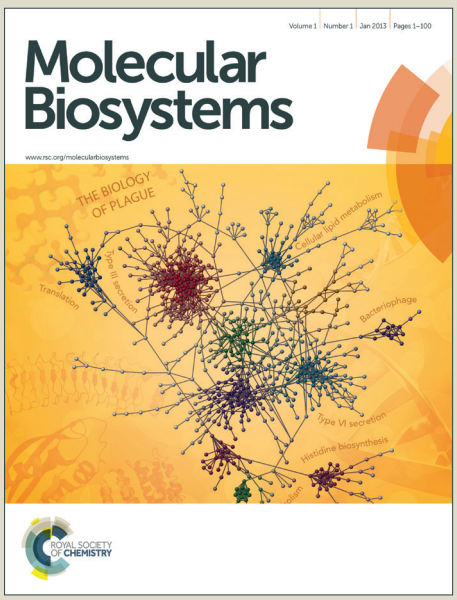
Molecular BioSystems

Accepted Manuscript



ROYAL SOCIETY OF CHEMISTRY

www.rsc.org/molecularbiosystems

Journal Name

ARTICLE TYPE

Cite this: DOI: 10.1039/xxxxxxxxxx

Improving Gene Regulatory Network Inference Using Network Topology Information†

Ajay Nair,*[abc] Madhu Chetty,[d] and Pramod P Wangikar[bef]

Received Date
Accepted Date

DOI: 10.1039/xxxxxxxxxx

www.rsc.org/journalname

Inferring the gene regulatory network (GRN) structure from data is an important problem in computational biology. However, it is a computationally complex problem and approximate methods such as heuristic search techniques, restriction of the maximum-number-of-parents (maxP) for a gene, or an optimal search under special conditions, are required. The limitations of a heuristic search are well known but literature on the detailed analysis of the widely used maxP technique is lacking. The optimal search methods require large computational time. We report the theoretical analysis and experimental results of the strengths and limitations of the maxP technique. Further, using an optimal search method, we combine the strengths of maxP technique and the known GRN topology to propose two novel algorithms. These algorithms are implemented in a Bayesian network framework and tested on biological, realistic, and *in silico* networks of different sizes and topologies. They overcome the limitations of the maxP technique and show superior computational speed when compared to the current optimal search algorithms.

1 Introduction

Inferring the gene regulatory network (GRN) structure from data is also known as GRN inference, top-down approach of GRN reconstruction, or reverse engineering of GRN. With the availability of high through-put DNA microarray data, many methods have been developed to infer the GRN [1–5]. Co-expression based methods [6] are simple, have low computational complexity, and are suited for large-scale networks; but they cannot infer causal interactions or model system dynamics. Models based on differential equations [7] are well established and can represent system dynamics accurately. However, they require detailed parameters from experiments, have high computational complexity, and have been mostly used for small-scale networks. Probabilistic graphical models such as Bayesian networks (BN) and dynamic Bayesian networks (DBN) fall in between the above methods in complexity and scale. They are based on the solid foundations of probability and statistics and are very popular since they can learn causality from data, integrate prior knowledge in modelling, be robust to noise in experimental data, and have an intuitive representation [8–11]. DBN can also model feedback loops using time series data.

Inferring graphical models from data is however computationally complex, since searching for the optimal graph from all the possible graphs is NP-hard with respect to the number of genes in the network [12,13]. Two common alternatives are using heuristic search techniques [14,15] or restricting the maximum-number-of-parents (maxP technique) for a gene during the search [16,17]. The drawbacks of the heuristic search methods are well known; they do not guarantee a global optima or the best network. In the case of maxP technique, the maximum number of transcription factors or regulators that can regulate a gene (denoted here by the name 'maxPval') is arbitrarily fixed. By using a small maxPval, computational complexity can be greatly reduced. However, from the knowledge of GRN topology and combinatorial regulation, it is well known that some genes are controlled by a large number of regulators [18,19]. Thus, using an arbitrary maxPval can prevent the representation of many regulatory interactions and affect the quality of inference. Further, the effect of maxP technique on GRN inference is not widely studied or reported, even though the method is widely used [16,17] and has also been suggested as an informative prior [20]. Although different methods of prior input [20] can reduce the computational complexity, the maxP technique is

[a] *IITB-Monash Research Academy, Indian Institute of Technology Bombay, Powai, Mumbai, 400076, India. E-mail: ajaynair@iitb.ac.in*
[b] *Department of Chemical Engineering, Indian Institute of Technology Bombay, Powai, Mumbai, 400076, India.*
[c] *Faculty of Information Technology, Monash University, Melbourne, Australia.*
[d] *Faculty of Science and Technology, Federation University, Victoria, Australia.*
[e] *DBT-Pan IIT Center for Bioenergy, Indian Institute of Technology Bombay, Powai, Mumbai, 400076, India.*
[f] *Wadhwani Research Center for Bioengineering, Indian Institute of Technology Bombay, Powai, Mumbai, 400076, India.*

† Electronic Supplementary Information (ESI) available: Theorems S1, S2, and S3 showing the number of network representations possible for directed graphs, decomposable scoring methods, and maxP technique respectively. See DOI: 10.1039/b000000x/

a generic method that does not require biological knowledge of the specific network or the organism being studied. It can also be combined with other methods, if required, to further improve the performance. Thus, this work focuses on the maxP technique.

Recently, optimal DBN structure learning algorithms for GRN inference have been proposed, namely BNFinder[21] and globalMIT[22]. They take advantage of the decomposable scoring functions and other mild assumptions valid for a GRN, in order to overcome the computational complexity of simple optimal search methods. Further, by being optimal in search, they overcome the uncertainties of heuristic search methods. These algorithms are currently suitable only for small networks since they need to search for all combinations of regulators for a gene. However, it may be possible to overcome this limitation if we can effectively use the knowledge that most genes in a GRN have very few regulators[18,19].

This paper has two main contributions. a) The theoretical and experimental analysis of the heuristic maxP technique to understand its strengths and limitations. The computational complexity of the maxP technique is compared with other inference techniques. This technique is also studied with different network topologies and parameters, and its theoretical complexity is compared with the actual inference time. b) The development of two novel algorithms that employ an optimal search method, but are based on the strengths of the maxP technique and are designed to take advantage of the known properties of the GRN topology. The hypothesis is that these steps help to reduce the exponential number of network searches required to find an optimal network in a GRN inference. The studies are carried out on 15 networks of varying sizes and data samples. The rest of the paper is organised as follows: section 2 develops the underlying theory; the experimental details are in section 3; the results and discussions are in section 4; and section 5 concludes the paper.

