figure S9: Transcription factor-target gene dependency of mRNA levels for alternative yeast compendia and gold standards
Results from **Figure SS8** are confirmed using an alternative microarray compendium and gold standard for *S. cerevisiae*: interacting and non-interacting gene pairs exhibit very similar distributions independently of the compendium and gold standard.

Examining the existence and extent of such dependencies yields a first impression of the level of difficulty of network inference in the three compendia. The *in silico* and *E. coli* compendia show a marked enrichment of strong dependencies between interacting gene pairs. This enrichment occurs for both positive and negative correlations in the *in silico* data, while only positive correlations are enriched in *E. coli* gene regulatory interactions. Absence of anti-correlation *in vivo*, even though conceivable theoretically and thus present in the *in silico* network, is consistent with previous studies.[37,60] Compared to *in silico* and *E. coli*, virtually no enrichment of strong dependencies is observed in *S. cerevisiae* gene regulatory interactions. Similar results are obtained with an additional, independent *S. cerevisiae* microarray compendium[27] and gold standard[39] (**Fig. SS9**, see **Supplementary Note 2** and **Supplementary Note 3** for a description of this dataset and gold standard).

Next we investigated whether the presence of operons in *E. coli* could be a reason for the higher performance of network inference in comparison to *S. cerevisiae*. At this point, we would like to note that participants had no information on the identity of the organisms or the genes during the challenge as gene names were replaced by random identifiers. The information of which genes are organized in operons was thus not available to the participants. Nevertheless, three participants specifically used clustering to deduce regulons. This was not successful as they rather scored below average: Other 6 (overall rank 31/35), Meta 5 (overall 18/35) and Regression 8 (overall 32/35). In contrast, the teams that ranked in the top third for *E. coli* did not take operons into account. These two points, namely the absence of information on operons during the challenge and the low performance of approaches that aimed to deduce operons, seem to argue that substantial operon-overfitting is not a likely reason for performance differences between *E. coli* and *S. cerevisiae*. Rather than the correlation structure within genes of the same operon, the correlation structure between transcription factors and their targets accounts for the observed performance differences.

The absence of such correlation structures or dependencies between mRNA levels of transcription factors and their targets in *S. cerevisiae* has two main reasons. The first reason is that the *S. cerevisiae* gold standards are more heavily dependent on chromatin immunoprecipi-

tation (ChIP) experiments than the *E. coli* gold standards. The physical binding of a transcription factor to the promotor region of a target gene is a required but not sufficient condition for effective gene regulation. transcription factor-target gene pairs derived from ChIP thus contain many false positives. However, in addition to strong evidence of binding, we also required the presence of strongly conserved transcription factor-binding motifs, which should filter out at least part of the false positives from the ChIP data[50] (**Supplementary Note 3**). The lesser quality of the *S. cerevisiae* gold standard compared to *E. coli* thus cannot explain the complete absence of dependency between transcription factor and target mRNA levels. The second reason for this absence of dependency is the greater complexity of gene regulation in eukaryotes. In particular, additional layers of regulation at the post-transcriptional and chromatin levels lead to a reduced dependency of mRNA concentration between regulators and targets, and thus an increased difficulty for expression-based inference of eukaryotic networks.[39,60] Hu *et al.*[39] found gene regulatory network inference in yeast very challenging. Despite the availability of microarray measurements for all 270 *S. cerevisiae* transcription factors, the performance of inference was hardly better than guessing. They discussed four potential reasons for the higher difficulty of inference in eucaryotes, namely the lack of operons, the high functional overlaps of transcription factors, transcription factors regulated on the protein rather than the transcript level as well as the fact that transcription factors regulate a high number of targets, albeit only mildly. Michoel *et al.*[60] found that the overall performance of network inference in *S. cerevisiae* is quite low and that better results can only be achieved on specific selected subsystems. According to Wu *et al.*,[88] *independent measurements of the actual activities of the transcription factors* would be required in order to resolve the subtle expression dependencies between transcription factors and their targets. Also, Herrgard *et al.*[37] pointed out that a lack of correlation might be due to the fact that *most transcription factors themselves are not significantly transcriptionally regulated and their expression remains at a low constitutive level. Instead, many transcription factors such as Mig1 glucose repressor in yeast are regulated by phosphorylation and localization as well as other post-transcriptional regulatory mechanisms.*

5.2 Information content of different experiment types

We sought to evaluate how informative different experiment types, such as time courses, drug perturbations, and genetic perturbations, are for network inference. To evaluate the information content of different experiment types across the three compendia, we first computed a weight for each individual microarray chip using a machine learning framework (described below). Subsequently, individual weights were averaged across chips of particular experiment types, such as time series, drug perturbations, gene and transcription factor knockout or overexpression experiments, as well as combinations thereof (**Fig. SS10**).

The chip weights were computed as follows. We applied feature subset selection approaches based on machine learning classifiers to the *in silico*, *E. coli*, and *S. cerevisiae* expression compendia to estimate the significance of individual features (here: microarray chips) with respect to their ability to correctly identify transcriptional regulatory interactions. Thus, an individual local binary classifier[61] was constructed for each transcription factor to distinguish known target genes from non-targets as defined by the given gold standards. As classifiers, both decision trees and support vector machines (SVMs) were employed. Decision trees were constructed[72] by selecting microarray chips as decision nodes based on information gain, which we interpret as chip specific weights. In addition, linear SVMs[17] were trained by optimizing SVM coefficients (α weights). In the case of linear SVM models, chip specific weights can be computed by an α-weighted linear combination of the support vectors. Default parameters were used for the training of both classifiers. As suggested by Mordelet and Vert,[61] a three-fold cross-validation was employed.

This analysis shows that direct transcription factor manipulations, *i.e.*, knockout/deletion or overexpression, are most informative for the inference of interactions involving this transcription factor (**Fig. SS10**). In comparison to these transcription factor specific perturbations, the remaining categories have on average much lower weights. The difference between weights of transcription factor specific knockouts and the other kinds of experiments is more pronounced for decision trees (that optimize chip weights) as compared to SVMs (that optimize example weights). Note that the *S. cerevisiae* compendium included only three transcription factor-deletion measurements for a single transcription factor (GCN4): results are therefore not as reliable as for the other compendia. Transcription factor overexpression experiments were also very rare in the examined compendia and were thus included in the same category as the knockouts in **Figure SS10**. We compared the information content of knockout *versus* overexpression experiments in a different *E. coli* compendium from the M3D database[27] (**Supplementary Note 2**) and found that chip weights from transcription factor specific overexpression are comparable to the knockout weights.

We conclude that direct transcription factor perturbations (knockout/overexpression) are the most informative experiments on average. However, other kinds of experimental conditions can be similarly informative for the inference of

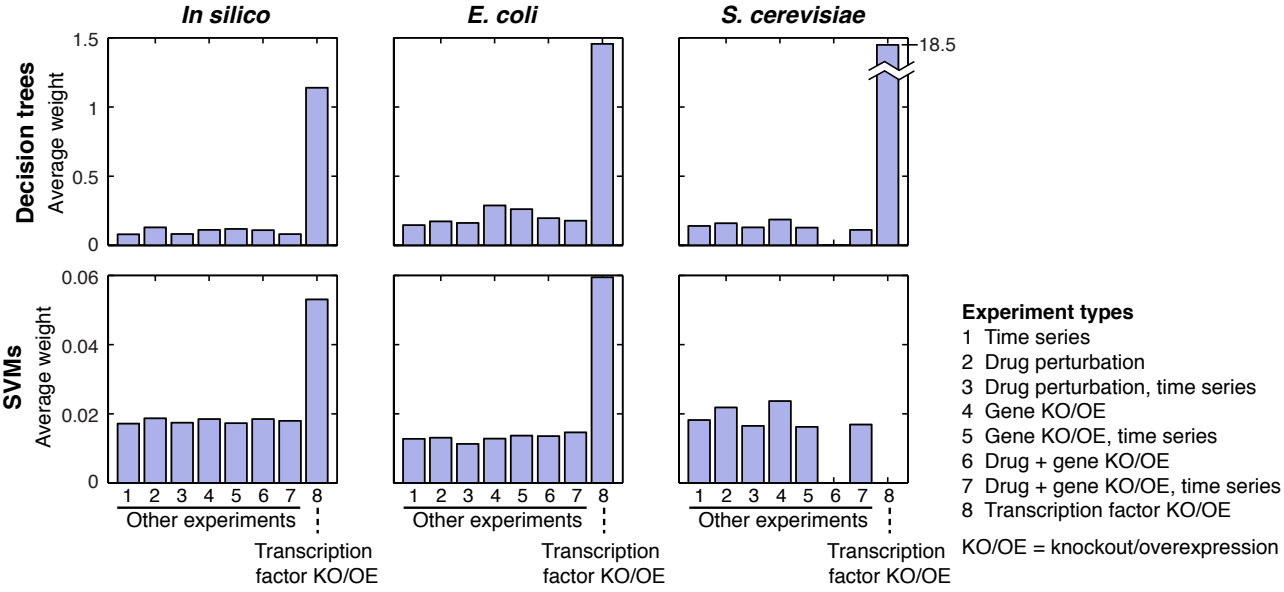

Figure S10: Information content of different experiment types for network inference
Feature selection was employed to estimate the utility of microarray chips from different experiment types to correctly identify gene regulatory interactions. Microarray measurements were categorized into time series, drug perturbations, gene knockout/overexpression (KO/OE) experiments, as well as combinations thereof. The bar at position 8 only refers to the weight of transcription factor knock out/over-expression experiments for inferring interactions involving that transcription factor. Knockout or overexpression of a transcription factor is by far the most informative experiment to detect regulatory interactions, independently of the classifier and compendium used. Note that in the *S. cerevisiae* compendium there were only three transcription factor deletion experiments available (all for the same transcription factor), results are thus not as reliable as for the other two compendia.

the targets of certain transcription factors (**Fig. SS11**), but here it will be much more difficult to decide *a priori* which kind of conditions that may be.

6 Integration of predictions

6.1 Why community integration can outperform the best individuals

Of the many ways to integrate results from an ensemble of predictions,[47] we have chosen the Borda count election method. It was originally developed by 18th-century political scientist Jean-Charles de Borda as a method to select candidates in a democratic election.[12] In this method, voters rank candidates in order of preference, and the winner of the election is the candidate with the best average rank. Similarly, DREAM5 participants provide a ranked list of transcription factor-target gene predictions. **Figure S12a** exemplifies three possible prediction lists where we denote transcription factor-target gene interactions by the letters $A, B, C, \ldots, X, Y, Z$. For example, Team 2 has high confidence that interaction A is a true interaction, but Team 1 and Team 3 are less certain about the truth of this interaction, having placed it at position 5 and 20 of the ranked lists, respectively. The same is true at the other end of the list. Team 2 considered Y to be a very unlikely interaction, but Team 3 located it at position 10 of its ranked list.

In this section, we wish to build some intuition to understand why the integration of predictions can outperform individual teams. Let us assume that we have P true transcription factor-target gene interactions (the *positives*) and N non-interacting transcription factor-target gene pairs (the *negatives*). In total, there are $T = N + P$ possible predictions. Different applications of the same network reconstruction algorithm that attempt to infer a given network can create lists of predictions in which the same interaction I can be placed in different rank positions. This will depend on the set of experiments used by the algorithm, random aspects of the method itself, biological variability, *etc.* Therefore, each method can be characterized by the probability $p_{\text{pos}}(r, I)$ that it places a given true interaction I at rank r. We make the simplifying assumption that this probability is the same for all interactions (*i.e.*, $p_{\text{pos}}(r, I)$ is independent of I), and refer to this probability as $p_{\text{pos}}(r)$. By randomly ranking interactions, we expect $p_{\text{pos}}(r) = 1/T$. However, if a method has better than random predictive power, there

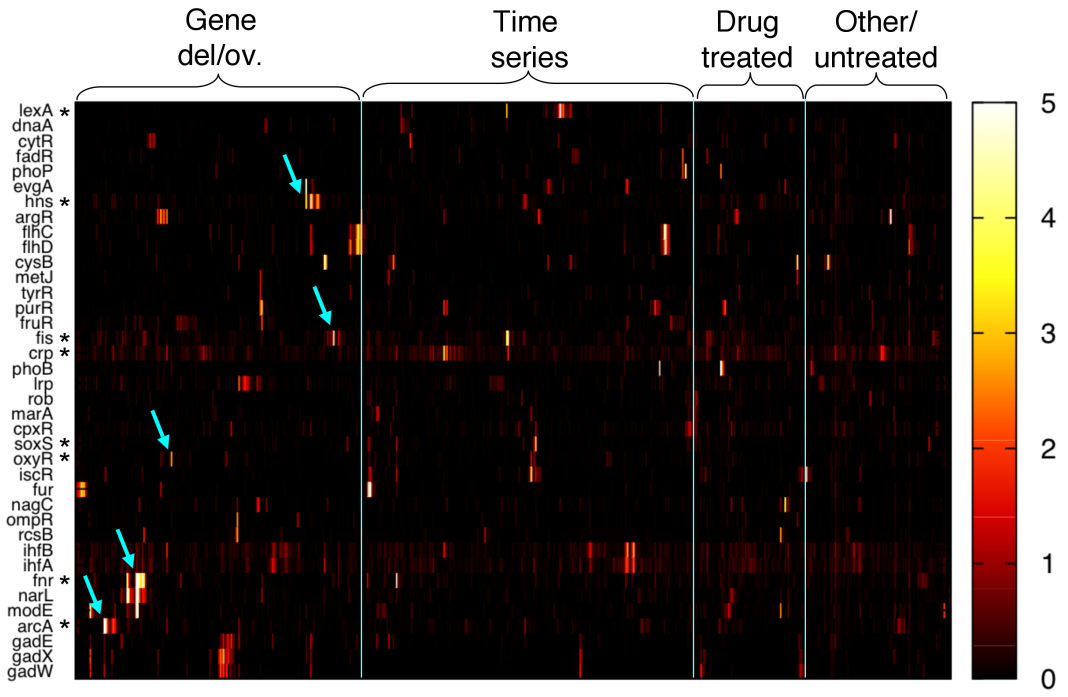

Figure S11: Significance of individual microarrays for network inference
The heatmap depicts the internal node weights of decision trees trained to predict the target genes of *E. coli* transcription factors. A single weight thus represents the importance of an individual microarray for the detection of known target genes for a given transcription factor. If a microarray of a transcription factor deletion mutant is available (transcription factors marked by *), it often receives particularly strong weights (cyan arrows). Even tough other types of experiments have lower weights on average (**Fig. SS10**), they can be similarly informative in some conditions (bright spots scattered through the matrix). Hierarchical clustering was employed to sort rows and columns of the heatmap.

will be some tendency for the method to have an enhanced probability of detecting true interactions at the top of the ranked predictions, such as:

$$p_{\text{pos}}(r) = \frac{1}{T} + \frac{b}{P}(1 - 2\frac{r-1}{T-1}),$$

where we introduce a parameter that accounts for bias, b. Note that if $b = 0$, the probability reduces to that of the random prediction method. There is also a corresponding probability of detecting a non-interacting transcription factor-target gene pair (a *negative*) at rank position r, $p_{\text{neg}}(r)$. Note that, as each ranked position in the prediction list must contain either an interacting or non-interacting transcription factor-target gene pair (a *positive* or a *negative*), the condition $P \cdot p_{\text{pos}}(r) + N \cdot p_{\text{neg}}(r) = 1$ holds, and therefore:

$$p_{\text{neg}}(r) = \frac{1}{T} + \frac{b}{N}(1 - 2\frac{r-1}{T-1}).$$

In order for these probabilities to be positive, b has to be smaller than the smaller of N/T and P/T. **Figure S12b** shows the results of these calculations where $P = 30$, $N = 70$, and $b = 0.2$. It can be shown that the average area under the precision recall curve (AUPR) for a method following these probability laws is 0.41 (for a random prediction we expect AUPR=P/T=0.3).

Let us now explore what happens to the average rank of a true interaction if we integrate the predictions of a community of K inference methods. The average rank assigned to a possible transcription factor-target gene interaction I, over the predictions of the K inference methods, is computed as

$$r_{\text{Borda}}(I) = \frac{1}{K}\sum_{j=1}^{K} r_j(I).$$

As an example, the true interaction A in **Figure S12a** has average rank 8.66, whereas non-interaction Z has average rank 19.67. If a method has better than a random probability to predict true interactions, then the average rank for a true interaction will be different from the average rank for a transcription factor-target gene pair that doesn't interact, given that the former is computed using the average over $p_{\text{pos}}(r)$ and the latter is computed using $p_{\text{neg}}(r)$. We will assume that all the teams in this commu-

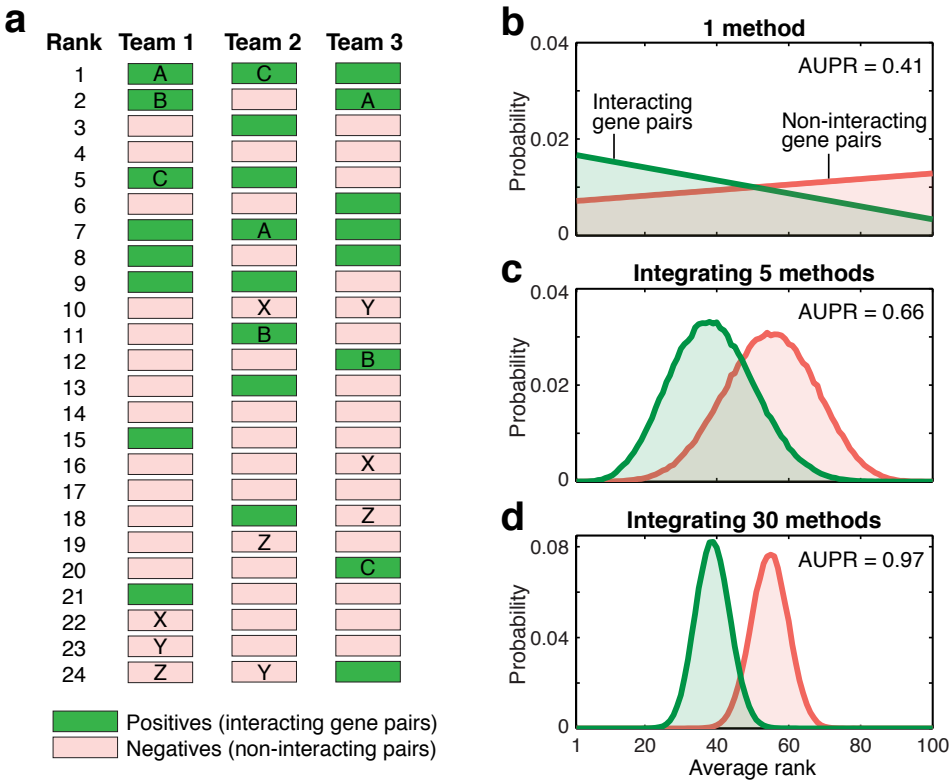

Figure S12: Why community integration can outperform the best individual inference methods (**a**) A hypothetical example of predictions submitted by 3 separate teams. The challenge is to integrate predictions made by each team into a single ranked list. (**b-d**) Two sufficient conditions for integration to outperform individual inference methods are: (1) each of the inference methods must have better than random predictive power (i.e., on average interacting pairs are assigned better ranks than non-interacting pairs); and (2) predictions of different inference methods must be statistically independent.[24] For illustrative purpose, we consider a simple scenario comprising T=100 candidate transcription factor-gene pairs, out of which P=30 interact and N=70 do not interact. For instance, individual inference methods that assign ranks to interacting and non-interacting pairs with the probabilities shown in **Panel b** suffice condition (1). Although interacting pairs are assigned better ranks on average, the probability of incorrect predictions (non-interacting pairs in the top-part or interacting pairs in the bottom-part of the prediction list) is considerable in this example, resulting in an AUPR of only 0.41 (for a random prediction, we expect AUPR=P/T=0.3). (**c**, **d**) If the assigned ranks are independent across methods (condition 2), the central limit theorem establishes that the average rank distribution will approach a Gaussian distribution. Its variance shrinks as more methods are integrated, thereby increasingly segregating interacting from non-interacting transcription factor-gene pairs (Panels **b** $\rightarrow$ **d**). Consequently, the probability that interacting pairs are ranked better than non-interacting pairs increases, resulting in an AUPR that tends to 1 (perfect prediction) as the number of integrated inference methods increases. Even though all predictions are based on the same datasets in the DREAM challenge, and are thus not statistically independent, we have shown that diversity arises due to method-specific capabilities to extract different kinds of information from the data (Fig. 2). Methods from different classes show greater levels of independence and thus contribute more to community performance (Figs. 2b and 3c). Since predictions are partially, but not completely independent, the AUPR increases as more methods are integrated (Fig. 3a) but tends to a value lower than 1 in practice.

nity have the same $p_{\rm pos}(r)$ and $p_{\rm neg}(r)$. The distribution of the average ranks for interacting and non-interacting pair pairs are shown in **Figure S12b-d** for an individual inference method and the integration of 5 and 30 inference methods, respectively. As the number of integrated methods increases, a true interacting pair is more likely to have a better rank than a non-interacting pair. As this is true for any interacting and non-interacting pair, then the interacting segregate from the non-interacting pairs. This feature of integration results in a larger AUPR curve as the number of included methods increases (e.g., AUPR=0.66 for 5 teams and 0.97 for 30 teams). It is clear then that the aggregation of even 5 teams outperforms (according to the AUPR metric) each of the participating members of the aggregate, whose typical AUPR is 0.41.