2 Theory

2.1 GRN topology

It is known a priori that the GRN structure has an exponential decrease in the indegree for both prokaryotes and eukaryotes[18,19]. Thus, most genes have only a few regulators and only a few genes have a large number of regulators. Our observations from the experimentally available GRNs of *Escherichia coli*[23], *Mycobacteria*[24], and *Bacillus subtilis*[25] also confirm this. Their median values of the indegree are 2, 1, and 1 respectively, while the maximum values of the indegree are 17, 11, and 16 respectively. Table 1 shows the percentage of genes that have an indegree of 1, 2, 3, and $\leq$3 (cumulative) in each of these organisms. It shows that 79% or more genes have ≤ 3 regulators in well known GRNs. Though our knowledge of the GRNs is not presently complete, it is known that many naturally occurring networks also show the property of exponential decrease in the indegree.

2.2 Computational complexity of GRN inference

The major challenge of GRN structure inference from data is the exponential search space of all possible graph structures. The complexity of the commonly used graph types for GRN inference

Table 1 Percentage of genes with the corresponding number of indegree in a few well studied organisms

indegree	% of genes		
	E. coli	*B. subtilis*	*Mycobacteria*
1	35	52	81
2	28	28	15
3	16	11	3
Cumulative	79	91	99

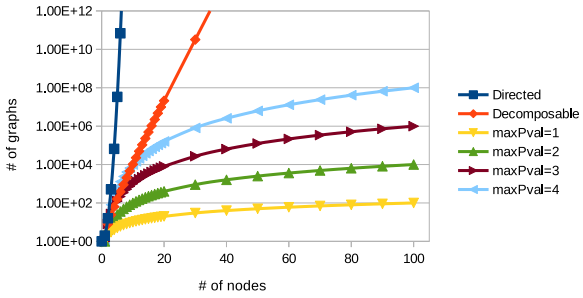


Fig. 1 Number of graphs to search for different number nodes in different types of networks

is discussed here. Note that, in graphical models the genes are known as nodes, the regulators of a gene are called its parents, the interaction between a regulator and a gene is represented by a directed edge from the regulator to its target gene in the graph. Thus, a GRN is represented as a directed graph.

For a directed graph representation, 2^{n^2} networks are possible for an n-node network (i.e., a network with n number of genes or nodes). The computational complexity of graph search in this case is super exponential in the number of nodes. Normal structure inference of a network using BN requires the inferred network to be acyclic which does not allow feedback loops in the network. However, when using networks with partial ordering of variables or a DBN algorithm, the acyclicity of a graph is not checked. Thus, a decomposable scoring function can be used, where a node and its parents can be independently inferred from rest of the nodes[26]. Thus, while using a decomposable scoring function on a directed graph with n-nodes, the number of network configurations required to be checked are $n2^n$. The computational complexity in this case is exponential. Further, for cases where the number of parents are restricted to a constant k (maxPval$=k$) for each node, the search space for an n-node graph has polynomial complexity of $O(n^k)$, for $k \ll n$. These three results are well known in the graph theory domain but since their references are not easily available, their proofs are provided in ESI†, Theorems S1, S2, and S3.

Figure 1 shows the number of graphs to search for a varying number of nodes, for the three different conditions: a) directed graph, b) directed graph with a decomposable scoring function, and c) directed graph with a decomposable scoring function and maxPval$=k$ limitation. It can be seen that the number of graphs to search reduces from a directed graph to a directed graph with

decomposable scoring function and maxPval limitation. The directed graph and the decomposable scoring graph show a linear curve on a semi-log scale plot. This means that their complexity of search is exponential which makes a scale-up to large networks inefficient. The maxPval$= k$ methods have saturation curves due to the polynomial complexity and are efficient to scale-up. For example, it is not feasible computationally to search for all the networks for a directed graph when the number of nodes is > 5. The computational feasibility of searching for all the possible graphs when using decomposable scoring functions are limited to 20–30 node networks[27]. The reason for these limitations can be explained as follows. Assuming one graph can be proposed and scored in one second, it will take a year to analyse $3.15E+07$ graphs. This is the approximate number of graphs possible with a 5-node directed network and a 20-node network with decomposable scoring (see figure 1). In contrast, all the possible graphs of a 100-node network with maxPval$= 2$ can be analysed in a day; from figure 1, a 100-node network with maxPval$= 2$ has $1.0E+04$ possible graphs, while $8.64E+04$ graphs can be searched in a day.

2.3 Optimal GRN inference algorithms

Reported optimal GRN inference algorithms such as globalMIT use decomposable scoring. However, they apply statistical techniques and mild assumptions valid for a GRN to bypass the exponential search spaces discussed above. For example, globalMIT assumes that all the genes have equal number of discrete gene expression states to reduce the computational complexity. It computes the statistical independence of nodes using the chi-squared test at user defined confidence level α, to penalise complex networks and prevent over-fitting. Readers interested in further details and derivations may refer to the original results[22,28]. By penalising complex networks, the algorithm keeps an upper bound on the number of parents a gene can have (the statistically significant number of parents), represented as $p*$, depending on the available data samples[22]. For example, with larger data samples, the $p*$ is also higher, since with larger data more interactions can be learned.

Thus, even though the algorithm is optimal, the whole search space of $n2^n$ possible graphs corresponding to the decomposable scoring functions need not be searched if the data samples are limited. This fact must be considered when comparing theoretical and experimental results.

2.4 maxP technique and improvement analysis

The computational complexity for a search with decomposable scoring is $n2^n$, which is a computationally difficult exponential complexity. However, in a search based on the maxP technique, the GRN inference is carried out by limiting the maximum number of parents for each gene to k (maxPval$= k$). The computational complexity here is n^k, which is a computationally efficient polynomial complexity.

The improvement of the maxP technique over the decomposable scoring method for the searched graph space is the ratio: $\frac{n^k}{n2^n} = n^{k-1}2^{-n}$. This ratio is the computational complexity improvement or the theoretical time performance improvement.