In order for the integration to outperform individual predictions, the methods being integrated need to be statistically independent, that is, the rank where an interaction is placed by a method has no statistical dependency on the rank where the same interaction is placed by any other method. If this assumption holds, then the central limit theorem of probability theory establishes that, as more predictions are averaged, the average rank distribution will approach a Gaussian distribution whose variance shrinks as the number of integrated teams increases. If the integrated methods have some predictive ability, then the mean ranks of the interacting transcription factor-target gene pairs will be better than the mean ranks of of the non-interacting transcription factor-target gene pairs, and the variances will eventually shrink to make all the interactions have average ranks that cluster tightly around their respective means. In our example it can be shown that more than 95% of the true interactions will have an integrated rank in the interval

$$[\frac{T}{2} - \frac{bT^2}{6P} - \frac{T}{\sqrt{3k}}, \quad \frac{T}{2} - \frac{bT^2}{6P} + \frac{T}{\sqrt{3K}}]\,,$$

whereas more than 95% of the non-interactions will have an integrated rank in the interval

$$[\frac{T}{2} + \frac{bT^2}{6N} - \frac{T}{\sqrt{3k}}, \quad \frac{T}{2} + \frac{bT^2}{6N} + \frac{T}{\sqrt{3K}}]\,.$$

Thus, when K (the number of members in the community) is large enough, the 95% intervals will cease to overlap, and 95% of the positives will be ranked above 95% of the negatives, producing excellent precision and recall. It is clear that as K increases, having a positive in the interval where the negatives concentrate will be extremely unlikely. That is, under the assumption of independent predictions, the area under the precision recall curve (AUPR) tends to 1 as the number of integrated methods increases.

In the real world scenario of the present DREAM5 network inference challenge, predictions from different methods are indeed considerably different (as shown in **Figs. 2b,c** in main text) but can obviously not be fully independent as all predictors use the same input data. Thus, as predictions are partially, but not completely independent, the AUPR increases when more predictors are integrated (see **Fig. 3a** in main text) but tends to a value lower than 1.

6.2 DREAM5 community networks

All 29 DREAM5 submissions were used to construct the community-based transcriptional regulatory networks. For each compendium, integration of the individual team predictions into a single community network was done using the Borda count method described in the previous section. The resulting community-based networks consists of the reordered lists of transcription factor-target gene pairs.

Each team was requested to submit a total of 100,000 predictions. Interactions not listed in the top 100,000 predictions were assigned a rank of 100,001. Upon completion of applying Borda's method, and for each dataset (*in silico*, *E. coli*, *S. cerevisiae*, *S. aureus*), the top 100,000 predictions were selected and called the community predictions (**Supplementary Data 3**).

Weighted voting.

Borda count voting amounts to an *unweighted* rank average over the individual predictions of an interaction. To explore how the community predictions are affected when combining only the best-performing methods and/or giving methods with a better performance a higher weight, we tested several *weighted* voting schemes. We stress that these weighted voting methods are only applied to build an intuition of how the performance of community predictions is affected by good/poor predictors. Weighted voting cannot be used in practice when inferring an unknown regulatory network, as in the case of *S. aureus* here, because the performance of the inference methods is not known.

The *weighted* average rank assigned to a possible transcription factor-target gene interaction I, over the predictions of the K inference methods, is computed as

$$r(I) = \frac{1}{\sum_{j=1}^{K} w_j} \cdot \sum_{j=1}^{K} w_j\, r_j(I)\,,$$

where w_j is a measure of performance of method j (e.g., the AUPR).

To gain a sense of the performance of *unweighted* (Borda count) and *weighted* community predictions, we systemat-

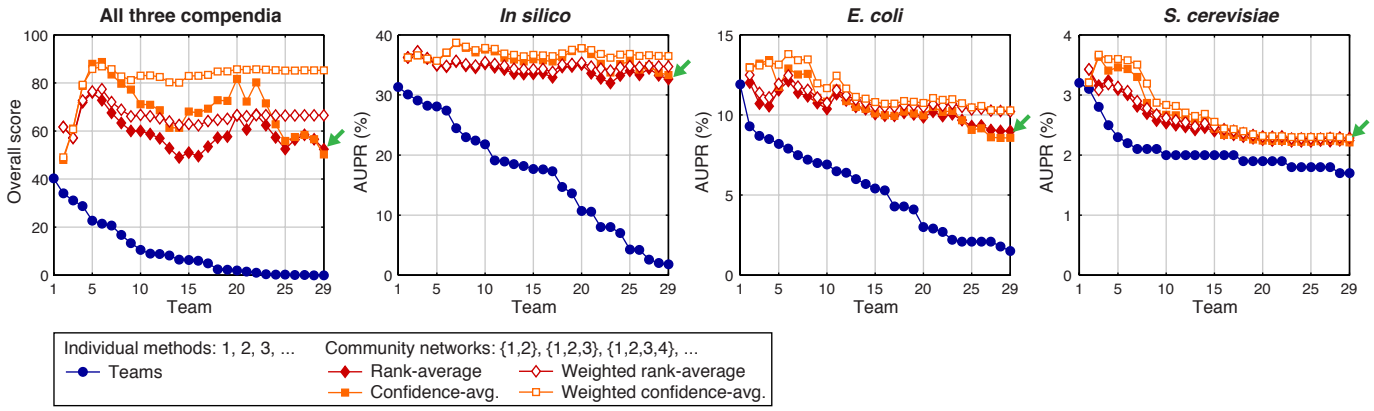

Figure S13: Community integration using unweighted and weighted voting
Community predictions were obtained by combining the two best teams, the three best teams, the four best teams, etc, using either unweighted (Borda count) or weighted voting. In addition to voting based on the *edge ranks* (*diamonds*), we also tested voting based on the *edge confidence values* (third column in the prediction format, **Supplementary Note 1**) assigned by the inference methods (*squares*). In accordance with Figure 3d of the main text, even unweighted voting is robust to inclusion of poor predictors. Although weighted voting based on edge confidence values performed best overall, the difference with the other approaches is relatively small on the three individual compendia. We stress that only the *unweighted* voting that combines *all methods at hand* (rightmost points marked with arrows) is truly unsupervised, *i.e.*, can be applied when inferring an unknown regulatory network, where the performance of the individual methods is not known *a priori*.

ically formed communities composed of the top two methods, the top three methods, the top four methods, *etc.*, until the last community, which contains all 29 methods applied by the participants of the challenge (**Fig. SS13**). This analysis confirms an observation made in the main text (**Figure 3d**): adding poor predictions hardly degrades the consensus of the more accurate predictions, even when using *unweighted* voting. Although *weighted* voting improves the performance overall, the difference is rather small on the individual compendia. *Therefore, and since the performance of inference methods is difficult to estimate when inferring an unknown regulatory network, integrating all inference methods at hand using unweighted voting seems to be a good choice.*

7 E. coli and S. aureus community networks

7.1 Network construction

Community predictions for *E. coli* and *S. aureus* were obtained using the Borda count method as described in the previous section. Note that these community predictions are *weighted* networks, as they assign a measure of confidence (the average rank) to edges. To obtain an *unweighted* network that classifies edges simply as present or absent, a confidence threshold must be chosen. Edges above the threshold are considered present, and those below absent. Choosing a threshold amounts to a trade-off between sensitivity and specificity. For the analysis of the *E. coli* and *S. aureus* community networks in the main text, we chose a cutoff of 1,688 edges, which corresponds to an estimated precision of 50% for *E. coli*. Precision was estimated based on the RegulonDB gold standard of interactions described in **Supplementary Note 3**.

Note that many genes and (potential) transcription factors still have no experimentally supported interactions in RegulonDB, *i.e.*, they are not part of our gold standard. Edges involving genes or transcription factors that are not part of the gold standard were ignored when computing the precision (*i.e.*, they were neither counted as true nor as false positives, instead they were simply excluded from the calculation). For example, at the cutoff of 1,688 edges, only 200 edges are part of the gold standard. Of these 200 edges, 100 were true positives and 100 were false positives, resulting in the estimated precision of 50%. Note that this is a conservative estimate, because some of the novel edges that are considered false positives may in fact be newly discovered regulatory interactions that are currently missing in RegulonDB, as our independent experimental validation of such novel interactions shows (**Figure 4** of the main text).

As discussed, there is no gold standard set of interactions that exist for *S. aureus*, which presents the problem of not being able to *directly* ascribe a precision with a given nework size. In this case, we make the assumption that the *S. aureus* network predictions perform similarly to the *E.*

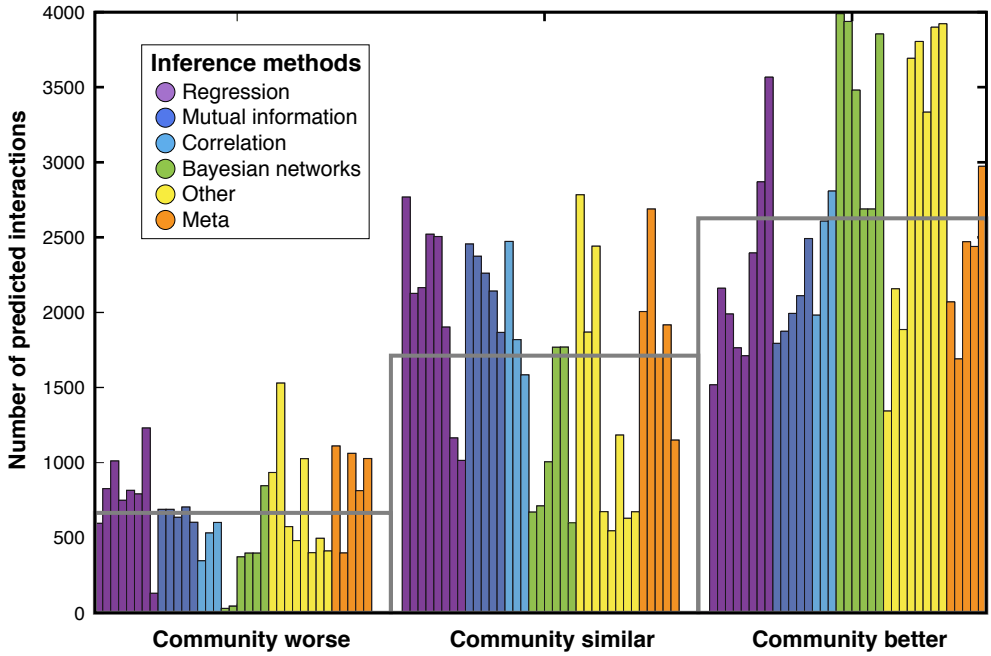

Figure S14: Rank improvement of individual edges through community integration
The plot shows the number of true interactions that were predicted worse, similarly, or better by the integrated community network than by the individual inference methods (the average over all methods is shown in grey). Interactions were counted as predicted better or worse than the individual methods if the difference $rank(integrated) - rank(method_i)$ exceeds 10,000 or -10,000 ranks, respectively. Otherwise, the predictions were considered similar. The majority of true interactions were ranked better (by a large margin of over 10,000 ranks) in the community network than even in the best individual predictions. Note that the ordering of methods is the same as in the main document (Table 1 and Figure 2).

coli community network. Under this assumption, we use the network size of 1,688 edges since this is the network size derived from the *E. coli* network at an estimated 50% precision.

7.2 S. aureus network evaluation using RegPrecise

RegPrecise is a database of transcriptional regulatory interactions in prokaryotes that have been inferred using manually reviewed, homology based methods.[63] For the DREAM challenge evaluations, we required all gold standard interactions to be experimentally supported and did not consider electronically inferred interactions. As the interactions in RegPrecise for *S. aureus* are largely electronically inferred, we did not consider RegPrecise as a gold standard for the overall evaluation. Nonetheless, the RegPrecise interactions represent a rich set of information that we have used to test our assumption that the 50% precision threshold derived in *E. coli* can be used as a proxy for performance in *S. aureus*.

RegPrecise contains 517 interactions comprised of 38 transcription factors and 446 target genes that match up with the genes represented in the microarray compendium supplied in the DREAM5 network inference challenge. All individual methods and the community network performance were evaluated using AUPR and AUROC (**Supplementary Note 4.1**). We performed the same analysis using RegPrecise interactions and report the performance in S15. Using the AUPR, we find that the *S. aureus* community network ranks 3^{rd}.

The 50% precision reported for *E. coli* correponds to 1,688 interactions. We identified 50% precision in the *S. aureus* community network from the RegPrecise analysis and this threshold corresponds to a network with 988 interactions. For the *E. coli* gold standard interactions reported in RegulonDB, there were 2,066 experimentally supported interactions comprised of 144 transcription factors and 999 target genes. Considering that the *E. coli* gold standard is 4 times the size of RegPrecise and only experimentally supported interactions from RegulonDB were used, we feel that the RegPrecise interactions understimate the number of true positive relationships. Therefore, we report the *S. aureus* community network using the 50% threshold in *E. coli* and perform further analyses on this network of 1,688 edges. For completeness, we provide the *S. aureus* community network using the RegPrecise 50% precision threshold

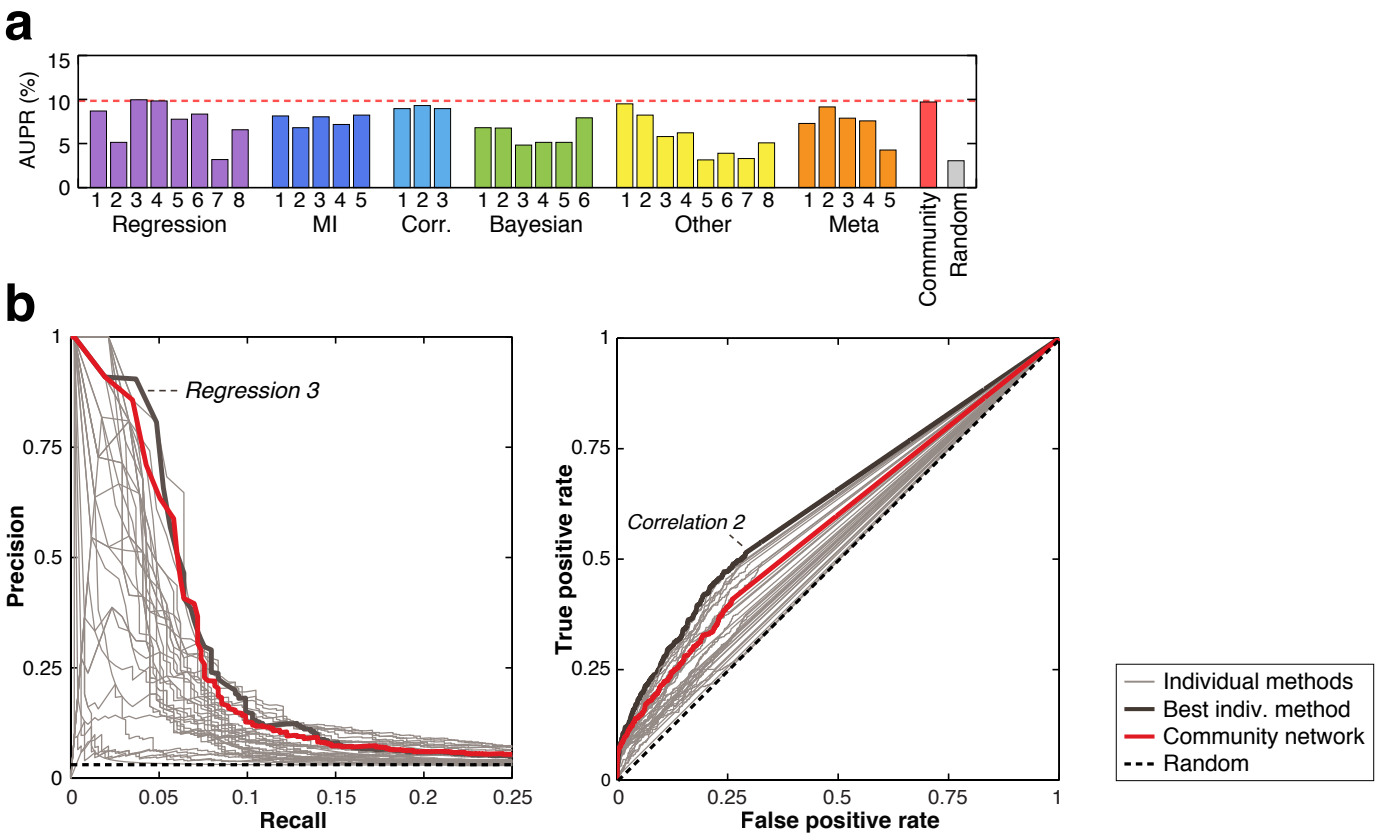

Figure S15: Performance evaluation for *S. aureus* using the RegPrecise database
(**a**)The Area Under the Precision Recall (AUPR) for all network inference algorithms (including the community integration) evaluated for *S. aureus* using the RegPrecise database as the "gold standard." (**b**) The precision/recall (PR) and receiver operating characteristic (ROC) are show with the community network highlighted in red and the top performing algorithm in black. The remaining DREAM participant sumbissions and the off-the-shelf predictions are shown in grey.

(988 edges) in **Supplementary Data 5**.

7.3 Analysis of network modules

Module detection.

We identified network modules, *i.e.*, groups of transcription factors and genes that are more densely connected among themselves than expected in a randomized network with the same degree distribution, using Newman's spectral method (including the greedy optimization step after the spectral decomposition).[62] We found that both the *E. coli* and *S. aureus* community networks are highly modular, as shown in **Figures 4a** and **b** of the main text for the two community networks at the 50% precision cutoff (1,688 edges).

Gene Ontology (GO) term enrichment.

GO term enrichment analysis was performed on each of the identified network modules for both the *E. coli* and *S. aureus* networks. GO term gene annotations were downloaded from the Gene Ontology website[m] for *E. coli*. For *S. aureus*, gene annotations were taken from the Affymetrix annotation files.[n] GO terms under the *biological process* branch of the ontology were used. Genes were mapped directly to the ontology and propagated to the root node. This process ensures that all parent GO terms recursively inherit the annotations of their child terms. GO terms annotated with less than 3 genes and GO terms annotated with greater than 500 genes were removed. The remaining GO terms were used as input to the GO term enrichment calculation.

Each identified network module can be represented by a set of genes. For each module, all associated GO term annotations were tested by counting the number of occurrences of the term in comparison to the number of occurrences of the term in the entire network. Statistical significance for each GO term was assessed using the hy-

[m] www.geneontology.org/GO.downloads.annotations.shtml
[n] www.affymetrix.com/support/support_result.affx

pergeometric distribution. Estimated p-values were then multiple hypothesis corrected using the q-value calculation.[82]

We found that the network modules—which were identified solely based on network connectivity—are also strongly enriched for specific biological processes, *i.e.*, network modules coincide with functional modules (**Fig. SS16** and **Fig. SS17** show module enrichments for the networks at the 50% precision cutoff, similar results were obtained at different cutoffs). The genes assigned to each module, as well as the p-values and q-values for the GO terms, are supplied in **Supplementary Data 6**.

8 Experimental validation

All cultures were grown in 1 mL of indicated growth medium in 14 mL Falcon tubes. Incubation of all cultures was performed in darkened shakers (300 RPM) at 37 °C. Experimental conditions for growth and transcription factor induction were all designed to reproduce the conditions under which the induction of these transcription factors have previously been studied. The wild-type *E. coli* strain used was BW25113. Knockout strains were constructed via transduction from the KEIO knockout library.[6]

8.1 Transcription factor selection

Transcription factors were selected from the *E. coli* community network with 50% predicted precision or greater. Selected transcription factors had at least 8 predicted, but untested target genes. The conditions under which these transcription factors are active are known and can be replicated during aerobic growth under laboratory conditions.