Table 2 Computational complexity improvement of the maxP technique over the normal decomposable scoring method for different values of maxPval$= k$ and n-gene networks

n	$k=1$	$k=2$	$k=3$
5	0.031	0.156	0.781
10	0.001	0.01	0.098
20	9.30E-007	1.90E-005	3.80E-004

The improvement for the various values of k and the number of genes in a network is given in Table 2. Note that for $n = 10$, only 0.1% of the original graph space is required to be searched by the maxP technique for $k = 1$. This reduction of the effective search space increases with larger n, which explains the popularity of the maxP technique. However, the limitation of this technique is that the number of regulators of a gene cannot be greater than a fixed value k and assigning a large value to k reduces the computational efficiency.

2.5 maxPiter algorithm and improvement analysis

The limitation of the maxP technique can be overcome by an iterative approach where, in the first iteration, all genes are inferred with maxPval$=k$. In the second iteration, genes that have exactly k regulators are selected (since these genes can potentially have more than k regulators) and the inference is repeated without any restriction on the number of regulators. We call this algorithm 'maxPiter' and the steps are shown in Algorithm 1. The maxPiter overcomes the disadvantage of maxP technique, which is the arbitrary limiting of the number of regulators of a gene, with the second iteration. Further, it retains the maxP technique's strength of reducing the search space, using the first iteration. Intuitively, maxPiter will be efficient for networks where most of the genes have a small number of regulators (since they will not be considered for the second iteration) and a few genes have large number of regulators, which is the characteristic of a GRN topology (discussed in section 2.1).

Algorithm 1 maxPiter

1: Infer network structure with maxPval $= k$
2: *selectNodes* $\leftarrow$ (set of nodes with # of parents = maxPval, in the inferred network)
3: Infer network structure for *selectNodes* with maxPval $\leq p*$ & update network with all nodes
4: Print the inferred network

Theorem 1. *The computational complexity of the maxPiter algorithm for an n-gene network with x-fraction of genes selected for the second iteration is* $n^k + xn2^n$.

Proof. The first iteration of the maxPiter algorithm is performed with maxPval$= k$ for all the n-genes. The computational complexity for this step is n^k, for $k \ll n$. In the second iteration, only the genes with a number of parents$= k$ (for instance, x-fraction of genes or xn-genes) are selected for the inference, while their parents can still be from any of the initial n-genes. Now there is

Table 3 Computational complexity improvement of the maxPiter over the normal decomposable scoring method for different values of maxPval$= k$ and n-gene networks assuming an *E. coli* like GRN

n	$k=1$	$k=2$	$k=3$
5	0.83	0.55	0.88
10	0.8	0.41	0.2
20	0.8	0.4	0.1

also no limitation on the number of parents for a gene. Thus, the number of possible parent combinations for a single gene is given by,

$$\sum_{i=k+1}^{n}\binom{n}{i} = \sum_{i=0}^{n}\binom{n}{i} - \sum_{i=0}^{k}\binom{n}{i}$$

$$= 2^n - \sum_{i=0}^{k}\binom{n}{i}$$

Since $k \ll n$ in the first iteration,

$$2^n - \sum_{i=0}^{k}\binom{n}{i} = O(2^n)$$

For xn-genes the number of possible parent combinations are $O(xn2^n)$ and the computational complexity of the second iteration becomes $xn2^n$. Thus, the overall computational complexity of maxPiter for the first and second iteration is $n^k + xn2^n$. □

The fraction of the graph space that maxPiter needs to search when compared to the decomposable scoring method is given by the ratio: $\frac{n^k+xn2^n}{n2^n}$. This ratio is also the computational complexity improvement or the theoretical time performance improvement of maxPiter. For the *E. coli* GRN given in Table 1, the approximate values of x for $k = \{1,2,3\}$ are $\{0.8, 0.4, 0.1\}$ respectively. Table 3 shows the computational complexity improvement for selected values of n and k for these values of x.

2.6 maxPincrement algorithm and improvement analysis

The disadvantage of maxPiter is that we first start with a maxPval$= k$ and since k should not be too large or too small, this choice seems critical to the improvement in time performance. Further, in the second iteration, once the nodes that have k parents are selected, inference is done for the full value of $p*$ though all of those nodes may not have a large number of parents due to the exponentially decreasing topology of the GRN. Thus, rather than limit the inference to only two-iterations, we can generalise the algorithm to have a maximum of $p*$-iterations and in each iteration the maxPval is incremented by 1. We call this algorithm the 'maxPincrement' and the steps are shown in Algorithm 2.

Algorithm 2 maxPincrement

1: Infer network structure with maxPval$= 1$
2: *selectNodes* $\leftarrow$ (set of nodes with # of parents = maxPval, in the inferred network)
3: **while** (*selectNodes* $\neq \emptyset$) and (maxPval$\leq p*$) **do**
4: maxPval:=maxPval+1
5: Infer network structure for *selectNodes* with current maxPval & update network
6: *selectNodes* $\leftarrow$ (set of nodes with # of parents = maxPval, in the inferred network)
7: Print the inferred network

Theorem 2. *The computational complexity of the maxPincrement algorithm for an n-gene network with x_{i-1} fraction of genes being selected for the i^{th} iteration is $n^1 + x_1 n^2 + .. + x_{p-1} n^p$.*

Proof. For cases where the number of parents are restricted to a constant k for each node, the number of graphs that needs to be searched for an n-gene network is given by $O(n^k)$, for $k \ll n$. Thus, for the first iteration, when the maxPval$=1$ and all the n-genes are inferred, the computational complexity is n^1.

When only x fraction of genes (xn-genes) are required to be inferred but the original n-genes need to be searched for k possible parents, then the number of graphs G to be searched is given by

$$G = O(xn \cdot n^k)$$

$$= O(x \cdot n^{k+1})$$

$$= O(xn^k)$$

Thus, in general, the computational complexity of inference for x fraction of genes with k possible parents in an n-gene network is xn^k. Then, for the second iteration when x_1 fraction of genes are selected and number of possible parents are 2, the complexity is $x_1 n^2$. Similarly, for i^{th} iteration the complexity becomes $x_{i-1} n^i$. Thus, the computational complexity of the maxPincrement algorithm is: $n^1 + x_1 n^2 + .. + x_{p-1} n^p$, where x_{i-1} is the fraction of nodes that have i number of parents in the i^{th}-iteration and p is the final iteration number.