Targets were chosen as follows. For each transcription factor, if confirmed target genes were available in the data set, 1-2 of these geness were chosen as positive controls. All unconfirmed target genes were used in qPCR unless suitable primers could not be obtained for the sequence (see **Supplementary Note 8.2** for primer design specifications). Where multiple targets were encoded within a single operon, only the first gene in the operon was used.

rhaR is the transcriptional activator of the rhamnose utilization operon. Native production of rhamnose activated genes is extremely low in the absence of a chemical inducer, and induction by rhamnose is slow, requiring 40-50 minutes to reach steady state.[26] Overnight LB cultures of both wild-type and ΔrhaR *E. coli* were inoculated 1:500 in minimal salt media (M9) + 0.2% casamino acids + 50 μM thiamine + 0.4% glycerol. Cultures were grown to early exponential phase (A600 ≈ 0.2, 3.5 hours) before addition of 0.2% (w/v) L-rhamnose[26] rhamnose and were incubated 45 minutes before stabilization for RNA extraction. RNA was also extracted from untreated samples of wild-type and ΔrhaR grown under the same conditions but without the addition of rhamnose. As rhaR is cotranscribed with its confirmed target rhaS, to avoid any problems with qPCR due to possible scarring at the C-terminal end of the rhaS transcript, the rhamnose transporter gene rhaT was used as a positive control.

purR is active under conditions of purine nucleotide deficiency. Overnight LB cultures of wild-type and ΔpurR *E. coli* were inoculated 1:500 in minimal salt media (M9) + 0.2% casamino acids + 6.6 μM thiamine + 0.4% glucose, with 100 μg/mL adenine added to activate purR and repress purine nucleotide biosynthesis.[18,73] Cultures were grown to exponential phase (A600 ≈ 0.2, 3.5 hours) with or without adenine, as previously described in,[18,36] before stabilization for RNA extraction. RNA was also extracted from untreated samples of wild-type and ΔpurR grown under the same conditions but without the addition of adenine.

gadE is a central transcriptional activator of the principal acid resistance system.[38] Overnight LB cultures of wild-type and ΔgadE *E. coli* were inoculated 1:500 in minimal salt media (M9) + 0.2% casamino acids + 0.4% glucose + 30 μM thiamine (pH 7). Cultures were grown to exponential phase (A600 ≈ 0.2, 3.5 hours) before adjustment of pH to 5.4-5.7 by addition of 45 μL 1M HCl. Cultures were incubated for an additional 2 hours before stabilization for RNA extraction.[16] RNA was also extracted from untreated samples of wild-type and ΔgadE grown under the same conditions but without the pH adjustment.

mprA (emrR) is known to respond to toxic molecules such as salicyclic acid (5 mM), 2,4-dinitrophenol (DNP, 0.5 mM), carbonyl cyanide m-chlorophenylhydrazone (CCCP, 10 μM), and carbonyl cyanide p-(trifluoromethoxy)phenylhydrazone (FCCP).[48] This transcription factor has been shown to interact directly with DNP, CCCP and FCCP, which reduces the capacity of mprA to bind to DNA.[15,89] Overnight LB cultures of wild-type and ΔmprA *E. coli* inoculated 1:500 in LB and allowed to grow to exponential phase (A600 ≈ 0.3, 2.5 hours) before addition of the trancriptional inducer, 10 μM CCCP. Cultures were incubated 30 minutes before stabilization for RNA extraction. RNA was also extracted from untreated samples of wild-type and ΔmprA grown under the same conditions but without the addition of CCCP.

cueR is a metal-binding transcription factor and the regulator of the primary copper homeostasis system in *E. coli*.[83,90] Overnight LB cultures of wild-type and ΔcueR *E. coli* were inoculated 1:500 in minimal salt media (M9) + 0.2% casamino acids + 0.4% glucose and grown to ex-

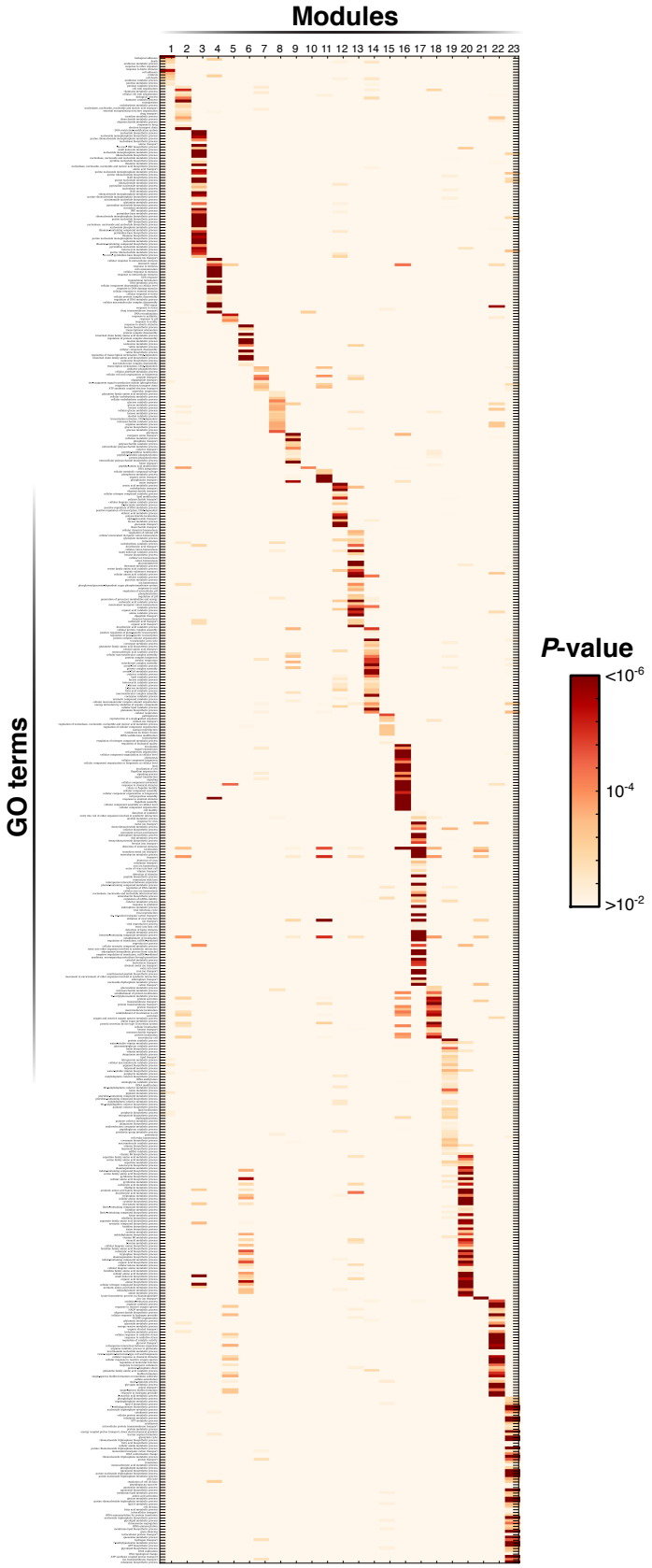

Figure S16: Functional enrichment of network modules in E. coli
Network modules are strongly enriched for very specific biological processes, with only few processes being enriched across more than one module. Only network modules that comprise at least five genes are shown.

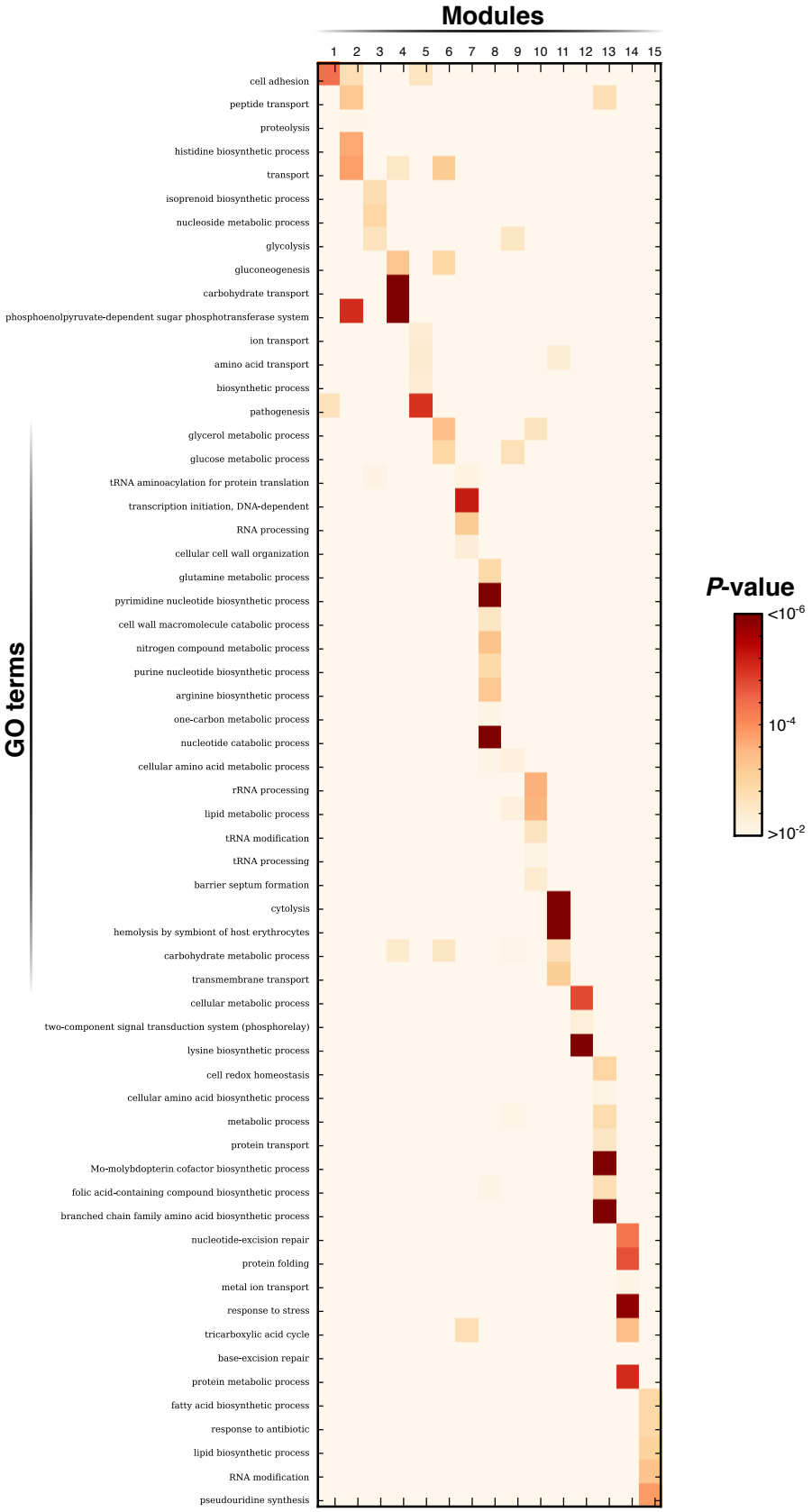

Figure S17: Functional enrichment of network modules in S. aureus

Network modules are strongly enriched for very specific biological processes, with only few processes being enriched across more than one module. Only network modules that comprise at least five genes are shown.

ponential phase (A600 ≈ 0.2, 3.5 hours) before addition of $CuSO_4$ (500 μM). Cultures were incubated 15 minutes before stabilization for RNA extraction. RNA was also extracted from untreated samples of wild-type and ΔcueR grown under the same conditions but without the addition of $CuSO_4$.

8.2 RNA Extraction and qPCR

Cultures were stabilized with RNAprotect Bacteria Reagent (Qiagen) according to the manufacturerâĂŹs protocol, and resulting cell pellets were stored overnight at −80 °C. RNA extraction was performed using RNeasy Mini Kit (Qiagen) and DNA contamination was eliminated using TURBO DNA-Free (Ambion, Austen, TX) according to the manufacturersâĂŹ protocols. Sample concentration was estimated using the ND-1000 NanoDrop spectrophotometer. Standard PCR using Taq polymerase and qPCR primers was used to test for DNA contamination in RNA samples after TURBO DNA-Free digestion (4% RNA suspension by volume, 30-35 cycles). RNA was stored at −80 °C.

cDNA for qPCR was synthesized from RNA using the Superscript III First Strand Synthesis kit (Invitrogen) and stored at −20 °C. rrsC and rrsH were used as endogenous standards. Quantitative PCR primers for each transcript of interest and the reference transcripts were designed based on the NCBI *E. coli* K12 MG1655 genomic sequence (Refseq NC_000913)° using Primer3Plus software[86] (Table S1). Primers were designed under the following constraints: amplicon size was 100-120 bp, the calculated primer melting temperature was 55 °C, GC content was 45-55%, and probabilities of primer-dimer/ hairpin formations were minimized. Primer specificity was confirmed via standard Taq polymerase PCR on genomic DNA.

qPCR reactions were prepared manually using the LightCycler 480 SYBR Green I Master Kit (Roche Applied Science) according to the manufacturer's instructions. qPCR reactions were performed with a total volume of 20 μL, containing 0.5 μM of forward primer and 0.5 μM of reverse primer and 10 μL 2× 480 SYBR Green Master Mix. Reactions were carried out in white LightCycler 480 96-well plates (Roche). One negative control (replacing cDNA with PCR H_2O) was performed for each primer set in all qPCR runs. PCR parameters were: denaturation (95 °C for 10 minutes), 30-35 cycles of three-segment amplification (95 °C for 10 seconds, 50 °C for 10 seconds, 72 °C for 10 seconds). The thermal cycling program was concluded with a dissociation curve (65 °C ramped to 95 °C, 10 seconds at each 1 °C interval) to detect non-specific amplification or primer-dimer formation. Cp values were obtained from the Roche LightCycler 480 software set with default parameters for basic relative quantification analysis. Results were recorded as relative fold change expression after normalization using reference gene expression as detailed in.[65] qPCR was performed using 3 biological replicates; technical duplicates were performed on separate days to assess variability. Full results can be found in **Supplementary Data 7** and a summary of the qPCR fold changes is shown in **Figure SS18**.

°www.ncbi.nlm.nih.gov/RefSeq

9 Methodological insights

In the evaluation across *E. coli*, *S. cerevisiae*, and *in silico* datasets, 10 teams and 2 off-the-shelf methods achieved an overall score of more than 10 and could thus be regarded as above average. Three of these successful teams used meta analyses (Meta 1-3, compare **Supplementary Note 10**) combining mutual information, z-scores, and correlation. The two off-the-shelf methods in this selection tested for mutual information in order to detect true gene regulatory interactions (Mutual Information 1 and 2). Lasso performs quite well (5 teams, Regression 1-5) if a proper stability selection approach (e.g. bootstrapping) is used. The remaining two teams (highest overall scores) applied novel, specifically designed network inference approaches based on random forests and ANOVA, respectively (Other 1 and 2, compare **Supplementary Note 10**). Other techniques that were applied by several different teams included Bayesian networks, correlation, and mutual information.

Although some base methodologies were applied frequently, extensive team specific modifications were implemented with significant effects on the resulting predictions (comparisons between teams) and team ranking. Data resampling techniques such as bootstrapping were of key importance for method performance. Lasso, for instance, due to its low computational complexity, is well suited for resampling techniques. Here, prediction improvements justified the added cost for repeated sampling. Five out of the ten most successful teams according to the overall score applied Lasso with resampling with the overall ranks of 3, 4, 6, 7 and 10. Two approaches of the regression category neglected resampling (Regression 7 and 8) and consequently exhibited a significantly lower performance (overall tranks of 26 and 32 out of 35 approaches). Lasso utilizes the coordinate descent procedure[30] that identifies a topology that is optimal with respect to a penalty parameter. Depending on this penalty parameter, Lasso preferentially selects regulators that independently contribute to the expression control of target genes. Bayesian networks are another class of network inference approaches that enforce an analogous conditional indepence assumption. However, Bayesian networks were not among the top ten

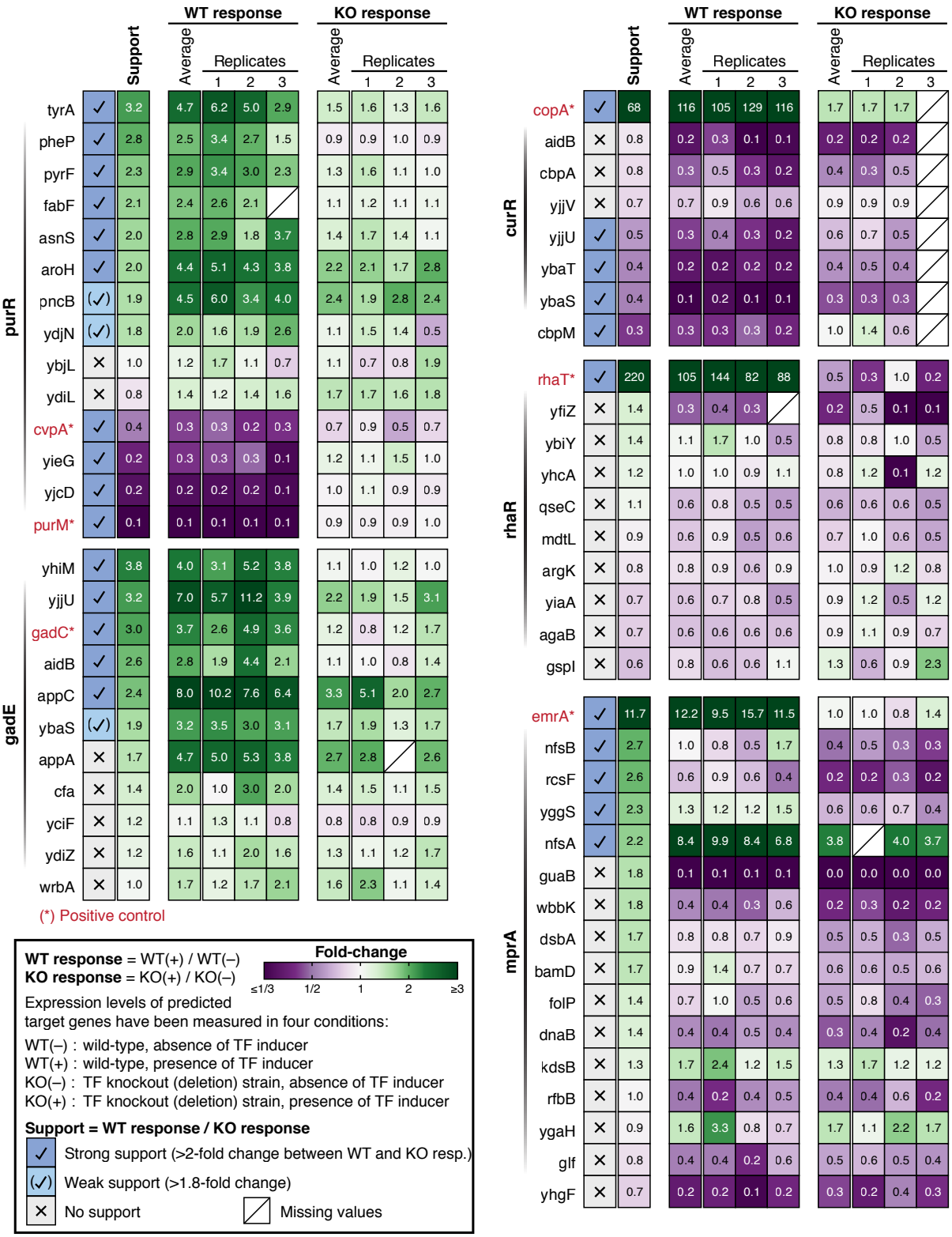

Figure S18: Experimental support for newly predicted interactions

From the *E. coli* community network, 53 *novel* predictions for 5 transcription factora (TFs) were tested experimentally (+6 positive controls, i.e., *known* targets from RegulonDB that were recovered by the community). For each predicted target gene, the expression levels were measured through qPCR in the presence and absence of TF inducer, both in wild type (**WT response**) and TF deleted strains (**KO response**) (three biological replicates for each condition). 20 predicted targets with a clear difference between WT and KO response to TF induction (fold-change>2) were considered *strongly supported*. 3 predicted targets that are borderline (fold-change>1.8) were considered *weakly supported*.

teams although several of them performed repeated sampling. This difference in performance might be explained by the different search strategies employed by Bayesian networks and Lasso. In contrast to the coordinate descent procedure used by Lasso, heuristic searches utilized by Bayesian networks cannot guarantee that an optimal topology will be detected.

Cluster 4 in the PCA plot (compare **Supplementary Note 4** and **Figure 2c** in main text) captures methods from the categories *Meta* and *Other*. In addition, this cluster includes unusual method variants such as *Mutual Information 1* (CLR) which uses a special scoring scheme that rescores a given $TF_x - TG_y$ edge to reflect TG_y and TF_x in the distribution of all target genes and all transcription factors, respectively. Similarly, *Bayesian 6* is the only method using dynamic Bayesian networks, *Regression 2* is the only method using group Lasso,[91] and *Regression 7*, which is the only participating Lasso implementation neglecting stability selection. The only meta-method not found in cluster 4 is *Meta 6*, which combines correlation with mutual information and accordingly falls into the correlation/mutual information cluster.

The analysis of network motifs (**Figure 2c** in main text) revealed that different approaches performed best at capturing different regulatory relationships. The motif bias patterns of methods based on Bayesian networks and Lasso were very similar. Both classes of approaches preferentially select transcription factors as regulators that independently contribute to the expression control of a target gene. Such an independence assumption does not hold for genes regulated by mutually dependent transcription factors, which are therefore difficult to detect by these approaches. Thus, Bayesian networks as well as Lasso preferentially predict cascade motifs, *i.e.*, they miss the additional edges that are part of feedforward loops. In turn, target genes regulated by linear transcription factor pathways are harder to detect by mutual information and correlation based approaches. Analogous to Bayesian networks and Lasso, correlation and mutual information-based methods also exhibit similar bias patterns in our network motif analysis. The direct comparison between these two motifs showed that methods seem to be merely able to shift preferences. Whereas lasso based methods might be able to resolve more cascade motifs correctly (in comparison to mutual information-based methods) they are prone to detect fewer of the existing feed-forward loops.

The strongest motif specificities are exhibited by methods that specifically take transcription factor deletion experiments into account. Obviously, the deletion of a transcription factor can be exploited to infer the downstream targets of this transcription factor. Exploiting transcription factor deletion experiments also markedly improves inference of high-outdegree transcription factors and high-indegree target genes. This is partly due to the fact that transcription factor deletions were carried out preferentially for transcription factors that regulate many genes. These experiments also help to orient the direction of interactions where both interaction partners are transcription factors. Lasso based techniques are the only other class of approaches that are (albeit to a much lesser degree) able to distinguish edge orientation. On the other hand, all approaches that used transcription factor deletions had more pronounced problems (in comparison to Lasso) avoiding the prediction of false indirect effects in cascade motifs. In a cascade motif, there is an indirect regulation $A \rightarrow B \rightarrow C$ but no direct regulation $A \rightarrow C$. If a measurement of the deletion of A is available, it frequently shows effects of a similar kind both on B and C. However, most methods specifically using transcription factor deletions do not distinguish well between direct and indirect effects and thus wrongly predict a false positive $A \rightarrow C$ interaction.

Apart from the restrictions on cascade motifs, the above observations suggest that knockouts are particularly informative for network reconstruction. We confirmed this conclusion using a machine learning framework that allowed us to evaluate the information content of different experiment types (see **Supplementary Note 5.2**). It is thus surprising that many established inference approaches neglect to exploit this information.

In contrast to Lasso that performed better on the artificial datasets, mutual information and meta-methods worked better on experimental data. The good performance of the meta methods contributes to the overall message of the present paper, *i.e.*, that combinations of different approaches frequently perform better than the underlying individual methods. We also found that — especially on the experimental datasets — Pearson correlation performs comparable to mutual information and Spearman correlation: capturing non-linear relationships between transcription factor and target gene expression proved of limited advantage.

10 Network inference methods

All participating teams of the challenge were required to submit a detailed description of their network inference methods. The method descriptions of 15 out of 29 teams are included in the following sections. The remaining teams chose not to disclose a detailed description of their method, an option that we allowed to encourage application of yet unpublished inference methods (see **Table 1** for short descriptions of these methods).

In addition, we tested six commonly-used off-the-shelf algorithms on the DREAM5 network inference challenge

data to provide readers with a point of reference regarding algorithm performance. These algorithms were run by the organizers of the challenge using default parameters to best simulate a user running the network inference method "naïvely" on the data, without optimizing or tuning its parameters. Note that only the independent submissions by the different challenge participants were used to form the integrated community networks — the 6 off-the-shelf methods were *not* included. The off-the-shelf methods are distinguished from the methods applied by the challenge participants by a "*" in their section heading below.

10.1 Regression 1 – Inferring gene regulatory networks with the stabilized Lasso

This approach to gene regulatory network (GRN) inference is based on the assumption that the expression level of the transcription factors that directly regulate a target gene are the most informative, among all transcription factors, to predict the expression level of the target gene. We therefore reformulate the problem as a feature selection problem: for each given target gene, which subset of transcription factor allows to best predict the expression of the target gene across experiments? To solve this feature selection problem, we use a Lasso sparse regression approach[84] combined with stability selection[59] to score the candidate features. We then rank all candidate transcription factor-target gene regulations by decreasing stability score to produce the final GRN prediction. This method will be described in more detail in.[35]

Data preprocessing. We started from the $N \times P$ raw gene expression matrix, which provides the expression level of the N genes in P experimental conditions. We wanted to investigate the performance of this approach without any refinement based on biological assumption, and therefore followed a completely "agnostic" approach where we discarded any information about the genes and the experimental conditions. As preprocessing, we simply centered and scaled to unit variance the expression levels of each gene, a standard procedure in the regression setting.

Splitting the GRN problem into many feature selection subproblems. In order to infer the global GRN, we treated each target gene in turn and tried to predict its direct regulators among all transcription factors. We therefore created N subproblems, where each subproblem focuses on a particular target gene. For a given target gene, we extracted the vector of expression of the target gene across the P experiments, and similarly extracted the matrix of expression of all candidate regulators across the P experiments. Note that if the target gene is not a transcription factor, then all transcription factors are candidate regulators. If the target gene is a transcription factor, then we considered all other transcription factors as candidate regulators, since this method cannot predict self-regulations. We then formalized the subproblem as a feature selection problem: which transcription factor among the candidate regulators are sufficient to explain the expression variations of the target gene?

Feature selection with Lasso regression. We solve the feature selection problem with the Lasso procedure.[84] In short, we estimated a linear model to predict the expression of the target gene from the expression of the candidate regulators. The Lasso procedure led to a sparse linear model, *i.e.*, to a linear model based only on a few transcription factors. The transcription factors selected by Lasso are therefore good candidates to regulate the target gene. We used the LARS algorithm[25] to solve the Lasso regression.

Stability selection. The direct use of Lasso to select transcription factors has two shortcomings. First, it is known to be an unstable procedure in terms of selected features. Second, it provides no confidence score for the selected features, which makes it difficult to aggregate the transcription factor selected for different target gene in a unique final list. We therefore performed a procedure known as stability selection[59] in order to overcome both issues. In short, for each subproblem, we generated many Lasso regression sub-subproblems by randomly varying the experiments and the candidate transcription factor used in the regression. The final score of a transcription factor-target gene is the number of times it was selected in the top 5 transcription factors by the Lasso in the corresponding sub-subproblems. We then ranked all transcription factor-target gene pairs by decreasing score.

Discussion. We proposed a fully automatic procedure to infer regulations, based on well-established procedures for feature selection. We discarded any biological information about the genes and experiments, such as the presence of replicates, time series or knock-down experiments, to assess the performance of a fully "agnostic" procedure. The method was among the top performers in the *in silico* challenge, and was less accurate on the real microarray data (see Results).

The idea to formulate a GRN as a series of feature selection problems has been proposed before, e.g., by the GENIE3 method[40] (also see **Supplementary Note 10.13**). Interestingly, the combination of Lasso with feature selection bears similarity with the GENIE3 method, which is based on randomized decision trees. The fact that both methods performed well in DREAM5 suggests that the principle of feature selection by randomized stability selection may be a powerful approach for GRN inference.

10.2 Regression 2 – Network inference with regularized linear regression methods

This submission to the DREAM5 network inference challenge is an application of linear regression methods. For each network, we split up the problem gene-by-gene and estimate the set of directed edges that point to a particular target gene by fitting two paired regression models. First, for steady-state (*i.e.*, non-time-series) data, the expression of target gene g is modeled as a linear function of the expression levels of all transcription factors, excluding g if it is a transcription factor. Second, for time-series data, the rate of change in the expression of g is modeled as a linear function of the expression levels of all transcription factors and of g itself to account for decay of gene product. Experiments with knock-out/deletion of g are omitted for the regressions. While rare, these experiments provide information that helps distinguish an edge pointing from g to h rather than in the opposite direction from h to g.[64]

Expecting shared patterns of active transcription factors in the two regressions, we combine them into a single model, which we fit using the group Lasso.[91] This regularization technique ensures that the (sparse) sets of transcription factors inferred for the two data types are the same. Finally, we apply bootstrapping to produce confidence levels, which serve as the basis for the submitted ranked list of edges.

Data preprocessing. We averaged all expression levels over technical replicates. For later reference, write n_1 for the number of averaged steady-state data points, and n_2 for the number of averaged time-series data points (one for each time point that has experimental conditions equal to those of the immediately following time point). All subsequent calculations with these data used weighting to account for the fact that different numbers of technical replicates were averaged.

Combining time-series and steady-state models. The steady-state data was modeled with a linear regression in which the expression level of target gene g was the response and the expression levels of the transcription factors were the predictors. The linear regression model for the time-series data had as a response the change in expression of g divided by the amount of time between the current and the subsequent time points. In this regression, the current expression levels of the transcription factors and of g are the predictors.

We combined the two models into a single regression model with $n_1 + n_2$ data points. We allowed for different variances between the two data parts but, for computational ease, pre-estimated and thereafter treated as known the ratio of the two variances. To estimate the ratio, we computed a sample variance for the steady-state data, and for the time-series data, we estimated the variance in a linear regression that included an intercept and an auto-regressive term (expression of target gene g at the current time point).

Applying the group Lasso. For a fixed target gene, g, let β_j and γ_j be the regression coefficients for transcription factor j in the steady-state and the time-series part, respectively. We computed estimates of these coefficients by minimizing the sum of the (weighted) residual sum of squares and a group Lasso regularization term that promotes sparsity by penalizing nonzero regression coefficients. The regularization term is the product of a tuning parameter λ and a sum over all involved transcription factors, with the j^{th} transcription factor contributing the square root of $\beta_j^2 + \gamma_j^2$ to the sum. Neither the intercepts nor the auto-regressive coefficient are penalized.

With this group Lasso penalty, the estimates of β_j and γ_j are either both zero or both nonzero. Hence, the pair (β_j, γ_j) represents a single transcription factor across both steady-state and time-series data. The pair being nonzero corresponds to an edge from the j^{th} transcription factor to the target gene. We remark that each coefficient pair is associated with two orthogonal predictors, which allowed for fast minimization of the objective function.[29,70] In the group Lasso approach, larger values of the penalty parameter λ encourage greater sparsity of the inferred network. The inferences were based on six values of λ (as fixed fractions of the smallest value that yields zero estimates of all β_j and γ_j).

Bootstrapping. For each value of λ, we resampled 200 datasets by drawing, with replacement, a sample of size n_1 from the averaged steady-state data and a sample of size n_2 from the averaged time-series data. We then applied the group Lasso procedure to each combined data set and determined the fraction of times each transcription factor appeared in the model for the considered target gene during the bootstrap resampling. These fractions were then averaged over the different values of λ to produce the submitted "confidence levels."

Discussion. This method was one of the best-performing methods for the *in silico* network, but did not perform as well on the two *in vivo* networks. While correctly inferring a higher level of sparsity in the *in vivo* networks, the method yielded fewer nodes with high in-degree and more nodes with high out-degree than there are in the *in vivo* gold standards. The performance on the *in vivo* networks may have been hurt by any one of the modeling assumptions, including linearity, as well as the limited preprocessing of the data.

10.3 Regression 3 – Sparse piecewise linear regression based on changepoint processes

The idea behind this approach is to apply a Bayesian piecewise linear regression model based on a multiple changepoint process to all genes, where the explanatory variables are taken from the set of candidate regulatory genes provided by the DREAM5 network inference challenge. The multiple changepoint process acts on a linear ordering of the chips, resulting from a preprocessing step that takes exogenous information about experimental conditions and perturbations into consideration. Inference is based on reversible jump Markov chain Monte Carlo (RJMCMC). Owing to the high computational costs, a preprocessing step based on L_1-regularized linear regression (Lasso) is included. The individual steps are discussed in more detail below.

Data preprocessing. All gene profiles were standardized to zero mean and unit variance.

Chip ordering. The multiple changepoint processes (described below) are based on a chip ordering. This ordering is based on experimental conditions and perturbations, and has been obtained as follows: (1) Chips belonging to the same experimental conditions and perturbations were grouped together, and an average expression profile for that group was computed. (2) The first principal component in the space of average expression profiles was computed and used to initialize a one-dimensional self-organizing map (SOM).[p] (3) The SOM learning algorithm was applied to obtain an ordering of groups. (4) Within each group, chips were ordered by the same process: Initialization of a one-dimensional SOM with the first principal component, followed by the application of the SOM learning algorithm. (5) The final chip ordering was given by the ordering thus obtained, subject to a manual correction to enforce the natural ordering of time series and perturbation dilution series as a rigid constraint.

Active interventions. Active interventions, like gene knockouts, were dealt with in the regression model by presenting inferred values only for the explanatory variables, but removing them for the target variable.

Gene filtering. Due to the high computational costs of the RJMCMC simulations, we applied a filtering step based on the modified Lasso approach proposed by Ahmed and Xing,[2] which is implemented in the software package TESLA. This method can be regarded as a piecewise linear sparse regression approach that is based on L_1-regularization with group-specific regression parameter vectors. In addition to the standard L_1-norm penalty term an additional L_1-norm regularization term was introduced, which penalizes deviations between vectors of regression parameters associated with different groups. The groups correspond to different experimental conditions and perturbations, as described above. The regularization constants were set so as to optimize the BIC score. We modified the original TESLA code to switch from logistic to linear regression. However, for lack of time we did not reprogram the function for computing the likelihood, and hence computed the BIC score with the original code based on logistic regression. For each gene, potential regulators were ranked according to the modulus of the regression parameter. The 20 highest-ranked regulators (corresponding to about 10% of of all regulators) were kept for the follow-up analysis.

Bayesian piecewise linear regression model. We adapted the model presented in[45] to create a piecewise linear regression model for the regulation of each gene, given its transcription factors. This model can learn the structure of the underlying regulatory network, as well as changepoints in the linear ordering of chips which separate different experimental conditions or perturbations. We incorporated sparse Poisson priors on the number of changepoints and the number of potential regulators for each gene. In contrast to,[45] this model assumes that the structure of the transcription factor-target gene interactions do not change with different experimental conditions, only the parameters associated with these interactions do. We learned the structure and parameters using RJMCMC.

RJMCMC simulations. The RJMCMC simulations for inferring changepoints and parameters of the Bayesian regression and multiple changepoint model were run on two high-performance computer clusters. For data set 2, simulations were started from two different initializations, and scatter plots of the marginal posterior probabilities of the edges were obtained for monitoring convergence. For 2×10^6 RJMCMC steps these scatter plots indicated sufficient convergence, but it turned out that these simulations could not be finished in time. We therefore had to reduce the simulation lengths to 10^5 RJMCMC steps, despite the fact that the scatter plots indicated a certain lack of convergence. The method could only be applied to data sets 1 and 2 for lack of time.

Submitted edge orderings. The submitted edge orderings were obtained as follows. For each target node, a set of 20 potential regulators passed the filter based on TESLA, corresponding to roughly 10% of the potential incoming edges. For these regulators, the original intention was to order the edges on the basis of the marginal posterior probabilities from the RJMCMC simulations. However, owing to hardware-related problems with the computer cluster (see below), the RJMCMC simulations did not reach the desired convergence level by the DREAM

[p] We used the R package `som` available on CRAN (`http:cran.r-project.org`)

deadline. To guard against false negatives resulting from potential entrapment of MCMC trajectories in metastable states, we computed the ranking from the arithmetic mean of the marginal posterior probabilities and the modulus of the corresponding regression coefficients, where the latter were rescaled to the unit interval. After ranking all the node-regulator pairs for those regulators that had passed the filter, the remaining edges were ranked on the basis of the modulus of the regression parameter obtained with TESLA. For those data sets for which no RJMCMC simulations could be run (see below), the rankings were solely based on the modulus of the regression parameters from TESLA.

Discussion

This method was among the top quarter of performers by overall performance, even though we had to grapple with some difficulties, which were as follows:

Data preprocessing. We spent substantial time on the preprocessing and the chip ordering, where we made various worrying observations. The correlation of the regression parameters obtained with Lasso from data subjected to different preprocessing schemes were rather modest, indicating that the preprocessing of the data matters a lot. Without explicit knowledge about the biological processes it is difficult to decide on the appropriate choice, and in the absence of this information we opted for a standardization to transform all gene profiles to zero mean and unit variance.

Chip ordering. In this approach, information about experimental conditions and perturbations resides in the chip ordering. It turned out that a one-dimensional ordering cannot group both the same experimental conditions and the same perturbation types together. This suggests that a two-dimensional multiple changepoint model would be a more appropriate approach. However, in the available time this model could not be implemented and tested, and we therefore decided to stick to the one-dimensional approach. Preliminary analysis based on TESLA showed poor correlation between the regression parameters obtained from different chip orderings. This indicates that the chip ordering matters a lot, but without extra biological knowledge it is impossible to decide what matters most: the experimental conditions, or the perturbations.

RJMCMC simulations. The net time available for the RJMCMC simulations - following the preprocessing steps and associated investigations - was about two weeks. Unfortunately, external circumstances beyond the control caused extended downtime of one of the computer clusters. As a consequence, the RJMCMC simulations could only be completed for two out of the four data sets, and even here the convergence was sub-optimal.

Lack of consistency between the methods. While we expect that the Bayesian regression and multiple changepoint model should achieve an improvement one TESLA, we were surprised by how low the edge rank correlations between the two methods were. This might be indicative of a certain paucity of true patterns in the data, possibly as a consequence of the choice of the pre-processing and chip ordering schemes.

Future Work. The proposed method critically depends on the pre-processing step, which is rather heuristic. Besides looking into a more principled alternative to SOMs for obtaining a chip ordering, especially such that explicit biological knowledge is included, we are working on extending the changepoint process to more than one dimension. As discussed above, this would allow groups of perturbations in addition to experimental conditions to be kept together, and we would expect that to be reflected in an improved network reconstruction accuracy.

10.4 Regression 7 – Simple L_1-regularization

In a graphical model, an edge between two variables means that these two quantities are conditionally dependent, given the remaining variables. In this case, these variables are genes, and for each gene we would like to find the transcription factors that (directly) influence a respective gene. This means we are looking for those target genes and transcription factors which are conditionally dependent, given the other transcription factors. For that purpose, target genes are regressed on the transcription factors. Non-zero regression coefficients indicate (conditional) dependencies, whereas (conditional) independence is indicated by zero coefficients. To identify zero and non-zero coefficients, we use L_1-type regularization techniques.

Data preprocessing. In order to detect sets of genes sharing common expression patterns we employed a clustering approach consisting of two steps: (1) computation of the pairwise similarity between genes and (2) the detection of clusters. As a similarity score (*cf.* (1)) we employed the Spearman correlation on the given raw data (*i.e.* using a data point per chip and gene from the provided data matrices). The pairs of genes with a Spearman correlation of above 0.8 were subjected to Markov clustering (*cf.* (2)) using an inflation parameter of 3.0. The data provided by DREAM5 shows a noticeable clustering structure. The identified clusters show very little between-gene variability suggesting to treat the gene clusters as a single entity when used as the response variable in the regression model (see below).