□

The computational complexity improvement or the theoretical time performance improvement of maxPincrement over a decomposable score method is the ratio: $\frac{n^1+x_1 n^2+..+x_{p-1}n^p}{n2^n}$. The comparison of the maxPincrement algorithm to the decomposable scoring method is given in Table 4. The calculations are done by assuming an *E. coli* like GRN shown in Table 1, in which approximately 80%, 40%, and 10% of the total genes need to be checked for more than 1, 2, and 3 regulators respectively. When compared with Table 3, it can be seen that maxPincrement gives better computational complexity improvement than maxPiter. Thus, by inferring with only the required number of regulators for each gene, this algorithm uses the knowledge of GRN topology more efficiently than maxPiter. Further, since the indegree of a GRN decreases exponentially, x_i should also decrease exponentially as i increases. Thus, for a typical GRN very few genes will be inferred with a larger value of maxPval, which will result in improvement of the computational time.

Table 4 Computational complexity improvement given by maxPincrement over the normal decomposable scoring method for maxPval=1 and n-gene networks assuming an *E. coli* like GRN

n	maxPval=1
5	0.862
10	0.146
20	0.0009

3 Experiment

3.1 Software and firmware details

GRN inference is carried out using the modified code of an optimal DBN learning algorithm — the globalMIT. There are two optimal inference algorithms available in the public domain - BNFinder and globalMIT. Both of them are based on the theoretical foundations developed by[26], use decomposable scoring, give similar inference results, and differ only in their scoring metrics. The algorithms implemented here require only a decomposable scoring method and are independent of the scoring metrics. We selected globalMIT since it has source codes in both MATLAB and C++, which allows flexibility in algorithm development. Since in this study we compare the performance of novel algorithms to the original algorithm, the results given are independent of the software platforms and hardware used.

All the simulations are carried out on Intel Core i7-3770@3.4GHz 8-core CPU with 12GB RAM and 32-bit software.

3.2 Algorithms

Four types of GRN inference algorithms are implemented.

1. Normal: The GRN inference is carried out using an existing optimal GRN inference algorithm, the globalMIT, which has a decomposable scoring function. The results of this inference are used as the benchmark for the time performance and the quality of inference, to compare the maxP, maxPiter, and maxPincrement algorithms.

2. maxP: The maxP technique for maxPval= k is implemented by limiting the number of regulators for each gene to the user-input k. By default globalMIT checks for the statistically significant number of regulators $p*$, depending on the amount of input data available (for details see section 2.3). The code is modified such that maxPval= $min(k, p*)$.

3. maxPiter: The first iteration is performed with the maxP algorithm for the user-input of maxPval= k, developed above. Another modified code of globalMIT is used for the second iteration, where the inference is performed only on the genes with k regulators from the first iteration. Here, there is no maxPval limitation and the regulators of the selected genes are chosen from all the genes available in the original input data.

4. maxPincrement: The first iteration is performed with the maxP algorithm for maxPval= 1. In subsequent iterations, a modified code of globalMIT which is capable of accepting maxPval increments and an updated list of genes in each iteration are implemented. The inference is carried out only on the updated list of genes; the regulators for each gene are selected from all the genes in the original input data; and the maximum number of regulators for each gene is limited to the value of maxPval.

3.3 Networks

Inference studies are carried out on 15 different networks of sizes ranging from 5 to 20 genes and having different topologies. Networks of more than 20 genes are not considered since it is computationally infeasible to perform the 'normal' inference run (as discussed in section 2.2) and get the benchmark data. Three types of networks are considered here: a) biological networks - these are the real regulatory networks with actual experimental gene expression data, b) realistic networks - these are real regulatory networks with realistically simulated gene expression data, and c) *in silico* networks - these are computationally generated networks with simulated gene expression data.

The ultimate objective of any GRN inference technique is to study biological networks. Thus, the techniques should be assessed by their ability to reconstruct biological networks. However, there are many limitations for these networks[29,30] such as all the interactions may not be known, which makes it difficult to compute the accuracy of inference; the data will be limited compared to the number of genes being studied and thus, the inference problem is undetermined; and the data will also be noisy due to experimental measurement errors that also affect the accuracy of inference. These problems can be fixed with the realistic networks because their true structure is known, they have the topology of a GRN, and the quality as well as quantity of the data can also be controlled. The realistic networks in this study use subnetworks of known regulatory interactions in *E. coli* or yeast[29] and thus, have the topology of a real GRN. The dynamics of these networks are simulated with a detailed kinetic model of gene regulation using a system of ordinary differential equations and a standard thermodynamic approach[30]. These realistic kinetic models are used to generate the gene expression data corresponding to the different biological experiments.

The *in silico* networks are computationally generated networks. Since the biological and realistic networks have similar network topologies, because they are part of the real GRN, the *in silico* networks are used to obtain networks of varying topologies. For example, linear networks (sparsely connected networks), intermediate dense networks, and densely interconnected networks are constructed *in silico*. Of these the linear and highly dense topologies are not typically found in GRN. The details of the networks along with their abbreviations are given below, where for example, n5e6 indicates that the network has 5 genes and 6 edges.

3.3.1 Biological networks.

IRMA network—(n5e6)[11] and *E. coli* SOS network—(n8e7)[31] are the two well studied and widely used biological networks with experimental data. These networks are shown in figure 2. IRMA network is a 5-gene in-vivo synthetic network constructed in yeast *Saccharomyces cerevisiae*, as a gold standard biological

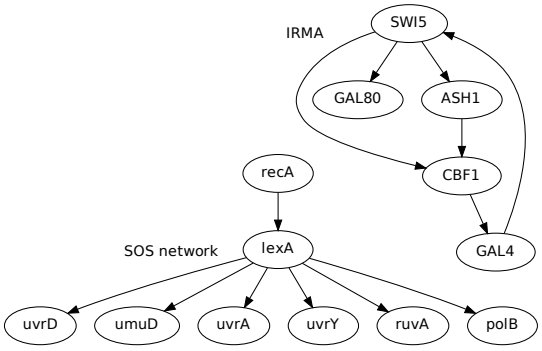


Fig. 2 Biological networks used in the study

network for assessing the GRN inference techniques. We used the two available time series gene expression data: a) 'switch-on' data corresponding to the shift of the organism from glucose to galactose-raffinose medium and b) 'switch off' data corresponding to a shift from galactose-raffinose to glucose medium. There are 5 experiments with 'switch-on' data and 4 experiments with 'switch-off' data with a total of 142 samples.