L_1-Regularization using the Lasso. We used an adaption of the method proposed by[58] who dealt with estimating

high-dimensional (undirected) graphs under the assumption of a multivariate normal distribution. In this special case, the method can be seen as an approximate version of a method proposed by.[30] The latter approach estimates the inverse covariance matrix ($\sum^{-1}$) of a high-dimensional multivariate normal distribution, with the sum of absolute values of the elements of this matrix being penalized. On one hand, the use of a penalty makes the matrix estimable; on the other hand, the chosen L_1-type penalty causes that some elements of the estimate of $\sum^{-1}$ were set to zero. Zero elements $(\sum^{-1})_{ij}$ in the inverse covariance matrix mean that the variables (or genes in this case) i and j are conditionally independent given the remaining variables (genes). In the situation given in the DREAM5 competition, however, the method[30] cannot be used, since no independent and identically distributed samples from a multivariate normal distribution are given. In addition, the estimated graph should be directed. In,[58] by contrast, the authors try to find conditional dependencies by regressing each gene separately on the remaining ones. A similar procedure can be applied to the DREAM5 data. A zero regression coefficient indicates that the target gene (the response) is not influenced by the respective explanatory gene (the regressor), given the other genes. To locate zero coefficients, a Lasso[84] penalty was used, so regression coefficients may be set to zero. Non-zero coefficients indicate conditional dependencies, and edges can be drawn from the corresponding explanatory genes to the target. This procedure was repeated with each gene serving once as the response. By construction, the resulting graph is directed. Since only transcription factors may regulate other genes, only transcription factors are considered as (potential) regressor genes (but all genes may serve as response). If any of the (potential) effector genes are deleted or over-expressed, this information is taken into account by adding indicator variables to the set of explanatory genes. If such an indicator is selected by the Lasso, an edge between the corresponding effector gene and the considered target is drawn as well. Also permutations can be taken into account by using indicator variables. If data come from time series, the considered target gene is also regressed on lag one of the transcription factors.

Obtaining confidence scores. We did not use stability selection[59] to compute confidence scores, but a more simple procedure. The Lasso, which is applied to the single regression problems to select variables (that is, effector genes), depends on a tuning parameter t, which determines the strength of penalization. More precisely, the residual sum of squares is minimized as a function of regression parameters $b_1, \ldots, b_p$, subject to $|b_1| + \ldots + |b_p| < t$. The smaller t, the higher the penalty and the less variables are selected. To derive a measure of confidence, we used the value of the tuning parameter where a considered edge is selected the first time. After normalization with respect to the most/least reliable edges (out of the first 100,000), these scores have values between zero and one.

Discussion. Though L_1-type regularization seems promising for the selection of edges in a gene regulatory network, the performance of this approach was rather bad on the DREAM5 data. This may be for two reasons. First, selection patterns of single regression models are rather unstable and hence less reliable. Second, it is doubtful that tuning parameters from different regression models can be directly compared. That means it is difficult to rank potential edges selected in regression models with different target genes. Tough stability selection is computationally much more expensive than the simple approach directly using the Lasso tuning parameter, it is apparently more reliable when judging on the relevance of edges in the network.

10.5 Regression 8 – Linear regression*

Linear Regression is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

A full description of this method can be found in[28] and is made available online.[q] This method was implemented as an attempt to estimate a Bayesian network. The computational challenge underlying Bayesian networks for gene regulatory network inference is to exhaustively search the space of possible regulatory relationships. Given the size of the DREAM5 datasets and number of regulatory relationships to be inferred a true implementation of a Bayesian network is intractable. By constraining the number of regulatory relationships considered and simplifying the mathematical model to a linear regression, a regulatory network can be inferred from the DREAM5 data. Given the underlying linear regression model and algorithm clustering results shown in Figure 2, this method was placed in the Linear Regression group.

10.6 Mutual Information 1 – Context Likelihood of Relatedness (CLR)*

CLR is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

A full description of this method can be found in[28] and is made available online.[r] The input into the CLR algorithm is a gene × condition matrix of expression values. The CLR algorithm progresses through 2 main steps. First, a matrix of mutual information values is calculated for all input pairs of genes i and j. As the authors suggest, mutual information was calculated using B-spline smoothing and the gene expression values were discretized into 10

[q] http://gardnerlab.bu.edu/data/PLoS_2007/
[r] http://gardnerlab.bu.edu/data/PLoS_2007/

bins. Second, CLR estimates the significance of a gene pair by comparing the mutual information value between gene i and j to a empirically defined background distribution of mutual information values. The significance of a gene pair is defined by a modified z-score. Transcription factor-target gene predictions are ranked according to the modified z-score and the highest scoring gene pairs are selected.

10.7 Mutual Information 2 – Mutual information*

Mutual information is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

A full description of this method can be found in[28] and is made available online.[s] For two random variables, X and Y, mutual Information is defined as:

$$I(X;Y) = \sum_{i,j} P(x_i, y_i) \log \frac{p(x_i, y_i)}{p(x_i)p(x_j)}$$

In this implementation, variables X and Y represent a transcription factor and target gene, respectively. Mutual information calculations require discretized data and as gene expression data from microarrays are continuous, a B-spline smoothing and discretization method is used.[22] The number of bins was set to 10 as suggested by the authors. Final transcription factor-target gene predictions were ranked and selected based on the highest to lowest mutual information scores.

10.8 Mutual Information 3 – Algorithm for the reconstruction of accurate cellular networks (Aracne)*

Aracne is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

A full description of this method can be found in[57] and is made available online.[t] As input, Aracne accepts a matrix of gene $\times$ condition expression values and a list of defined transcription factors. The Aracne algorithm can be separated into 2 main steps. First, a matrix of mutual information values is calculated for all input pairs of genes i and j. Aracne estimates mutual information using the Gaussian Kernel estimator.[8] Statistically significant relationships are determined and the non-significant edges are removed. Second, an additional pruning step is performed based on the the theoretical property known as the data processing inequality (DPI).[21] Given a gene interaction network, the DPI aims to eliminate any edges that can be explained through the remaining interactions in the network. Consider the small network $g_i \leftrightarrow g_j, g_j \leftrightarrow g_k, g_i \leftrightarrow g_k$, an edge between g_i and g_j will be removed if $I(g_i, g_j) \leq min[(g_i, g_k), (g_j, g_k)]$. After pruning, the remaining transcription factor-target gene relationships are then ranked based on their mutual information values.

The Aracne software package has several parameters that can be adjusted by the user. As stated, default parameters were used to best estimate a naïve application of Aracne, however, it should be noted that better results are likely to be achieved after tuning of two parameters, 1) the mutual information p-value cutoff, and 2) the DPI tolerance parameter.

10.9 Mutual Information 4 & 5 –The DREAM5 network inference challenge with a combination of fast tools

In the DREAM5 network inference challenge, the task is to discover relationships between genes from four gene expression datasets. We use mutual information and Fisher's score as the scoring function to compute a probabilistic dependency between pairs of variables, and combine it with the BLCD-HITON-PC search algorithm[52] to find the Y arcs in the network. An example of a Y substructure is $A \rightarrow C$, $B \rightarrow C$, and $C \rightarrow D$. The Y substructures have special properties that make causal discovery possible under plausible assumptions. In this example the arc $C \rightarrow D$ is a Y arc. The Y arcs in a Bayesian network represent unconfounded causal influences under assumptions.[54] Since the Y arcs represent unconfounded causal influences we expect that they could discover arcs with high precision and can be used to orient some of the edges that pair-wise mutual information and statistical analysis (Fisher's score) cannot accomplish.

Mutual information. We used the implementation for fast calculation of mutual Information for all pairs of genes in the challenge.[71] For the largest network (Network 4) in the challenge with 5950 genes and 536 chips, it took about 19 minutes to generate the pair-wise mutual information for the entire set of genes in the system.

Fisher's score. We used correlation coefficients (CC) implemented in MATLAB to find the correlation between all pairs of genes in the networks. This gave us additional information to access relationship for the whole set of pairs in the network. It is also very fast, which took 27 seconds to complete the whole run for Network 4.

BLCD-HITON-PC. We implemented the Bayesian local causal discovery algorithm (BLCD)[52,53] to efficiently identify unconfounded direct causal relationships of gene

[s] http://gardnerlab.bu.edu/data/PLoS_2007/
[t] wiki.c2b2.columbia.edu/califanolab

variables in the network with high precision. It consists of 2 steps: Markov Blanket step and Y arcs step. For the Markov Blanket generation step in BLCD, we used the computational causal discovery method HITON-PC[3,4] which outputs the parents and children set for each target variable. For the Y arcs generation step, it searches for the Y arcs locally in the parents and children set. The second step can be run in parallel for efficiency. For Network 4, we separated the Y arc discovery task into 15 groups and finished running in less than 2 and a half hours for each group. BLCD outputs the probability of a Y arc which we converted to a value of 1 or 0 using a threshold of 0.5. We used binary output with threshold of 0.5 since precision of Y arcs is high and we put higher weight than mutual information and Fisher's score.

Combination. We first combined the results of mutual Information and Fisher/Correlation coefficients score (CC) based on the ranking. Two ranking arrays are averaged and normalized into one array (called MICC array) containing numbers in the range from $[0, 1]$ so that the edge with number closer to 1 represents a stronger dependent relationship. Secondly, this array is combined with the BLCD binary output such that the arcs with BLCD output '1' and their reverse arc with BLCD output '0' are averaged with the MICC array. This process could fix part of the direction problem from mutual information and Fisher's score. For example, $I(a, b) = 0.5$ represents the confidence of which a can regulate b by mutual information or correlation coefficient or their combination; $I(a, b) == I(b, a)$; If BLCD outputs $blcd(a, b) = 1$ and $blcd(b, a) = 0$, we would average the result and the final output for these 2 arcs will be $s(a, b) = 0.75$ and $s(b, a) = 0.25$. This could output higher precision and recall than transcription factor or correlation coefficient or their combination if the BLCD output is correct.

Discussion. The reason we combine transcription factor and correlation coefficient is based on the preliminary experimental result for similar gene expression networks. It shows that if both mutual information and correlation coefficient can output good AUPR (Area under Precision and Recall curve), their combination can produce a better one; but it is possible that the CC performance is worse than mutual information, and then the combination result can be worse than the mutual information itself. So we would rather submit 2 results: Submission DSL is based on mutual information + correlation coefficient + BLCD-HITON-PC, and Submission DSL2 is based on mutual information + BLCD-HITON-PC.

There are some search strategies in BLCD based algorithm we could optimize to improve the performance. The original BLCD searches for Y arcs from a Markov Blanket set of each node that is derived by greedy search; however, it is not efficient to run the greedy search for such a high-dimensional dataset for this challenge. Thus, we run an efficient algorithm HITON-PC to output a smaller set (parents and children set), but it may reduce the recall. The threshold of output would also affect the final performance. If the threshold is too high (for example, 0.9), the number of Y arcs would be very small, so it would not change the result by just using mutual information or correlation coefficient; if the threshold is too low (for example, 0.1), the number of Y arcs would be very large but the precision may be reduced, so it may not positively affect the final result. Therefore, we use the default 0.5 as the threshold.

10.10 Correlation 2 – Pearson's correlation*

Pearson's correlation is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

Pearson's correlation coefficient r was calculated between all transcription factors x and all target genes y as follows:

$$r_{xy} = \frac{n\sum x_i y_y - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}},$$

where n is the number of measurements of x and y. x to y relationships were ranked via r_{xy}, i.e. positively correlated gene pairs receive the highers confidence.

10.11 Correlation 3 – Spearman's correlation*

Spearman's correlation is one of the 6 commonly-used, off-the-shelf algorithms run by the DREAM organizers.

Spearman's correlation ρ was calculated between all transcription factors x and all target genes y as follows:

$$\rho_{xy} = 1 - \frac{6\sum d_i^2}{n(n^2-1)},$$

where n is the number of conditions that x and y have been sampled and d is the difference in rank order between gene x and gene y over the n conditions. x to y relationships were ranked on ρ_{xy} and the most correlated gene pairs were selected.

10.12 Bayesian 6 – Regulatory network inference with Bayesian networks

We built a simple Bayesian network to model the influence of a potential transcription factor on a gene. The network includes observed variables for inputs, such as perturbations, knockouts, or overexpression applied in each

experiment. It then relates these to hidden variables for unobserved quantities, such as the magnitude of the influence or whether the perturbation affected this relationship. These hidden variables are then related to random variables for the observed quantities of transcription factor and target gene expression levels. An alternate network for the absence of regulatory influence modeled each expression level as an independent random variable with random observation noise. For each possible pair of transcription factor to target gene relationships (transcription factor-target gene), we computed the relative probability of each model (no influence, $A \rightarrow B$, $B \rightarrow A$) given the observation data. The highest probability for each pair was selected and sorted to give the final prediction.

Method. The Bayesian network used for the simplest case of a steady-state observation experiment involving a single transcription factor-target gene pair is shown in Figure SS19. The nodes of the network are:

- *Perturbation Enabled.* If a perturbation is present for the experiment, and which one (combinations of perturbations were treated as a unique perturbation, to keep the model simple).
- *Affected.* Bernoulli random variable for whether the target is affected by the perturbation
- *Perturbation Factor.* Gaussian random variable for the amount of perturbation
- *Knockout/Overexpression* Knockout or overexpression factor of the target for a given experiment
- *Edge Influence.* Gaussian random variable for the magnitude of influence of the transcription factor on the target gene
- *Transcription factor Level, Target Level.* Expression levels of the transcription factor and target genes, normalized to mean 1.0, variance 0.01 of observations from steady-state experiments.
- *Transcription factor Level Observed, Target Level Observed.* Observations with Gaussian noise

The null hypothesis of no relationship was a simple model where each expression level is an independent Gaussian random variable observed with Gaussian noise.

A more complex model was used for time series experiments, a one-step dynamic Bayesian network modeling the change in the target as a function of its prior level and the prior level of the transcription factor. The null hypothesis modeled each level independently as returning to mean with some time constant.

The data were analyzed using a Bayesian network library, to calculate the relative probability of the respective models (no edge, $A \rightarrow B$, or $B \rightarrow A$) given the observations. These calculations were then performed pairwise

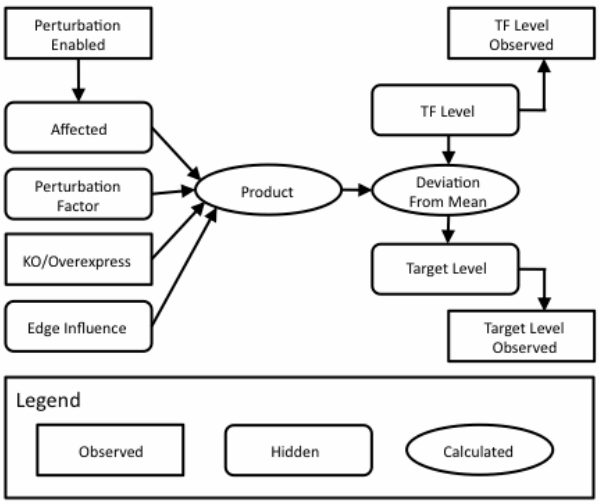

Figure S19: Inference method design for Bayesian 6.
The organization and categorization of information for the design of the Bayesian classifier.

for all possible transcription factor-transcription factor or transcription factor-target gene interactions in each network on an internal computing cluster, since the method is quite computationally intensive. Transcription factor-target gene interactions were modeled in both directions and the highest probability taken, on the assumption that the magnitude of influence was significant even if directionality was suspect. The 100,000 most likely edges were then selected and sorted.

Discussion.

This method did not perform well on any of the DREAM5 data sets. There are several arbitrary constant parameters in the model that define the prior distributions of the hidden variables. These were trained on a small (50-node) yeast network generated from GeneNetWeaver,[56] using a conjugate simplex method to find the parameter values that gave predictions best matching the actual network. The resulting model and parameters were then validated on the DREAM4 dataset, on which they performed reasonably. However, it was clear during the training phase that the prediction accuracy was quite sensitive to these parameters. It is likely that the DREAM5 datasets were sufficiently different from the training set that the parameters were no longer suitable.

There are a couple of ways in which this issue can be addressed in the future. Most simply, the parameters can be trained on the actual dataset; in the absence of a known network, we could optimize for fit of predicted probabilities to an expected distribution based upon general features of the target network (*e.g.* in- and out-degree distributions). This would, however, be quite computationally

intensive. Alternately, the parameters themselves can also be incorporated into the model as hidden variables. This would require more complex Bayesian network modeling algorithms, and the library we used did not support model selection over such second-order networks (*e.g.* a random variable whose mean and variance were themselves defined by other random variables). It might also be possible to reconfigure the Bayesian network itself to reduce its dependence on such arbitrary parameters.

10.13 Other 1 – Inferring regulatory networks using tree-based methods

This algorithm, called GENIE3 (for "Gene Network Inference with Ensemble of Trees"), decomposes the prediction of a regulatory network between p genes into p different regression problems. In each of the regression problems, the expression pattern of one of the target genes is predicted from the expression patterns of all known transcription factors, using a tree-based ensemble method called Random Forests.[13] The importance of a transcription factor in the prediction of the target gene expression pattern is taken as an indication of a putative regulatory link. Putative regulatory links are then aggregated over all genes to provide a ranking of interactions from which the whole network is reconstructed.

GENIE3 does not make any assumption about the nature of gene regulation, can deal with combinatorial and non-linear interactions, produces directed gene regulatory networks, and is fast and scalable. This method is described in more detail in[40] and available online.[u]

Network inference procedure. The GENIE3 procedure works as follows:

1. For gene $j = 1$ to p:
   - Generate the learning sample of input-output pairs for gene j: $LS^j = \{(x_k^{TF}, x_k^j), k = 1, \ldots, N\}$, where the input x_k^{TF} is the vector of expression values of all known transcription factor genes (except gene j if it is a transcription factor) in the k^{th} experiment, the output x_k^j is the expression value of gene j in the k^{th} experiment, and N is the number of experiments.
   - Use the Random Forests method on LS^j to compute confidence level $w_{i,j}$, for each transcription factor gene $i \neq j$.
2. Aggregate the p individual gene rankings to get a global ranking of all regulatory links.

Tree-based ensemble methods. The basic idea of tree-based methods in regression is to recursively split the learning sample with binary tests based each on one input variable (here, the expression of one potential transcription factor). These tests are optimized in such a way as to reduce as much as possible the variance of the output variable (here, the expression of the target gene) in the resulting subsets of samples. Candidate splits for numerical variables typically compare the input variable values with a threshold which is determined during the tree growing.

Single trees are usually very much improved upon by ensemble methods, which average the predictions of several trees. In the network inference procedure, we used the Random Forests method,[13] *i.e.* each tree is built on a bootstrap sample from the original learning sample, and at each test node, K attributes are selected at random among all candidate attributes before determining the best split. We used $K = \sqrt{n}$, where n is the number of input variables, and grow ensembles of 1000 trees.

We selected the Random Forests method (with this value of K) because it leads to the best performance among other tree-based ensemble methods on the prediction of the *E. coli* regulation network.[40]

Variable importance measure. To associate a confidence level $w_{i,j}$ to the regulation of target gene j by the transcription factor gene i, we directly exploit the variable importance measure of the expression of i as derived from the tree-based model learned for j. While several variable importance measures have been proposed in the literature for tree-based methods, we consider in this procedure a measure that computes at each test node $\mathcal{N}$ the total reduction of the variance of the output variable due to the split, defined by:[14]

$$w(\mathcal{N}) = \#S.Var(S) - \#S_t.Var(S_t) - \#S_f.Var(S_f)$$

where S denotes the set of samples that reach node $\mathcal{N}$, S_t (*resp.* S_f) denotes its subset for which the test is true (*resp.* false), $Var(.)$ is the variance of the output variable in a subset, and $\#$ denotes the cardinality of a set of samples. For a single tree, the overall importance of one transcription factor gene is then computed by summing the w values of all tree nodes where the expression of this transcription factor was used to split. For an ensemble, the importance is then obtained by averaging importance scores over all trees in the ensemble.

Global regulatory link ranking. Each tree-based model thus yields a separate ranking of the transcription factors as potential regulators of a target gene in the form of weights $w_{i,j}$ computed as sums of total variance reductions. The sum of the importances of all input variables for a tree is equal to the total variance of the output variable explained by the tree, which in the case of unpruned trees (as they are in the case of Random Forests ensembles) is usually very close to the initial total variance of the output.

[u] http://www.montefiore.ulg.ac.be/~huynh-thu/software.html

As a consequence, if we trivially order the regulatory links according to the weights $w_{i,j}$, this is likely to introduce a positive bias for regulatory links towards the more highly variable genes.[40] To avoid this bias, we first normalized the gene expression values so that they all have a unit variance in the training set, before applying the tree-based ensemble method. This normalization indeed implies that the different weights inferred from different models predicting the different gene expressions are comparable.

Discussion. We developed GENIE3, a gene network inference algorithm based on feature selection with Random Forests. This method was the overall top performer of the challenge. It was also the top performer on the *in silico* generated data but did not perform as well as some other teams on the *in vivo* microarray data. Interestingly, setting Random Forests parameter K to n would have significantly improved the performance on the *in silico* data but with a decrease of the performance on the *in vivo* networks. One potential reason for the better performance on the *in silico* benchmarks is that the *in silico* dataset probably contains more statistically useful experiments than the *in vivo* datasets in which there may be some redundancy among the experiments or some bias in their selection. Such differences in terms of the quality of the data might affect more the non parametric approach than alternative approaches that make stronger assumptions. Other potential reasons may originate from the fact that the *E. coli* and *S. cerevisiae* gold standard networks are not complete and noisy, as well as from the discrepancy that certainly exists between the simulation model used to generate the *in silico* data and the *in vivo* regulation mechanisms of *E. coli* and *S. cerevisiae*. In the future, we would like to investigate further these differences.