The *E. coli* SOS network is the other well studied biological network widely used to assess GRN inference techniques. However, compared to the IRMA network this is not a gold standard network which means all the possible interactions need not be known. Thus, classifying false-positives and true-negatives during inference need not be accurate but following the general practice, we proceed with the known network as the gold standard. The *E. coli* SOS network used here is based on the 8-gene network studied by Ronen et al.[31]. It contains 4 experiments on UV radiations with 50 time series measurements for each experiment.

3.3.2 Realistic networks.

We have used 6 realistic networks along with their time-series data. These networks are shown in figure 3. Of these, 2 networks are the widely used gold standard DREAM3 *E. coli* networks of 10-genes[29,30,32] which are referred here as DMn10e11 and DMn10e15. The time-series data for these networks corresponded to the normalised noisy measurements of mRNA expression levels and consists of 4 experiments with 21 time points each.

Since 20-gene gold standard networks are not available in DREAM or other sources, four 20-node realistic networks (referred here as GNWn20e50, GNWn20e33, GNWn20e31, and GNWn20e19) are generated using GeneNetWeaver-v3.1.1 Beta[29,33]. This software is used for generating realistic networks and their data for the DREAM competitions. These GNW networks are 20-node subnetworks of the original *E. coli* GRN that contains 1565 genes and 3758 edges, extracted using default parameters. After the extraction of the subnetworks, the gene names are anonymised, kinetic model is generated, and datasets are generated using default parameters. For the study, only the time-series data with 10 experiments and 21 time points each are used.

3.3.3 In-silico networks.

Both real biological networks and realistic networks have the typical GRN topology since they are subnetworks of the actual GRN. For analysing different types of network topologies, the *in silico* networks are constructed. They are of two sizes, 10-nodes (namely n10e9, n10e26, and n10e45, shown in figure 4) and 20-nodes (namely n20e19, n20e36, and n20e150, shown in figure 5) in order to make the comparison easier with the realistic networks of similar sizes. Unlike the biological or realistic networks used, these networks are designed to have three general topologies of sparse (having a linear topology; n10e9 and n20e19), intermediate dense (n10e26 and n20e36), and highly dense networks (n10e45 and n20e150). The time-series data for these networks are generated using forward sampling in the BNT toolbox[34] with the network parameters generated by random sampling from a uniform distribution. Data of 1000 and 2000 samples were generated for the 10 and 20-node networks respectively.

YUn20e9 is constructed with GeneSim[35] which is another popular gene regulatory network simulator that produces the gene expression levels at discrete time steps for an input network. The simulator uses a linear dynamical system of equations and requires the strength of gene-gene regulations in the input network. YUn20e9 network and data is provided with the globalMIT. The topology of this network is shown in figure 5.

3.3.4 Data used for the study.

The 15 networks and their number of data samples are given in Table 5. The real biological networks, DREAM3 networks, and GNW networks are inferred with all the available samples. For the *in silico* networks, the number of samples used are limited to sample size of DREAM3 experiments so that time performance and quality of inference can be compared across different networks. Only for specific studies on the effect of large samples (n10e9LS, n10e26LS, and n10e45LS), are all the available samples considered. Large sample studies are not performed on 20-gene networks due to their prohibitive computational complexity for 'normal' inference.

3.3.5 Indegree distribution.

Figure 6 shows the box plot of indegree of genes in all the 15 networks. In the vertical axis the *in silico* networks are at the top half, below are the realistic networks, and the two biological networks are at the bottom. It can be seen that the biological and realistic networks have a median indegree of 1 or 2, which is the characteristic of the GRN topology (see section 2.1). Moreover, the indegree distribution of the biological and realistic networks are not uniform. Some networks such as GNWn20e50 have the same value for the first quartile and the median, while other networks such as DMn10e15 have their third quartile value equal to the median. This is expected since the GRN is known to have an exponentially decreasing indegree distribution rather than a uniform distribution. However, the dense and intermediate dense *in silico* networks such as n20e150, n10e45, and n10e26 show a more uniform distribution of indegree and a larger median value of indegree. Further, sparse networks such as n10e9 and n20e19 do not show any variation in the indegree distribution. Thus,

overall the different networks selected show a range of variations in the topologies and indegree distribution, as required for the study.

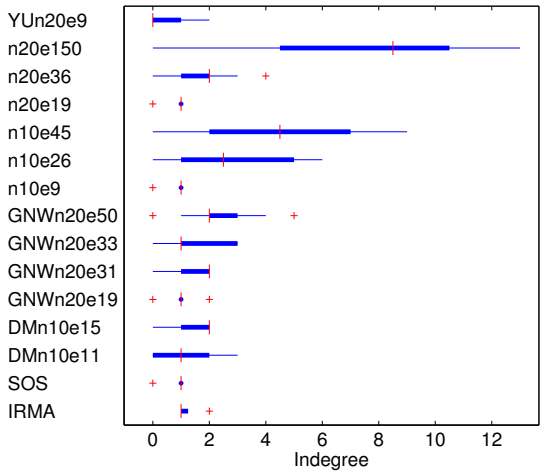


Fig. 6 The indegree distribution of genes in the different networks

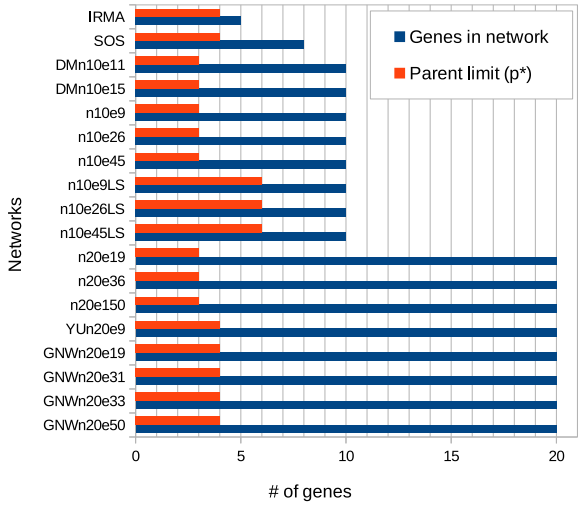


Fig. 7 Number of genes and the median number of parent limit ($p*$) for all the genes in each network for normal inference

3.4 Procedure

3.4.1 Performance measures.

The accuracy of inference or the inference performance is calculated using AUC-ROC (area under the ROC curve) and AUC-PR (area under the Precision-Recall curve) values using the results from different values of α, the variable for confidence level of network inference in globalMIT. The inference performance improvement or the 'fraction of normal inference measures' is the ratio of the inference performance of a particular algorithm to the inference performance for the benchmark condition.