In principle, any feature selection algorithm[74] could be substituted to Random Forests within the general procedure. Method *Regression 1* (see **Supplementary Note 10.1**) could be seen as an instance of this procedure. We have also carried out experiments with feature ranking based on linear models trained with support vector regression but these methods were not as successful as tree-based ensemble methods. In the future, we plan to consider other feature selection techniques.

10.14 Other 2 – Inferring gene regulatory networks by ANOVA

This approach to network inference is based on the assumption that transcription factors (TFs) and their corresponding target genes (TGs) exhibit mutual expression dependencies in at least a subset of the measured experimental conditions (time points, perturbations etc). Such candidate interactions, *i.e.* pairs of a TF and a TG, are ranked by a score s. The score s can be any measure of dependency between the expression of the TF and its TG. Frequently used measures of dependency are based on Pearsons or Spearmans correlation coefficients, mutual information, or in case of Bayesian network inference on conditional probability tables.

We evaluate candidate interactions by η^2,[19] a non-parametric, non-linear correlation coefficient obtained from a two-way analysis of variance (ANOVA). It is fast, easy to apply and does not require discretization of the input data. Refinements of this approach also enable incorporation of additional information, *e.g.* the specific over-expression or deletion of genes in given experiments. This method will be described in more detail in.[44]

Data preprocessing. Basal gene levels can be quite different between experiments. To account for these differences, we transformed the absolute expression values into expression fold changes. Fold changes were computed by mapping each measured condition to one or more control conditions from the same experiment. Control conditions were defined via their set of treatments (such as knockout, over-expression, drug treatments) that is required to be a subset of the treatments applied in the given measured condition. More than one fold change may be computed if more than one control is available. For example, the control conditions for an experiment where two genes are deleted (double knockout) may be a single knockout and/or the wild type. In the case of time series, we required the time points for measured conditions to match the time points of the corresponding controls.

Note that each control usually has several replicates. Fold changes were computed by subtracting the (log-transformed) gene expression values of the controls, averaged across the replicates, from the given chips. As mentioned above, the mapping is not unique, *i.e.* a given chip can have 0, 1 or more control replicate sets assigned. For instance, from the 805 chips (487 different replicate sets) of the *E. coli* compendium, we could compute gene fold changes for 599 chips (379 replicate sets). Because of the multiplicity of controls, we obtained a total of 935 fold changes per gene (602 replicate sets).

Comparison of TFs and putative TGs by η^2. We employed a two-way ANOVA to test the differential expression of TFs and their putative TGs. The ANOVA compares the means of populations that are classified in two different ways, or the mean responses in an experiment with two factors. The factors analyzed here are the expression of two different genes (factor or dimension A) across a range of conditions (factor B) that is represented as a 2 rows by 602 columns matrix.

In this analysis we tested (1) if at least two experimental conditions exhibit significant differences in their population means (*i.e.*, differential expression) and (2) if these

differences exceed the differences between the expression of the TF and a TG. Phrased in terms of the two-way ANOVA, the strength of an association is proportional to the fraction of the variance across conditions (factor A) in the total variance. Such a fraction (F-value) follows the F-statistic, which can be used to derive the statistical significance of the involved factors as p-values.

Refinement of the basic ANOVA. Genetic perturbations are valuable as they help to establish directed causal relationships between regulators and their TGs. The information whether a TF was subjected to genetic perturbations (deletion or overexpression) was taken from the provided chip-feature descriptions. Conditions that indicate a perturbation of the currently tested TF are given a higher weight than other conditions. Informally, the weight is processed by inserting (w-1) additional copies of such a condition into the ANOVA matrix. Note that conditions where non-TFs or TFs other than the currently tested TF are perturbed receive the standard weight. We set $w = 50$ based on an analysis we performed on expression data obtained from the M3D database[27] and a gene regulatory network obtained from RegulonDB.[31]

Discussion. For the detection of dependencies we proposed the measure η^2 that is derived from an analysis of variance (ANOVA). To our knowledge, η^2 has not been widely applied to network inference or to other problems in Bioinformatics, although it has a number of features that can facilitate the detection of gene dependencies. Like Pearson's correlation, but in contrast to Bayes conditional probability tables or mutual information, η^2 does not require the discretization of the input data. This increases the robustness of this method as inappropriate discretization might lead to loss of signal. In contrast to Pearson's linear correlation coefficient, η^2 is a non-parametric, non-linear correlation coefficient.

Some of the known *E. coli* interactions identified by this approach were quite interesting biologically. For instance, an interaction between the multiple antibiotic resistance (`mar`) genes `marA` and `marB` was active after antibiotic treatment but not in growth phase experiments.[44] The measure η^2 allowed us to detect such local correlations arising from condition-specific interactions. This increased sensitivity is due to the effective utilization of replicated measurements to model the measurement error and to estimate the statistical significance of TF-TG dependencies.

In the future, we intend to further improve this ANOVA-based inference approach by including a dedicated treatment of time series and by using conditional correlations to distinguish direct from indirect interactions.

10.15 Other 3 – Network inference through Boolean networks

The underlying model in this method is a Boolean network where the topology and logic is inferred from continuous expression data. It is based on the information theoretic conditional entropy criterion,[5] but assumes that the Boolean state of the network is not directly observed. Instead we are given a dataset of continuous measurements that reflect probabilistically the Boolean state.

For every gene X, we want to find the set of regulators Y that give the best conditional entropy score $H(X|Y)$ among all sets of putative transcription factors up to some size limit. Since conditional entropy is computed for discrete random variables, and we have continuous measurements, we first transform the continuous data into a set of discrete observations. The simplest way would be to discretize the continuous data, *e.g.* set all values above some threshold to 1 and all values below it to 0. Instead, we interpret a vector of continuous values as a probability distribution over all possible Boolean vectors of the same dimension. To put it simply, instead of creating one Boolean vector with probability 1 for every continuous vector, for every continuous vector we create all possible Boolean vectors of the same dimension, and assign each such vector a probability. The probabilities are chosen as follows: we first normalized the continuous expression values of every gene to have mean 0 and standard deviation 1.5 (a value determined empirically). After normalization, we set the probability that a single (one-dimensional) continuous value c corresponds to the Boolean value 1 to $\frac{1}{1+e^{-c}}$(the logistic function with input c). The probability that a continuous vector $\bar{c}$ corresponds to a specific Boolean vector $\bar{b}$ then becomes:

$$p(\bar{b}|\bar{c}) = \prod_{b_i=1} \frac{1}{1+e^{-c_i}} \prod_{b_i=0} \left(1 - \frac{1}{1+e^{-c_i}}\right),$$

where c_i (b_i) is the value of the i^{th} entry of $\bar{c}$ ($\bar{b}$). Note that by setting the standard deviation we avoid using any parameters in the logistic function. Given a continuous dataset of N samples (experiments/microarrays) that are assumed to be independent and identically distributed, the probability of seeing the Boolean vector $\bar{b}$ in this dataset is:

$$P(\bar{b}) = \frac{\sum_{\bar{c}^i \in samples} p(\bar{b}|\bar{c}^i)}{N}$$

In other words, for each Boolean vector we sum the probabilities that each continuous vector corresponds to it.

With the probability distribution over all Boolean vectors in hand, we can use information theory to evaluate different topologies of the network. Similar approaches were

previously used for network reconstruction.[46,49] Denote by $H^C(X|Y)$ the conditional entropy for a gene X and a set Y of regulators as computed using continuous data. We selected for every gene X the set Y of regulators that gave the best $H^C(X|Y)$ score among all sets and transcription factors of size ≤ 3.

For time-series, X was taken from time t and Y from time $t-1$. Otherwise, we assumed samples were in steady state and X and Y were taken from the same experiment. Experiments in which X is perturbed were discarded when computing $H^C(X|Y)$.

For the DREAM5 network inference challenge, the transcription factor-target gene relationships identified by the above procedure were ranked according to the conditional entropy of the regulator set to which the transcription factor belonged to. This constituted 4,738, 13,120 and 17,519 interactions for networks 1,3 and 4 respectively. Since the challenge allowed submitting up to 100,000 regulatory interactions, we added to the list of predictions those pairs of genes that had the highest correlation, ranked by their correlation. In case of time series, the correlation was computed between the level of the regulator at time $t-1$ and the level of the regulatee at time t. In order to cope with the large number of regulator sets in networks 3,4 we used a computer cluster to distribute the tasks.

Note that this method is inherently incompatible with the scoring scheme of DREAM5, because it assigns a score to the set of regulators of every gene and not to every transcription factor-target gene pair separately. Another difference is the limit of three regulators per gene that we imposed. This limit allowed us to examine all sets of 3 regulators, a very large set given the DREAM5 network's size. In practice, however, some genes have more than 3 regulators. A speedup version in which regulators are selected incrementally can allow more regulators per gene.[34] Finally, a set of 3 regulators will always score better in practice (sometimes insignificantly better) than a set of 1 or 2 regulators, and we did not set a criterion that prevents the addition of regulators that do not improve the score significantly.

An advantage in this method's focus on sets of regulators is that there is a natural way to derive the regulatory logic given the set of best scoring $H^C(X_i|Y)$ for every gene i. This can be achieved by performing steepest descent on the function , *i.e.* on the total entropy of the network. The partial derivative with respect to every continuous variable can be computed exactly.

A tool implementing an extension of this method is available for download at: acgt.cs.tau.ac.il/modent/

10.16 Other 6 – Network inference using quantitative modeling and evolutionary algorithms

The method presented here for inference of DREAM5 gene networks is based on quantitative modeling using an integrative evolutionary approach. The model used is a single-layered artificial neural network (ANN) and allows for extraction of qualitative information on connections, with the advantage of maintaining the ability to simulate quantitative behavior. Given the high dimensionality of the four datasets, the networks are less suitable for direct quantitative modeling, as simulation for model evaluation is computationally expensive, and the gene interaction space is very large. Although reverse engineering is difficult, quantitative models are valuable for networks of this size, as they allow for large scale in-silico simulation of the real system. We have included in the workflow several mechanisms to reduce the search space required for reverse engineering. These include grouping of tightly correlated genes into modules and filtering of putative transcription factors for each gene, prior to model inference by evolutionary optimization. Additionally, models have been obtained for each gene at a time for non-transcription factor genes, while whole network analysis has been performed for the transcription factor subnetwork only.

Data preprocessing. Due to the requirements of modeling and the inferential approach, which uses ANNs, the expression values in the datasets had to be scaled to values on the interval $[0, 1]$. Scaling was performed differently for the *in silico* and *in vivo* data, as values in the former were more homogeneous than the latter, due to lack of experimental differences. For the former, all values were scaled by dividing by the maximum value in the dataset. However, in the *in vivo* datasets, individual genes had very different ranges of expression. Additionally, we have observed that, for some knockout (KO) genes, expression values were very large. In consequence, each gene vector (column) has been first translated so that the minimum value over all experiments is close to 0, after which the scaling of all values (as above) was performed. This ensured a common scale for gene values and, at the same time, brought KO genes to a low level of expression, while not affecting the oscillations seen in the data.

Module computation. One dimensionality reduction mechanism was grouping tightly correlated genes (values from all available experiments) into modules, and considering these as a single gene in the network. These may correspond to operons, which are common functionality groupings in GRNs. Module computation was based on pair-wise Pearson correlation and thresholds used were 0.9 for Networks 1, 3 and 4 and 0.95 for Network 2 (the higher threshold used for the last dataset was due to the limited number of experiments available, compared to the others).

Regulators of a module, as well as all transcription factors included, were considered to regulate all genes within that module.

Transcription factor filtering. A second mechanism, aimed at reducing the search space, filtered out the regulators that were not likely to influence a certain gene. Filtering was based on KO data and the Pearson correlation coefficient of the expression patterns for the gene and the transcription factor, (all experiments). Specifically, if a KO experiment for the transcription factor was available, the transcription factor was maintained in the list of possible regulators for the current gene only if the absolute value of the log-ratio for that gene was larger than 0.5 or if the absolute value of the correlation was larger than 0.3. These thresholds are not particularly high, and have been chosen in this way because the aim was to filter out those transcription factors with no effect, (to obtain a smaller set of transcription factors, which the inferential algorithm can further refine), and not to obtain the final list at this stage. In the case where no KO experiment existed, the correlation threshold was decreased to 0.1. This strategy was applied to all datasets and resulted, on average, in filtering out between 50 and 60% of the possible regulators

Model. The GRN has been modeled as a single-layered ANN, consisting of one neural unit per gene. Each unit i takes as input the expression values of the regulators of gene g_i (*i.e.* g_j) at time point t and computes the expression level for gene g_i at time $t+1$, using the input weights w_{ij} and the logistic function for activation.

Algorithm. The algorithm, based on the idea of nested optimization introduced by,[41] was chosen for extension here as it performed well in a previous comparison of evolutionary algorithms for GRN inference.[78] In the original algorithm, optimization is divided into structure and parameter searches. The former is performed by a genetic algorithm, which aims at finding the correct regulators for each gene, *i.e.* the structure, given a connectivity threshold. Each candidate structure is then evaluated by the parameter search phase, using back propagation, with wild-type time series data for training. The algorithm uses a divide-and-conquer approach, *i.e.* optimization is performed for one gene at a time. An extension of this algorithm has been developed and used here. This performs an additional optimization stage, (for the complete network), starting with multiple single gene models, with different connectivity thresholds. In this way, the connectivity for each gene is also optimized to fit the data. This second optimization stage is only performed on the sub-network of transcription factors. Again, the focus is dimensionality reduction, and was possible because the behavior of the non-transcription factor genes does not affect any other genes in the network. Additionally, the algorithm has been enhanced to include other types of data, along with wild-type time series data. Thus, population initialization is based on knockout and over-expression experiments, while model evaluation takes into account the simulation ability for steady state and time series knockout experiments, as well as perturbation time series.

Qualitative information extraction. Multiple runs were performed and the models obtained were analyzed to extract the network layout. A weight was computed for each interaction using the number of appearances in these different models and on the magnitude of corresponding ANN weights.

Discussion. Upon publication of the gold standards, we have analyzed the effect of the dimensionality reduction techniques employed. The modules obtained were validated using interactions in the data, with groupings sharing all transcription factors in the synthetic network, and most transcription factors in the *in vivo* gene expression data, (with better results obtained for the *E. coli* network). This indicates that usage of modules is relevant for this analysis, although sometimes hindered by noise in *in vivo* data. Transcription factor filtering, however, was not as successful, as it also filtered out a large number of true interactions (29%, 41% and 46% for the three networks), which explains the poor qualitative results obtained overall. This indicates that very low correlation or KO log-ratios, in the data for two genes, do not exclude the possibility of meaningful interaction, especially in the case of *in vivo* data. We have re-applied the algorithm without the filtering step (for the *E. coli* network only, due to time restrictions), but results did not show significant improvement, as the search space becomes huge without this preliminary step.

The low number of direct interactions retrieved may also be due to the fact that a quantitative model displays good simulation abilities even when interactions are indirect (*i.e.* if gene a regulates gene b which in turn regulates gene c, the algorithm may only (correctly) identify the effect of gene a on gene c, but for which a direct interaction does not exist). Given the stochastic nature of the inferential algorithm, and the high dimensionality and noise in the data, this could be one explanation here. However, the quantitative nature of the model does have its advantages, *e.g.* the possibility of simulating continuous behavior. In the future, further development of the algorithm will use stochastic evaluation for candidate models, to overcome noise over-fitting. Additionally, as transcription factor filtering has proved to be unreliable, we have also incorporated a custom mutation procedure within the algorithm, (based on KO experiments and correlations), which has demonstrated promising preliminary results on smaller networks, and will be applied to DREAM5 data in the near future.

10.17 Other 7 – Detecting interactions by generalized logic

We responded to the DREAM5 network inference challenge with a generalized logical network modeling (GLN) approach. In GLN modeling, an interaction is represented by the generalized logic or truth table, similar to many other approaches for discrete dynamic networks such as discrete dynamic Bayesian networks. What distinguishes this GLN approach is its network reconstruction from discrete data using a χ^2 test. Through this test, GLN modeling accounts for linear and nonlinear interactions, combinatorial effects, complexity of interactions, and time delay, all through the significance of the χ^2 test statistic.

Log transform on data from Network 1 (*in silico* yeast network). On data from Network 1 only, we applied the natural log transform $\ln(x)$. When a value is zero, we set it to $\ln(y/10)$, where $y =$ *the smallest positive number*. There is no negative value in the data set. We noticed that the original data, pooled for all genes, are uniformly distributed between 0 and about 1, far from a normal distribution. The intention of the log-transform is to be more sensitive to knockout or suppressed genes. Therefore, through quantization (next step) on the log-transformed data, we can distinguish knockout or suppressed values from normal values, without jeopardizing over-expressed genes at large values around 1, since the log function is close to linear at around 1. For the other three networks, we used the original data without the log transform because they are already symmetrical, uni-modal and bell shaped, close to a normal distribution.

Quantization of continuous observations to discrete ones. We quantized the original continuous observed data for each gene into a various number of discrete levels. This was achieved by determining the number of levels for each gene first and then using an optimal distance based clustering method. We first determined the number of quantization levels k for each gene using the Gaussian mixture model through the R package MClust. It chooses an optimal number of components that maximizes the Bayesian information criterion (BIC), considering both the likelihood of a fit and the number of parameters used in a Gaussian mixture model. However, if only a single Gaussian component is obtained, we set the quantization levels k to 3 instead of 1. We do this to capture changes in a normal random variable. Given the number of levels for each gene, k, we developed and applied a distance-based clustering method called Ckmeans.1d.dp (available in the R package repository) to quantize the continuous values of each gene to discrete ones. This method minimizes the sum of squares of within cluster distances by dynamic programming, the same objective of the standard k-means algorithm. The difference is that the algorithm guarantees to return an optimal clustering, but the k-means method does not.

GLN modeling to obtain highly significant interactions. The GLN modeling establishes interactions through the χ^2 test among nodes in a network. An interaction is defined as a many-to-one relationship between a set of parent nodes and a child node. We first converted the four given sets of DREAM5 files for each network to four trajectory collection files. The χ^2 test assesses the association between hypothetical parents and a child by the deviation from the expected counts when no association exists. Each gene node, starting with "G", was treated as an internal node in the network; each perturbation node "P" was an external node in the network. Each chip was treated as a single steady state snapshot. Therefore, for each trajectory collection file of a network, there was the same number of trajectories as chips. The time course information was thus not utilized. This decision was made based on the observation that the time courses were usually short and may not provide much dynamic transitions of the system. Correspondingly, we used a Markov order of zero for the modeling. Other parameters in GLN modeling were set as follows. In each run, we set the maximum number of regulatory (parent) nodes per node to 2. We did not allow self-cycle as it would lead to the choice of each node as its own parent in a zero-Markov order network. The α, or test size, for each χ^2 test was set to a very small number, specifically, 10^{-60} for network 1, 3 and 4, and 10^{-40} for network 2 based on the size of each network. This small k was set to counter the multiple testing effect. As only ranks are relevant, we did not perform permutation test to obtain a more accurate p-value for each interaction. After each run on a network, interactions were ranked by their p-value in the χ^2 tests. If p-values were same, interactions with less degrees of freedom were ranked higher. If there was still a tie, then an interaction with a higher χ^2 value would be ranked higher. When reporting the modeling results, we returned all interactions that were considered significant, regardless of how many interactions were reported for each node. Interactions with more than one parent were broken into multiple parent-child pairwise interactions sharing the same p-value for final submission.

Discussion. In the DREAM5 network inference challenge, data from three of the four networks were no longer synthetic, but were instead collected from *in vivo* biological experiments. We expect less drastic change in the data sets than what was relied on by the z-score test in the previous DREAM challenges. Rather, we anticipated that the χ^2 test based GLN modeling may work with the real biological data sets, now perhaps recording consistent but less drastic expression change in response to gene knockout or environmental perturbations.

In this network reconstruction, we did not address direct versus indirect interactions. We also did not consider the

temporal relationships in the data sets due to sparse sampling, despite that GLN is capable of doing so. These may have limited the performance of the GLN algorithm in the DREAM5 network inference challenge.

10.18 Other 8 – Finding gene-gene interactions by concurrence of change between conditions

We developed a computational method to detect gene-gene interactions by concurrence of changes in genes between different experimental conditions. Specifically, change in expression level of gene i from one condition to another, should cause shift in expression of target or downstream genes of gene i. The approach is motivated by a need to detect transcription factors and their target genes through knockout experiments.

The two-sample t-test is used to detect shift in expression from one condition to another. In order to assert an interaction based on concurrence in expression shift of involved genes, it is necessary to detect the shift. The p-value from the two-sample t-test is the significance of the shift of a gene between two conditions. For each pair of conditions, a p-value was obtained for each gene. Two conditions were considered the same if and only if the data were collected with the same experiment identifier and under the same experimental setup. Experiments with different identifiers were considered as different conditions despite the same experimental setups. Experiments with different setups, under different perturbations or at different time points, were different conditions, regardless of their experiment identifiers.

Concurrence of changes is identified by a χ^2-test on a probability table. Under two conditions, shift concurrence was defined as a shift in expression of transcription factor-target gene combinations or neither in targets nor transcription factor combinations. A probability table (Table 10.18) illustrates probabilities of four possible events that can occur under two conditions.

Concurrence in a shift among genes involved in an interaction was detected by a χ^2-test. For each interaction, a contingency table was created to record the number of combinations of shifts/no shifts for a given transcription factor-target gene pair collected from all pairs of different conditions. A p-value (p_x) from a χ^2-test on the contingency table was the significance of either the concurrence (in blue shading in Table 10.18), or non- concurrence (in yellow shading) of a candidate interaction under all conditions. We used the sign of Pearson's correlation coefficient, CC, between the shift/non-shift count of the transcription factor-target gene pair to differentiate concurrence versus non-concurrence. Finally, a confidence score was assigned to each potential interaction as:

$$score = \begin{cases} \frac{p_x}{2} & CC < 0 \\ 1 - \frac{p_x}{2} & CC \geq 0 \end{cases}$$

To generate potential interactions, all combinations of 1 and 2 transcription factors were enumerated to form an interaction with every target gene. According to the confidence scores of each interaction, top 100,000 interactions were reported for each network.

For Network 1, a log transformation has been applied to emphasize the knockout or suppression effect of a gene, as the original data were not close to a normal distribution to sacrifice small but significant expression values.

Discussion. In the DREAM5 netowrk inference challenge, data from three networks were generated from real biological experiments on three organisms. Nonlinear regulation patterns are most likely in these organisms. But linear regulation patterns were the most reported by this method, though it is a non-parametric approach. Since the table measured concurrence was created by addition, the knockout or knockdown effects were smoothed. And we have not considered the consistency of concurrence of changes.

10.19 Meta 1 — Inferring gene regulatory networks using knockout data and resampling

This method for the DREAM5 network reconstruction challenge was a pipeline closely resembling that of the best-performing pipeline from the DREAM4 challenge.[32,51,69] This pipeline is composed of three core methods that we have shown can be combined in a mutually reinforcing manner using a resampling approach. The three core methods are: 1) median-corrected z-scores (MCZ), which assigns confidence to regulatory edges based on a z-score defined from the genetic knockout data; 2) time-lagged Context Likelihood of Relatedness (tlCLR)[32,51] which is based on Context Likelihood of Relatedness (CLR),[28] and explicit use of the time-series data and mutual information[77] to assign confidence to regulatory interactions; and 3) the Inferelator 1.0 (Inf1)[10,11] which learns dynamics as well as topology by explicitly using the time-series data to parameterize a linear ordinary differential equation model using an l_1 constrained linear regression and model selection method.[93] For the DREAM5 network inference challenge we wanted to use as much of the winning inference pipeline from DREAM4 as possible while making some modifications to account for the differences between the DREAM4 and DREAM5 datasets.

The DREAM4 dataset was composed only of simulated data, and contained a knock-out of every gene, including

Between two conditions	No shift in targets	Shift in targets
No shift in Transcription factor candidate	$p_t p_c$	$(1-p_t)p_c$
Shift in Transcription factor candidate	$p_t(1-p_c)$	$(1-p_t)(1-p_c)$

Table S1: χ^2 contingency table for Other 8
Between two conditions, p_t is the p-value measuring the probability of no-shift of a target gene; p_c for a transcription factor candidate gene. The top-left cell stands for the probability of neither shifted, while the bottom-right cell the probability of both shifted.

all transcription factors. The core method 1, MCZ, took advantage of this complete genetic knock-out data. The DREAM5 data were either from *in vivo* experiments, or simulated to mimic *in vivo* experiments, and contained relatively few knock-outs of transcription factors, particularly in light of the order-of-magnitude increase in the number of genes. Additionally, some of the experiments contained replicate measurements and/or a matched wild-type, *i.e.*, a wild-type measurement performed by the same lab. We took advantage of the replicate measurements and matched wild-type measurements whenever possible. We developed a new core method 1, which we refer to as KOs + PKOs (knock-outs plus pseudo knock-outs). In this method we ranked regulatory interactions for the transcription factors for which knock-outs were available. In order to alleviate the issue of having relatively few KOs of transcription factors we identified conditions which behaved similarly to KOs, *i.e.*, only a few regulators were expressed at levels significantly lower than their wild-type expression. We called these conditions PKOs, and ranked regulatory interactions based on these conditions as well.

The pipeline for DREAM5 consisted of three core methods: 1) KOs + PKOs, 2) tlCLR, and 3) the Inferelator 1.0. Each method was run on 50-80 bootstrapped samples of the data and combined as shown in Figure SS20. For a detailed description of the tlCLR and Inferelator methods and the method for combining core method results, we refer the reader to.[32]

Methods. We denote by X the $N \times M$ DREAM5 data matrix consisting of M measurements of N genes, where each measurement is a different experimental condition: a wild-type control, a gene deletion, *etc.* X can then be split into subsets comprising groups of similar conditions: taking the rows corresponding to all annotated wild-type conditions gives the wild-type matrix X^{wt}. We indicate matrices for time-series (X^{ts}) and genetic or environmental perturbation (X^{pt}). Unlike the DREAM4 data, these groups are not mutually exclusive: some conditions may be time series that include genetic and/or environmental perturbations. X^{pt} is defined as all perturbations not part of time series experiments, while X^{ts} is all time-series data regardless of perturbation. In the following text, we will also refer to two additional subsets that are particularly useful for inference: the set of first and last time points in all time-series conditions (X^{flts}) and the set of all steady-state gene deletions (X^{ko}).

Knock-Outs. For each condition in X^{ko} that contained a knock-out of a transcription factor, x_j, we ranked the putative targets of x_j, storing the results in a column of Z^{ko}, the matrix that stores the confidence for each regulatory interaction. To build Z^{ko}, we found the genes which were differentially expressed in the knock-out conditions of transcription factor x_j relative to their wild-type expression. We denote the conditions where x_j was knocked out by $C = c_1, \ldots, c_p$ where $c_1 \ldots c_p$ index the columns of X. Note that C will contain only one index unless there are replicate experiments of the knock-out of x_j. X^C is the matrix formed from these rows of X. Similarly, $W = w_1, \ldots, w_q$ denotes the columns of X corresponding to the wild-type control for that set of experiments (*i.e.*, a wild-type measurement from the same lab that conducted the knock-out experiment), and X^W the resulting matrix. If no such experiment-specific wild-type existed, we used the set of all wild-type conditions, X^{wt}, in place of X^W. Differential expression for X^C relative to X^W was then calculated using one of two different methods. When X^C and X^W both contained more than three replicates (*i.e.*, three columns) we used Significance Analysis of Mircoarrays (SAM),[85] as it has been previously shown to successfully identify differentially expressed genes. We developed a method $diff\_exp$ to calculate differentially expressed genes when either X^C or X^W contained fewer than three replicates. In short, for each transcription factor, x_j, for which a knock-out existed we performed the following steps: calculate a z-score for the log-difference in expression, calculate a z-score for the fold-change in expression, and combine the two using Stouffer's method. These steps are described mathematically below. Note that these steps rank the putative targets of one transcription factor, x_j, and hence constitute one column of Z^{ko}. This algorithm is repeated for each transcription factor.

First, the data are transformed from log-space (in order

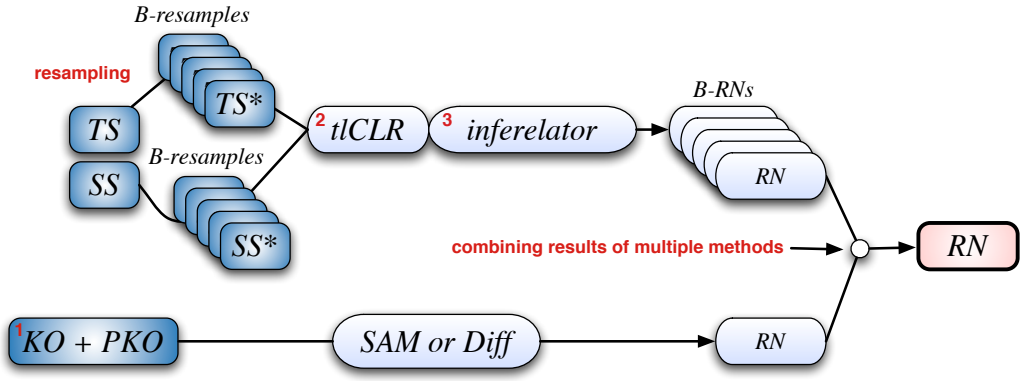

Figure S20: Network inference pipelines tested for Meta 1.
We developed an inference pipeline closely resembling the winning pipeline from DREAM4.[32] This pipeline is composed of three core methods (1-3 above), which are combined in a mutually reinforcing manner using a resampling approach. 1) Core method 1 uses SAM and differential expression analysis on the genetic knock-out data, and conditions that behave like knock-outs (which we refer to as pseudo knock-outs), to generate a ranked list of regulatory interactions. 2) Core method 2, time-lagged Context Likelihood of Relatedness, uses a mutual-information based algorithm, and an underlying linear ordinary differential equation model to infer a putative regulatory network. 3) Core method 3, Inferelator 1.0, uses an underlying linear ordinary differential equation model and l_1 constrained linear regression to refine the network from core method 2, and assigns dynamical parameters as well as topology. Resampling is used to generate many permutations of the input time-series and steady-state matrices, which results in an ensemble of putative networks inferred by core methods 2 and 3. Each network in the ensemble is then combined with the results of core method 1. The final network is the median confidence score of each edge from the ensemble.

to calculate raw differences in expression), and averaged across replicates (in order to reduce the noise of the measurement):

$$x_i^C = \sum_{j=1}^{r^C} 2^{X_{i,j}^C}/p, \qquad i = 1, \ldots, N$$
$$x_i^W = \sum_{j=1}^{r^W} 2^{X_{i,j}^W}/q, \qquad i = 1, \ldots, N$$

Next z_{diff}, the z-score of the log absolute difference in expression, and z_{fc}, the z-score of the fold change in expression, are calculated:

$$z_{diff} = \frac{d_1 - median(d_1)}{sd(d_1)}, where\, d_1 = log_2 \left| x^W - x^C \right|$$
$$z_{fc} = \frac{d_2 - median(d_2)}{sd(d_2)}, where\, d_2 = log_2 x^C - log_2 x^W$$

z-scores are combined using Stouffer's method and referred to as z_{comb}. Note an important difference between the two z-scores: high positive values of z_{diff} signify a large absolute change in expression in either direction, while the signed value of z_{fc} shows both magnitude and direction of fold change. Therefore $|z_{fc}|$ is used to combine z-scores. Negative values in z_{comb}, which stem from insignificant entries in z_{diff}, are set to zero, after which we can use the signs of values in z_{fc} to assign direction (over-/under-expression) to the z_{comb} values:

$$z_{comb} = max\left(\frac{z_{diff} + |z_{fc}|}{\sqrt{2}}, 0\right) \times sign(z_{fc})$$

The $diff\_exp$ method generated z-scores with positive values indicating relative over-expression and negative values indicating relative under-expression that are comparable to the z-scores assigned by SAM.

Pseudo Knock-Outs. We also inferred pseudo knock-out conditions: conditions that show a similar pattern of change in expression as the annotated knock-outs. In other words, these conditions show significant under-expression for a small number of transcription factors. To find these pseudo knock-outs, we constructed a new matrix X^C that was the union of sets X^{pt} and X^{flts}, *i.e.*, that contained all steady-state perturbations and the first and last time points of each time-series experiment. We also constructed the corresponding matrix X^W of wild-type conditions where each row contained the wild-type control for the corresponding row of X^C, or $median(X^C)$ if no wild-type control existed.

We then calculated, for each gene in each condition, the matrix P^Z of z-scores of the change in expression and the matrix P^Δ of absolute change in expression value. P^Z and P^Δ are defined element-wise as below. Note that the standard deviation and median for each gene were calculated for that gene across all conditions:

$$p_{i,j}^Z = (X_{i,j}^C - X_{i,j}^W)/\sigma(X_{1\ldots C,j}^C), \qquad i = 1, \ldots, C \quad j = 1, \ldots, N$$
$$p_{i,j}^\Delta = \left| X^C i, j - median(X_{1\ldots C,j}^C) \right|, \qquad i = 1, \ldots, C \quad j = 1, \ldots, N$$

We sorted P^Z such that the most under-expressed genes were at the top of the list, and P^Δ such that the genes

with the largest absolute change in expression were at the top of the list. We then recorded, for each condition, the genes which met a percentile threshold p in both lists. Finally, those conditions with a cutoff n or fewer genes were labeled as potential knock-out conditions for the genes meeting the percentile cutoffs.

We explored the effect of different p and n values using a grid search, evaluating different cutoff values by comparing the number of unique transcription factors annotated as pseudo knock-out genes and the fraction of known deletions selected as knocked-out genes in PKO conditions. p and n values were chosen to give the highest number of unique pseudo knock-out transcription factors and recall of known deleted genes with the lowest possible number of pseudo knock-outs (n) per condition. Once optimal thresholds were set and pseudo knock-outs determined, the pseudo knock-out conditions were analyzed using the same pipeline as the annotated deletions. The analysis of pseudo knock-outs was used to produce a matrix Z^{pko} analogous to the Z^{ko} matrix produced by analyzing annotated deletion conditions. We then used Z^{ko} and Z^{pko} to filter the set of genes for which to infer regulatory interactions, as in.[32]

Combining Results. After this filtration step we applied a resampling approach to core methods 2 and 3 to generate an ensemble of networks (in the form of lists ranked by confidence). For each network in the ensemble we combined the rankings with the rankings from core method 1 (KOs + PKOs) using a rank based approach, as in.[32,51] From this ensemble of networks we picked the median confidence of each putative regulatory interaction as the final confidence value, as in.[32] This final confidence metric includes support from each of the core methods.

Discussion. The core of this inference pipeline is the Inferelator 1.0,[11] which uses l_1 constrained regression to select parsimonious network models (as biological networks are known to be sparse). In order to improve the completeness of the network we combine the results of Inferelator 1.0 with those of tlCLR. We combine the two output networks in such a way that each method is given equal weight in the final network. To add more completeness to this network we add the network inferred by KOs + PKOs, again giving equal weight to both networks. Hence, in the final network, the edges with the highest rank will be those that are present in all three networks. There are relatively few such edges, thus the beginning of the ranked list contains those edges most likely to be correct. This is reflected in the different between AUPR scores and AUROC scores, as the former are strongly affected by the correctness of the top few predictions while the latter measures performance across the whole set of predictions.

We tested the efficacy of the PKO method by comparing the DREAM5 scores of inference runs performed with and without the PKOs. Inclusion of PKOs resulted in a gain of 2.4 points in overall score over the no-PKO run (+2.6/+2.1 for AUPR/AUROC). Networks 1 and 3 showed improvement in both AUPR and AUROC, but no improvement was seen in network 4; we are not sure at this time what accounts for the lack of improvement in network 4 beyond the general difficulty of inference on that network.

10.20 Meta 3 — Analyzing subsets of heterogeneous gene expression compendia to elucidate transcriptional regulatory networks

The provided expression compendia were created by combining many unrelated experiments, such as specific gene knock-outs, over-expressions, drug treatments, as time series or steady state measurements. In order to exploit the different types of information included in these mixed datasets, we applied three distinct approaches to different subsets of data, and then combined the results into a final prediction. This method will be described in more detail elsewhere.[67]

We employed SysGenSIM,[66,68] a simulation toolbox, to generate artificial transcriptional regulatory networks and to simulate gene-expression data for experiments similar to those in the provided compendia.[67] This allowed us to evaluate a variety of inference algorithms using the AUROC and AUPR. After thorough simulation studies we decided to use three different methods on distinct data subsets.

Perturbation-response analysis. We identified experiments concerning TF knockout or over-expressions. The goal was to identify the potential targets of strongly perturbed TFs. Similar to the approaches we employed in previous DREAM network inference challenges,[66,76] we calculated for all potential target genes the relative response to the perturbations (superscript indicates the perturbation):

$$R_{TF_i \to T_j} = \frac{T_j^{TF_i} - T_j^{WT}}{T_j^{WT}}$$

In addition we employed double knockout or over-expression. To gain more confidence in $TF_i \to T_j$ we calculated (when possible):

$$R_{TF_i \to T_j} = \frac{T_j^{TF_i,TF_k} - T_j^{WT}}{T_j^{WT}} - \frac{T_j^{TF_k} - T_j^{WT}}{T_j^{WT}}$$

The double knockout or over-expression can have extra information for $TF_i \to T_j$, however the response to the double perturbation might be explained by the effect of TF_k, so this must be subtracted. In some cases TF_k was

not perturbed alone and then we could not subtract its effect, accepting to make some mistakes as a trade off for also identifying real targets of TF_i. In case of triple knockouts, we proceeded in a similar way. S^1_{ij}(the confidence in $TF_i \rightarrow T_j$) was obtained by averaging the absolute $R_{TF_i \rightarrow T_j}$ values from single, double and triple knockout and over-expression experiments.

Partial correlation analysis. This was applied to steady state data (we only used the last time point of the time series), with the goal to exploit the expression correlation between a TF_i and its targets. We applied full order partial correlation[75] using the GeneNet R package.[v] S^2_{ij} is the absolute value of partial correlations ω_{TF_i,T_j}.

Co-deviation analysis. This was applied to chips with drug perturbations, and chips featuring non-TF single gene perturbations. The goal was to check if target T_j consistently makes large deviations in expression when TF_i makes large deviations. We first converted the gene expression values into z-scores. Then, for each transcription factor TF_i:

1) The data was split into two subsets: one group of observations D^H_i with $z_i > d$ (TF_i is 'high') and another other group of observations D^L_i with $z_i < -d$ (TF_i is 'low'); best results were obtained with $d = 0.5$,

2) To identify the potential targets T_j of TF_i, for each T_j a two-sided t-test was performed to check whether its mean in D^H_i is significantly different from its mean in D^L_i. S^3_{ij} is the absolute value of the t-statistic t_{ij} (test performed for T_j when datasets are formed based on deviations of TF_i).

Combining. The edge confidences S^1_{ij}, S^2_{ij} and S^3_{ij} were combined into a single confidence score by using a weighted mean of the ranks. The weights were obtained through an optimization process on simulated data.

Discussion. This method was among the top performers on the *in vivo* data, but had only average performance on the simulated data. These results reveal that the combination of different inference techniques is indeed useful, especially with heterogeneous compendia, provided that these data are correctly subdivided. Moreover, even if gene expression simulations are more and more realistic, it clearly emerges that inference methods having good performances on synthetic datasets cannot always be expected to obtain the same results on *in vivo* gene expression data. This emphasizes the need for realistic simulation, to generate data with properties similar to those observed in *in vivo* data, an issue in which we have put much effort when creating SysGenSIM.

[v] strimmerlab.org/software/genenet

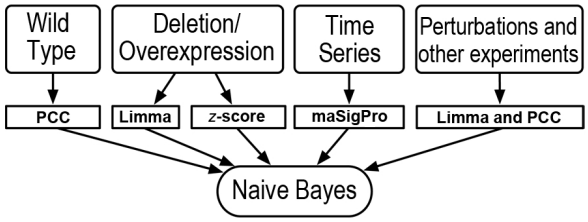

Figure S21: Inference method design for Meta 5. The organization of the data and the the algorithms that were used as input to the naïve Bayesian classifier.

10.21 Meta 5 – A naïve Bayes based approach to network inference

In this methodology we start by splitting the data according to the experiment type, then we analyze them according to specific statistical methods. Each method is used to evaluate the plausibility of the edges. Finally, to combine the statistics we follow a naïve Bayes approach.