The time performance is measured as the time taken for the algorithm to perform an inference at $\alpha = 0.999$, the recommended value for the best inference performance[28]. The time performance improvement or the 'fraction of normal time' is the ratio of time taken by a particular algorithm for inference to the time taken for inference in the benchmark condition.

3.4.2 Normal inference.

The normal method of inference is carried out with globalMIT for 13 different values of its confidence parameter $\alpha = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.85, 0.9, 0.95, 0.99, 0.999]$. From these results, the time and inference performance measures are calculated and used as a benchmark for other algorithms. The total number of inferences carried out are $13(\alpha\ s) * 15(networks) = 195$ inference runs.

3.4.3 maxP.

In this method, the number of regulators of each gene are limited by a fixed maxPval. The method is repeated for maxPval$= \{1,2,3,4\}$. maxPval> 4 is not considered since globalMIT's $p*$ is less than 5 for the majority of the networks (see figure 7). For each maxPval, the inference is carried out for all 13 different values of α. Again, the inference and time performance measures, and their performance improvements to the benchmark are calculated. The total number of inferences carried out are $4(\text{maxPvals}) * 13(\alpha\ s) * 15(networks) = 780$ inference runs.

3.4.4 maxPiter.

The inference procedure for the novel maxPiter algorithm is similar to maxP but the inference is performed twice corresponding to the two iterations. Thus, the total number of inferences carried out are $4(\text{maxPvals}) * 13(\alpha\ s) * 15(networks) * 2(iterations) = 1560$ inference runs.

3.4.5 maxPincrement.

In the novel maxPincrement algorithm, maxPval$= 1$ is set initially and the number of iterations are determined by the algorithm itself. Thus, the total inference runs can vary but assuming two iterations for each network, there are $13(\alpha\ s) * 15(networks) * 2(iterations) = 390$ inference runs

4 Results and discussion

4.1 Results for normal inference

Normal inference to obtain the benchmark values is carried out with the default values. Figure 7 shows the median $p*$ value for all the genes in each network (see section 2.3 for details on $p*$). For networks in which larger data samples are available, the $p*$ is also higher. It can be seen that only the IRMA network has $p*$ close to the number of genes, due to the large number of samples available compared to the network size. Thus, as discussed in section 2.3, the experimental results of only the IRMA network are expected to match the theoretical calculations.

4.2 Results with maxP

The maxP technique is a popular method of overcoming the computational complexity of the GRN structure inference. The objective of this study is to find the improvement in time performance and the quality of inference of this technique when compared to the normal algorithm. Further, the experimental results are compared with the theoretical calculations.

Table 6 Comparison of the theoretical and experimental time performance improvement for the 5-gene IRMA network with maxP

maxPval	Theoretical	Experimental
1	0.031	0.031
2	0.156	0.140
3	0.781	0.838

4.2.1 Time performance improvement.

Table 6 shows the comparison of the experimental time performance improvement of the 5-gene IRMA network to the theoretical calculations of a 5-gene network (given in Table 2). It can be seen that the theoretical calculations are correlated to the experimental results. Further, the theory (see section 2.4) also suggests that time performance is dependent only on the number of genes and the maxPval, and not on the connectivity of the original network (assuming that the effect of the sample size is negligible, for example, by keeping the sample size constant for networks of the same sizes). The experimental results of the time performance improvement analysis for the different networks shown in figure 8, validate this theory. It can be seen that for the networks with an equal number of genes and similar sample sizes (such as n10e9 and n10e45, YUn20e9 and n20e150), the time improvement is same even when the number of edges are very different.

It is also clear from figure 8 that the improvement is present only for cases where maxPval$< p*$, as expected. For example, in 10-gene networks with lower samples where $p* = 3$ (see fig 7), the time performance is good only till maxPval$= 2$ and for higher maxPvals, the time performance is same as the normal run. Further, for maxPval$< p*$ the fraction of normal time is almost always less than 20%, and in some cases negligible. Thus, the time performance improvement is very large for all the networks.

Figure 8 also shows the averaged improvement for the different types of networks: biological networks — AVG_BioNets (consisting of the IRMA and SOS networks); 10-gene networks — AVG_10 (consisting of DMn10e11, DMn10e15, n10e9, n10e26, and n10e45); 10-gene large-sample networks — AVG_10LS (n10e9LS, n10e26LS, and n10e45LS); 20-gene networks — AVG_20 (YUn20e9, n20e19, n20e36, n20e150, GNWn20e19, GNWn20e31, GNWn20e33, and GNWn20e50); and all the networks combined — AVG_all. It can be seen that AVG_10LS shows the best improvement since maxPval$< p*$ throughout.

4.2.2 Inference performance improvement.

The inference performance improvement for all the networks at different maxPvals is shown in figure 9 for AUC-ROC and AUC-PR. The networks within the histogram representation are ordered by decreasing performance. It can be seen that at maxPvals of 3 and 4, most of the networks show the same performance as the benchmark method since maxPval$\geq p*$. For maxPval$= 1$, more than 50% of the networks and for maxPval$= 2$ nearly half of the networks have inferior performance compared to normal inference since 'fraction of normal AUC-ROC/AUC-PR' < 1. These inferior performances are caused by the many missed regulator to gene interactions due to the low value of maxPval. However, non-intuitively, very few networks also show better performance at lower maxPvals. This is because some of the false-edges learned in normal method are now being discarded due to the maxPval restriction. This observation has also been reported previously[36].

Importantly, as seen in figure 9, there are three networks that show inferior performances for both AUC-ROC and AUC-PR: a) n20e150, b) n10e26LS, and c) n10e45LS. The reasons for their poor performance are:

- n20e150 has one of the highest average indegree value of 7.5 among all the networks considered here. Thus, by restricting the number of regulators to maxPval=1, 2, or 3, we artificially reduce the number of regulators that can be learned, which affects the inference performance.

- n10e26LS and n10e45LS have a large number of data samples to perform inference (see Table 5) and thus, can efficiently learn more complex networks. Again, by restricting the maxPval, we artificially reduce the number of edges being learned. This causes the inferior inference performance.

When the averaged performances in figure 9 are considered, even though the overall average, the AVG_all, is quite good, the AVG_10LS and the three networks discussed above consistently show inferior performance.