Method. To analyze the data, the four statistics summarized below have been used:

- *Pearson correlation coefficient (PCC):* a standard method used to get a basic guess about the relationship between each pair of genes.
- *Limma:* a linear models approach to assess differential genes expression. This is a commonly used algorithm for analyzing experiments when the number of samples is limited.[80]
- *maSigPro:* a method for the analysis of experiments that include time series. In working with this tool a sequential experiment design matrix has been used.[20]
- *z-score:* the standard z statistic. We used it in experiments when other tools could not be applied.

We dealt with each type of experiment design using one (or two) of the above statistics. The statistic used for each kind of experiment is shown in Figure S10.21. In deletion experiments we put an edge between the deleted gene and all the significantly expressed genes. In the other experiments we find the clusters of co-expressed genes and put an edge between each transcription factor and all other genes. The combination of the statistics was evaluated using a naïve Bayes approach.

The goal is to compute the probability that an edge belongs to the network given the experimental evidence. Let us denote with X the variable that is equal to 1 when a given edge belongs to the network and to 0 otherwise. Let us also define with $Y_1 \dots Y_m$ the values of statistics assessing the given edge. We would like to compute the probability $P(X = 1|Y_1 \dots Y_m)$. By Bayes's theorem, this

quantity can be written as:

$$\begin{aligned} P(X=1|\ \ Y_1\ \ \ldots Y_m) & \\ &= \frac{P(Y_1 \cdots Y_m|X=1)P(X=1)}{\sum P(Y_i \cdots Y_m|X=x)P(X=x)} \\ &= \frac{\prod P(Y_i|X=1)P(X=1)}{\sum \prod P(Y_i|X=x)P(X=x)} \end{aligned}$$

where the equality holds by assuming independence between the statistic values. In order to compute the given formula, we specify the probabilities for $P(X=1)$ and $P(X=0)$. We evaluated them by exploiting the fact that in scale free networks: $k^{-\gamma}$ is the probability that a randomly chosen node has exactly k edges (where $\gamma \in (2,\ldots,3)$). Here, as suggested by Barabasi and Albert,[7] we set $\gamma = 3$. It follows that the number of edges $e(N)$ in a scale free network of size N can be computed as: $e(N) = \sum_{k=1}^{N} N \times P(k) \times k = N \sum_{k=1}^{N} \frac{1}{k^2}$. We approximate the quantity $\sum_{k=1}^{N} \frac{1}{k^2}$ with $\frac{\pi^2}{6}$ (its limit for $N \to \inf$). The *apriori* probability $P(X=1)$ of picking up an edge belonging to a network of size N is given by $\frac{e(N)}{N^2} = \frac{\pi^2}{6N}$. Finally, each of the $P(Y_i|X=x)$ distributions has been manually set by taking into consideration the characteristics of each statistic.[87]

Discussion. Gene network reverse engineering is a major challenge in computational biology and, the presented method is one of the simplest approaches that can be developed for a problem of this complexity. Indeed, simplicity has been one of the goals we strived to attain in its design. In fact, past DREAM contests emphasized that simpler methods could perform as well as others. Also, when *in vivo* networks are to be analyzed, data scarcity and its quality demand for classifiers built using a small number of well understood parameters. This method showed average performances in the DREAM contest.

An interesting facet of this methodology is that it performed remarkably better in the case of *in vivo* networks than with *in silico* ones: it is among the top performers (it ranks third) when the *in silico* dataset is not considered (it ranks 15^{th} otherwise). A number of interesting questions could be raised by this observation: what is in synthetic datasets that set them apart from natural ones? Should one strive to optimize new algorithm more aggressively on natural dataset? Could the culprit be found in the quality of *in vivo* data, so that most of these methods will perform much better when this quality increases? We believe that the answers to these questions may help in better understanding current tools and in developing new ones.

References

[1] D. Abdulrehman, P. T. Monteiro, M. C. Teixeira, N. P. Mira, A. B. Lourenco, S. C. dos Santos, T. R. Cabrito, A. P. Francisco, S. C. Madeira, R. S. Aires, A. L. Oliveira, I. Sa-Correia, and A. T. Freitas. Yeastract: providing a programmatic access to curated transcriptional regulatory associations in *saccharomyces cerevisiae* through a web services interface. *Nucleic Acids Res*, 39:136–140, 2010.

[2] A. Ahmed and E. P. Xing. Recovering time-varying networks of dependencies in social and biological studies. *PNAS of USA*, 106(29):11878–11883, 2009.

[3] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification. part i: Algorithm and empirical evaluation. *Journal of Machine Learning Research*, 11:171–234, 2010.

[4] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification. part ii: Analysis and extensions. *Journal of Machine Learning Research*, 11:235–284, 2010.

[5] C. Arndt. *Information Measures: Information and its description in Science and Engineering*. Springer, Berlin, 2001.

[6] T. Baba, T. Ara, M. Hasegawa, Y. Takai, Y. Okumura, M. Baba, K. Datsenko, M. Tomita, B. Wanner, and H. Mori. Construction of *escherichia coli* k-12 in-frame, single-gene knockout mutants: the keio collection. *Molecular Systems Biology*, 2(2006.0008), 2006.

[7] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[8] J. Beirlant, E. Dudewicz, L. Gyorfi, and E. van der Meulen. Nonparametric entropy estimation: An overview. *Int J Math Stat Sci*, 6(1):17–39, 1997.

[9] B. M. Bolstad, R. A. Irizarry, M. Astrand, and T. P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.

[10] R. Bonneau, M. T. Facciotti, D. J. Reiss, A. K. Schmid, M. Pan, and *et al.* A predictive model for transcriptional control of physiology in a free living cell. *Cell*, 131:1354–1365, 2007.

[11] R. Bonneau, D. J. Reiss, P. Shannon, M. T. Facciotti, L. Hood, and *et al.* The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets. *Genome Biology*, 7, 2006.

[12] J. C. Borda. Memoire sur les elections au scrutin, 1781.

[13] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[14] L. Breiman, J. H. Friedman, R. A. Olsen, , and C. J. Stone. *Classification and Regression Trees*. Chapman & Hall, 1984.

[15] A. Brooun, J. J. Tomashek, and K. Lewis. Purification and ligand binding of emrr, a regulator of a multidrug transporter. *J. Bacteriol.*, 181(16):5131–5133, 1999.

[16] N. A. Burton, M. D. Johnson, P. Antczak, A. Robinson, and P. A. Lund. Novel aspects of the acid response network of *e. coli* k-12 are revealed by a study of transcriptional dynamics. *J. Mol. Biol.*, 401(5):726–742, 2010.

[17] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[18] K. Y. Choi and H. Zalkin. Regulation of *escherichia coli* pyrc by the purine regulon repressor protein. *J. Bacteriol.*, 172(6):3201–3207, 1990.

[19] J. Cohen. Eta-squared and partial eta-squared in fixed factor anova designs. *Educational and Psychological Measurement*, 33(1):107, 1973.

[20] A. Conesa, M. J. Nueda, A. Ferrer, and M. Talon. maSigPro: a method to identify significantly differential expression profiles in time-course microarray experiments. *Bioinformatics*, 22(9):1096–1102, 2006.

[21] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, NY, 1991.

[22] C. O. Daub, R. Steuer, J. Selbig, and K. S. Estimating mutual information using b-spline functions–an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, 5:118, 2004.

[23] J. Davis and M. Goadrich. The relationship between Precision-Recall and ROC curves. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 233–240, Pittsburgh, Pennsylvania, 2006. ACM.

[24] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, volume 1857, pages 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[25] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Ann. Stat.*, 32(2):407–499, 2004.

[26] S. M. Egan and R. F. Schleif. A regulatory cascade in the induction of rhabad. *J. Mol. Biol.*, 234(1):87–98, 1993.

[27] J. J. Faith, M. E. Driscoll, V. A. Fusaro, E. J. Cosgrove, B. Hayete, F. S. Juhn, S. J. Schneider, and T. S. Gardner. Many microbe microarrays database: uniformly normalized affymetrix compendia with structured experimental metadata. *Nucleic Acids Res*, 36:866–870, 2008.

[28] J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, and *et al.* Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5:54–66, 2007.

[29] R. Foygel and M. Drton. Exact block-wise optimization in group lasso and sparse group lasso for linear regression. Technical report, University of Chicago, Department of Statistics, 2010. available at: http://arxiv.org/abs/1010.3320.

[30] J. H. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9:432–441, 2008.

[31] S. Gama-Castro, H. Salgado, M. Peralta-Gil, A. Santos-Zavaleta, L. Muniz-Rascado, H. Solano-Lira, V. Jimenez-Jacinto, V. Weiss, J. S. Garcia-Sotelo, A. Lopez-Fuentes,

L. Porron-Sotelo, S. Alquicira-Hernandez, A. Medina-Rivera, I. Martinez-Flores, K. Alquicira-Hernandez, R. Martinez-Adame, C. Bonavides-Martinez, J. Miranda-Rios, A. M. Huerta, A. Mendoza-Vargas, L. Collado-Torres, B. Taboada, L. Vega-Alvarado, M. Olvera, L. Olvera, R. Grande, E. Morett, and J. Collado-Vides. RegulonDB version 7.0: transcriptional regulation of escherichia coli k-12 integrated within genetic sensory response units (Gensor units). *Nucleic Acids Research*, 39(Database issue):D98–105, 2011.

[32] A. Greenfield, A. Madar, H. Ostrer, and R. Bonneau. Dream4: Combining genetic and dynamic information to identify biological networks and dynamical models. *PLoS one*, 5:e13397, 2010.

[33] C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, K. D. Macisaac, T. W. Danford, N. M. Hannett, J. Tagne, D. B. Reynolds, J. Yoo, E. G. Jennings, J. Zeitlinger, D. K. Pokholok, M. Kellis, P. A. Rolfe, K. T. Takusagawa, E. S. Lander, D. K. Gifford, E. Fraenkel, and R. A. Young. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431(7004):99–104, 2004.

[34] R. F. Hashimoto, E. Dougherty, M. Brun, Z. Zhou, M. L. Bittner, and *et al.* Efficient selection of feature sets possessing high coefficients of determination based on incremental determinations. *Signal Process*, 83(4):695–712, 2003.

[35] A. Haury, F. Mordelet, P. Vera-Licona, and J. Vert. TIGRESS: trustful inference of gene regulation using stability selection. *arXiv:1205.1181*, May 2012.

[36] B. He and H. Zalkin. Regulation of *escherichia coli* pura by purine repressor, one component of a dual control mechanism. *J. Bacteriol.*, 176(4):1009–1013, 1994.

[37] M. J. Herrgard, M. W. Covert, and B. O. Palsson. Reconciling gene expression data with known genome-scale regulatory network structures. *Genome Res.*, 13(11):2423–2334, 2003.

[38] F. Hommais, E. Krin, J. Y. Copp'ee, C. Lacroix, E. Yeramian, A. Danchin, and P. Bertin. Gade (yhie): a novel activator involved in the response to acid environment in *escherichia coli*. *Microbiology*, 150(1):61–72, 2004.

[39] Z. Hu, P. J. Killion, and V. R. Iyer. Genetic reconstruction of a functional transcriptional regulatory network. *Nat Genet*, 39:683–687, 2007.

[40] V. A. Huynh-Thu, A. Irrthum, L. Wehenkel, and P. Geurts. Inferring regulatory networks from expression data using tree-based methods. *PLoS one*, 5(9):e12776, 2010.

[41] E. Keedwell and A. Narayanan. Discovering gene networks with a neural-genetic hybrid. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 2:231–242, 2005.

[42] I. M. Keseler, J. Collado-Vides, A. Santos-Zavaleta, M. Peralta-Gil, S. Gama-Castro, L. MuÃśiz-Rascado, C. Bonavides-Martinez, S. Paley, M. Krummenacker, T. Altman, P. Kaipa, A. Spaulding, J. Pacheco, M. Latendresse, C. Fulcher, M. Sarker, A. G. Shearer, A. Mackie, I. Paulsen, R. P. Gunsalus, and P. D. Karp. EcoCyc: a comprehensive database of escherichia coli biology. *Nucleic Acids Research*, 39(Database issue):D583–590, 2011.

[43] P. Kheradpour, A. Stark, S. Roy, and M. Kellis. Reliable prediction of regulator targets using 12 drosophila genomes. *Genome Research*, 17(12):1919–1931, 2007.

[44] R. Küffner, T. Petri, L. Windhager, and R. Zimmer. Inferring gene regulatory networks by anova, In preparation.

[45] S. Lebre, J. Becq, F. Devaux, M. P. H. Stumpf, and G. Lelandais. Statistical inference of the time-varying structure of gene regulation networks. *BMC Systems Biology*, 4(1):130, 2010.

[46] S. Liang, S. Fuhrman, and R. Somogyi. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. In *Pacific Symposium on Biocomputing*, volume 3, pages 18–29, 1998.

[47] S. Lin. Rank aggregation methods, 2010.

[48] O. Lomovskaya, K. Lewis, and A. Matin. Emrr is a negative regulator of the *escherichia coli* multidrug resistance pump emrab. *J. Bacteriol.*, 177(9):2328–2334, 1995.

[49] F. M. Lopes, D. C. Martins, and R. M. Cesar. Feature selection environment for genomic applications. *BMC Bioinformatics*, 9:451, 2008.

[50] K. D. MacIsaac, T. Wang, D. B. Gordon, D. K. Gifford, G. D. Stormo, and E. Fraenkel. An improved map of conserved regulatory sites for saccharomyces cerevisiae. *BMC Bioinformatics*, 7:113, 2006.

[51] A. Madar, A. Greenfield, E. Vanden-eijnden, and R. Bonneau. Dream3: Network inference using dynamic context likelihood of relatedness and the inferelator. *PloS one*, 5(3):e9803, 2010.

[52] S. Mani, G. Cooper, and A. Statnikov. Causal discovery algorithms based on y structures, Under review.

[53] S. Mani and G. F. Cooper. A bayesian local causal discovery algorithm. In M. Fieschi and *et al.*, editors, *Proceedings of the World Congress on Medical Informatics*, pages 731–735, 2004.

[54] S. Mani, P. Spirtes, and G. Cooper. A theoretical study of y structures for causal discovery. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 314–323, 2006.

[55] D. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano, and G. Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences of the United States of America*, 107(14):6286–6291, 2010.

[56] D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano. Generating realistic *in silico* gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239, 2009.

[57] A. A. Margolin, K. Wang, W. K. Lim, M. Kustagi, I. Nemenman, and A. Califano. Reverse engineering cellular networks. *Nature Protocols*, 1:662–671, 2006.

[58] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34:1436–1462, 2006.

[59] N. Meinshausen and P. Bühlmann. Stability selection. *J. Royal. Statist. Soc. B.*, 72(4):417–473, 2010.

[60] T. Michoel, R. De Smet, A. Joshi, Y. Van de Peer, and K. Marchal. Comparative analysis of module-based versus direct methods for reverse-engineering transcriptional regulatory networks. *BMC Systems Biology*, 3(1):49, 2009.

[61] F. Mordelet and J. P. Vert. SIRENE: supervised inference of regulatory networks. *Bioinformatics*, 24:76–82, 2008.

[62] M. E. J. Newman. Modularity and community structure in networks. *PNAS*, 103(23):8577–8582, 2006.

[63] P. Novichkov, O. Laikova, E. Novichkova, M. Gelfand, A. Arkin, I. Dubchak, and D. Rodionov. Regprecise: a database of curated genomic inferences of transcriptional regulatory interactions in prokaryotes. *Nucleic Acids Research*, 38(Database issue):D111–118, 2010.

[64] J. Pearl. *Causality. Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, UK, 2009.

[65] M. W. Pfaffl. A new mathematical model for relative quantification in real-time rt-pcr. *Nucl. Acids Res.*, 29(9):e45, 2001.

[66] A. Pinna, N. Soranzo, and A. de la Fuente. From knockouts to networks: Establishing direct cause-effect relationships through graph analysis. *PLoS one*, 5(10):e12912, 2010.

[67] A. Pinna, N. Soranzo, V. De Leo, and A. de la Fuente. Elucidating transcriptional regulatory networks from heterogeneous gene-expression compendia, In preparation.

[68] A. Pinna, N. Soranzo, I. Hoeschele, and A. de la Fuente. Simulating system genetics data with SysGenSIM, In press.

[69] R. J. Prill, D. Marbach, J. Saez-Rodriguez, P. K. Sorger, L. G. Alexopoulos, and *et al.* Towards a rigorous assessment of systems biology models: the dream3 challenges. *PloS ONE*, page e9202, 2010.

[70] A. T. Puig, A. Wiesel, and A. O. Hero. A multidimensional shrinkage-thresholding operator. In *IEEE Workshop on Statistical Signal Processing*, pages 113–116, 2009.

[71] P. Qiu, A. J. Gentles, and S. K. Plevritis. Fast calculation of pairwise mutual information for gene regulatory network reconstruction. *Computer Methods and Programs in Biomedicine*, 94(2):177–180, 2009.

[72] J. R. Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[73] R. J. Rolfes and H. Zalkin. Purification of the *escherichia coli* purine regulon repressor and identification of corepressors. *J. Bacteriol.*, 172(10):5758–5766, 1990.

[74] Y. Saeys, I. Inza, and P. Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23:2507–2517, 2007.

[75] J. Schäfer and K. Strimmer. An empirical bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21:754–764, 2005.

[76] A. Scheinine, W. Mentzen, E. Pieroni, G. Fotia, F. Maggio, G. Mancosu, and A. de la Fuente. Inferring gene networks: Dream or nightmare? part 2: Challenges 4 and 5. *Annals of the New York Academy of Sciences*, 1158:287–301, 2009.

[77] C. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 1948.

[78] A. Sîrbu, H. J. Ruskin, and M. Crane. *Stages of Gene Regulatory Network Inference: the Evolutionary Algorithm Role*, chapter 27, pages 521–545. 2011.

[79] M. E. Smoot, K. Ono, J. Ruscheinski, P. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics (Oxford, England)*, 27(3):431–432, 2011.

[80] G. Smyth. limma: Linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor, Statistics for Biology and Health,*, pages 397–420. Springer, 2005.

[81] G. Stolovitzky, R. J. Prill, and A. Califano. Lessons from the DREAM2 challenges. *Ann N Y Acad Sci*, 1158:159–195, 2009.

[82] J. D. Storey. A direct approach to false discovery rates. *J. Royal. Statist. Soc. B*, 64:479–498, 2002.

[83] J. V. Stoyanov, J. L. Hobman, and N. L. Brown. Cuer (ybbi) of *escherichia coli* is a merr family regulator controlling expression of the copper exporter copa. *Molecular Microbiology*, 39(2):502–512, 2001.

[84] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc. B*, 58(1):267–288, 1996.

[85] V. G. Tusher, R. Tibshirani, and G. Chu. Signicance analysis of microarrays applied to the ionizing radiation response. *PNAS of USA*, 98:5116–21, 2001.

[86] A. Untergasser, H. Nijveen, X. Rao, T. Bisseling, R. Geurts, and J. A. M. Leunissen. Primer3plus, an enhanced web interface to primer3. *Nucl. Acids Res.*, 35(suppl 2):W71–W74, 2007.

[87] A. Visconti, R. Esposito, and F. Cordero. Tackling the dream challenge for gene regulatory networks reverse engineering. In *AI*IA 2011: Artificial Intelligence Around Man and Beyond, XIIth Internationall Conference of the Italian Association for Artificial Intelligence*, Palermo, Italy, 2011 (to appear).

[88] M. Wu and C. Chan. Learning transcriptional regulation on a genome scale: a theoretical analysis based on gene expression data. *Briefings in Bioinformatics*, 2011.

[89] A. Xiong, A. Gottman, C. Park, M. Baetens, S. Pandza, and A. Matin. The emrr protein represses the *escherichia coli* emrrab multidrug resistance operon by directly binding to its promoter region. *Antimicrobial Agents and Chemotherapy*, 44:2905–2907, 2000.

[90] K. Yamamoto and A. Ishihama. Transcriptional response of *escherichia coli* to external copper. *Molecular Microbiology*, 56(1):215–227, 2005.

[91] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. Royal. Statist. Soc. B*, 68(1):49–67, 2006.

[92] C. Zhu, K. J. Byers, R. P. McCord, Z. Shi, M. F. Berger, D. E. Newburger, K. Saulrieta, Z. Smith, M. V. Shah, M. Radhakrishnan, A. A. Philippakis, Y. Hu, F. De Masi, M. Pacek, A. Rolfs, T. Murthy, J. LaBaer, and M. L. Bulyk. High-resolution DNA-binding specificity analysis of yeast transcription factors. *Genome Research*, 19(4):556–566, 2009.

[93] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *J R Statist Soc B*, pages 301–320, 2005.