4.2.3 Discussion.

The theoretically expected improvement in time performance of the maxP technique is correlated to the actual experiments. Further, the time performance improvement, as suggested by the theory, is seen to be dependent only on the number of genes and maxPval, and not on the connectivity of the original network. The main advantage of the maxP technique is that it takes only a very small fraction of the normal inference run-time for low maxPval. Thereby, it achieves a large reduction in the computational time. However, the drawback is that the networks that have high connectivity or large data samples show inferior inference performance. This inferior performance is caused by the missed regulator-gene interactions, which is due to the low value of maxPval required to obtain the time performance improvement.

4.3 Results with maxPiter

The maxPiter is a novel algorithm proposed here. It performs an iterative inference and thus, keeps the strength of the maxP technique in reducing the computational time and overcomes the limitation of restricted maxPval.

4.3.1 Time performance improvement.

The comparison of the theoretical time performance improvement and the experimental run-time improvement for the 5-gene IRMA network is given in Table 7, for two different values of α. The column 'genesIter' shows the number of genes that are selected for the second iteration. It can be seen from the table that the experimental run-time improvements correlate well with the theoretical calculations.

The 'fraction of normal time' or the time performance improvement for all the networks and for different maxPvals is shown in

figure 10. The histograms are ordered by the increasing number of nodes in the networks. Within the networks of same number of nodes the biological, realistic, and *in silico* networks are separately clustered. The figure also shows the averaged fraction of time taken for the different network types. At maxPval=2 the AVG_all is 0.29 which means that the network inference can be completed in 29% of the normal inference time, on an average for all the different types of networks. Note also that the time improvement for maxPval=1 for most networks is inferior to maxPval=2. However, contrary to other networks, the improvement of AVG_20 for maxPval=2 is inferior to maxPval=3. These facts show that there is some optimum value for the maxPval that is specific to the different types of networks. This is because, values lower than optimum cause more genes to be selected for the second iteration and higher values cause many genes to be unnecessarily checked for a large number of parents in the first iteration. Both these situations increase the inference computation time.

4.3.2 Inference performance improvement.

The inference performance improvement of the maxPiter algorithm for the performance measures of AUC-ROC and AUC-PR, is shown in figure 11. It shows the inference performance improvement for individual networks and the averaged values of different network types. It can be seen that all the maxPvals show an inference performance very close or equal to the normal inference algorithm.

4.3.3 Discussion.

It is seen that the inference performance of maxPiter is almost the same as that of the normal inference algorithm, irrespective of the maxPval chosen. Thus, this algorithm overcomes the drawback of the maxP technique's inferior inference performance for highly connected networks or network inference with large data samples.

The experimental time performance improvement matches the theoretical prediction. It is also seen that the time performance of maxPiter varies with different maxPvals. The maxPval is chosen by the user for an inference. At maxPval=2, the network inference can be completed in 29% of the normal inference time on an average for all the different types of networks. Further, it is also seen that different types of networks can have different optimum values of the maxPval to improve the computational speed. Thus, the computational speed improvement is also dependent on the user's judgement of the optimum value of maxPval.

4.4 Results with maxPincrement

The maxPincrement is the second novel algorithm proposed here, which is designed to overcome the sensitivity of the maxPiter algorithm to the value of maxPval.

4.4.1 Time performance improvement.

The theoretical and actual time performance improvement for the 5-gene IRMA network is given in Table 8 for two different values of α. The column 'genesIter' shows the number of genes that are selected for the first, second, and subsequent iterations. It can be seen that the theoretical and experimental values correlate. It should be noted that the theoretical calculations do not consider the computational overhead of iterative function calls which will be reflected in the experimental values.

Table 8 Comparison of theoretical and actual time performance improvement of 5-gene IRMA network.

α	genesIter	theoretical	experimental
0.95	5, 3, 2	0.437	0.556
0.999	5, 3	0.125	0.27

The time performance improvement for each individual network and their averaged values for different types of networks are shown in figure 12. Most networks are inferred in less than 40% of the normal time. On an average, all the different networks (AVG_all) are inferred at 22% of the normal time. Larger networks take lesser time than average, which makes this method better at scaling. Theory also shows (see Table 4) that improvements are bigger for larger networks.

4.4.2 Inference performance improvement.

The inference performance improvements in the form of AUC-ROC and AUC-PR, are shown in figure 13. It can be seen that both the performance measures match very closely or equal the normal inference.

4.4.3 Discussion.

The maxPincrement algorithm is proposed as an alternative to maxPiter algorithm's dependence on the user-defined parameter of maxPval. The experimental improvements correlate well with the theoretical calculations. The maxPincrement shows a reduction of 7% in the inference run time when compared to the maxPiter algorithm and takes only 22% of the time for normal inference, on an average. The inference performance is same as that of the maxPiter algorithm and the normal inference. Further, the maxPincrement is also found to be better at scaling to larger networks.

5 Conclusion

The optimal inference of GRN structure using Bayesian networks is known to be NP-hard. One of the popular methods of reducing computational complexity is the restriction of the indegree of each gene — the maxP technique. However, detailed studies of this method have not been reported. Here we report the theoretical analysis and the detailed experimental studies of this method. The results show that the improvement of experimental run-times match the theoretically predicted improvements. Further, exponential improvements in run-times are achieved for lower maxPvals and the time performance improvement also does not depend on the network connectivity. However, the limitation of this method is that the quality of network inference deteriorates for densely connected networks and for networks with larger input data.

We proposed and developed two algorithms called 'maxPiter' and 'maxPincrement' based on the optimal GRN inference technique. They combine, a) the strength of maxP technique to reduce the computational time of inference and b) the knowledge

of GRN topology of the exponential decrease in indegree, to reduce the effective graph search space in inference. Thus, they have lower computational time than the existing optimal GRN inference methods. Further, the quality of the inferred network is equivalent to the existing inference technique for both these algorithms. For maxPiter at maxPval=2, the network inference can be performed at 29% of the original time on an average, for different types of networks. It may be noted that the time performance improvement of this algorithm is sensitive to the user-input value of maxPval for the different types of networks. The maxPincrement, which does not have such a drawback, is an improved alternative to maxPiter. On an average, it takes only 22% of the normal inference time and this improvement is higher for larger networks.

Scalability to larger networks is always difficult with optimal GRN inference methods due to the exponential increase in search space of all possible networks. However, the topological information of a typical GRN shows that most genes have only a small number of regulators and only a few genes have a large number of regulators. The algorithms proposed here take advantage of this and avoid checking for all possible parents for all the genes. Thus, the network search space is reduced. Note that the algorithms do not assume or require the networks to have a GRN topology. Results from the *in silico* networks show that the algorithms perform well on non-GRN network topologies also. Another advantage of these algorithms is that they do not need specific biological knowledge of the networks or the organism under study. If biological knowledge is available, other standard methods exist, which can be used along with the methods proposed here to further improve the performance.

Acknowledgement

AN thanks Jithin for the help in data collection and analysis of the GRN topology; Hairprasad and Shalu for their valuable suggestions.

Contribution

AN, MC, and PPW formulated the study. AN developed the algorithms, performed the experiments, and analysed the results. AN and MC wrote the paper. All the authors have approved the final manuscript.

References

1 H. de Jong, *Journal of Computational Biology*, 2002, **9**, 67–103.
2 W. Lee and W. Tzou, *Briefings in Bioinformatics*, 2009, **10**, 408–423.
3 R. De Smet and K. Marchal, *Nat Rev Micro*, 2010, **8**, 717–729.
4 D. Marbach, J. C. Costello, R. KÃijffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, The DREAM5 Consortium, M. Kellis, J. J. Collins and G. Stolovitzky, *Nature Methods*, 2012, **9**, 796–804.
5 Y. X. R. Wang and H. Huang, *Journal of Theoretical Biology*, 2014, **362**, 53–61.
6 M. B. Eisen, P. T. Spellman, P. O. Brown and D. Botstein, *Proceedings of the National Academy of Sciences*, 1998, **95**, 14863–14868.
7 T. S. Gardner, D. d. Bernardo, D. Lorenz and J. J. Collins, *Science*, 2003, **301**, 102–105.
8 N. Friedman, *Science*, 2004, **303**, 799–805.
9 T. S. Gardner and J. Faith, *Physics of Life Reviews*, 2005, **2**, 65–88.
10 M. Bansal, V. Belcastro, A. Ambesi-Impiombato and D. di Bernardo, *Molecular Systems Biology*, 2007, **3**, 78–88.
11 I. Cantone, L. Marucci, F. Iorio, M. A. Ricci, V. Belcastro, M. Bansal, S. Santini, M. di Bernardo, D. di Bernardo and M. P. Cosma, *Cell*, 2009, **137**, 172–181.
12 D. M. Chickering, Learning from Data: Artificial Intelligence and Statistics V, 1996, pp. 121–130.
13 D. M. Chickering, D. Heckerman and C. Meek, *J. Mach. Learn. Res.*, 2004, **5**, 1287–1330.
14 D. Heckerman, D. Geiger and D. M. Chickering, *Machine Learning*, 1995, **20**, 197–243.
15 R. Aghdam, M. Ganjali, X. Zhang and C. Eslahchi, *Molecular BioSystems*, 2015.
16 A. V. Werhli and D. Husmeier, *Statistical Applications in Genetics and Molecular Biology*, 2007, **6**, year.
17 D. Husmeier, *Bioinformatics*, 2003, **19**, 2271–2282.
18 R. Albert, *Journal of Cell Science*, 2005, **118**, 4947–4957.
19 A.-L. BarabÃąsi and Z. N. Oltvai, *Nature Reviews Genetics*, 2004, **5**, 101–113.
20 S. Mukherjee and T. P. Speed, *Proceedings of the National Academy of Sciences*, 2008, **105**, 14313–14318.
21 B. WilczyÅĎski and N. Dojer, *Bioinformatics*, 2009, **25**, 286–287.
22 N. X. Vinh, M. Chetty, R. Coppel and P. P. Wangikar, *Bioinformatics (Oxford, England)*, 2011, **27**, 2765–2766.
23 H. Salgado, M. Peralta-Gil, S. Gama-Castro, A. Santos-Zavaleta, L. MuÃśiz-Rascado, J. S. GarcÃ■a-Sotelo, V. Weiss, H. Solano-Lira, I. MartÃ■nez-Flores, A. Medina-Rivera, G. Salgado-Osorio, S. Alquicira-HernÃąndez, K. Alquicira-HernÃąndez, A. LÃşpez-Fuentes, L. PorrÃşn-Sotelo, A. M. Huerta, C. Bonavides-MartÃ■nez, Y. I. Balderas-MartÃ■nez, L. Pannier, M. Olvera, A. Labastida, V. JimÃĭnez-Jacinto, L. Vega-Alvarado, V. d. Moral-ChÃąvez, A. HernÃąndez-Alvarez, E. Morett and J. Collado-Vides, *Nucleic Acids Research*, 2013, **41**, D203–D213.
24 J. Sanz, J. Navarro, A. ArbuÃĭs, C. MartÃ■n, P. C. MarijuÃąn and Y. Moreno, *PLoS ONE*, 2011, **6**, e22178.
25 N. Sierro, Y. Makita, M. d. Hoon and K. Nakai, *Nucleic Acids Research*, 2008, **36**, D93–D96.
26 N. Dojer, Proceedings of the 31st International Conference on Mathematical Foundations of Computer Science, Berlin, Heidelberg, 2006, pp. 305–314.
27 S. Ott, S. Imoto and S. Miyano, *Pacific Symposium on Biocomputing.*, 2004, 557–567.
28 L. M. de Campos, *J. Mach. Learn. Res.*, 2006, **7**, 2149âĂŞ2187.
29 D. Marbach, T. Schaffter, C. Mattiussi and D. Floreano, *Journal of Computational Biology*, 2009, **16**, 229–239.

30 D. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano and G. Stolovitzky, *Proceedings of the National Academy of Sciences*, 2010.

31 M. Ronen, R. Rosenberg, B. I. Shraiman and U. Alon, *Proceedings of the National Academy of Sciences of the United States of America*, 2002, **99**, 10555–10560.

32 R. J. Prill, D. Marbach, J. Saez-Rodriguez, P. K. Sorger, L. G. Alexopoulos, X. Xue, N. D. Clarke, G. Altan-Bonnet and G. Stolovitzky, *PLoS ONE*, 2010, **5**, e9202.

33 T. Schaffter, D. Marbach and D. Floreano, *Bioinformatics*, 2011, **27**, 2263–2270.

34 K. P. Murphy, *Computing Science and Statistics*, 2001, **33**, 2001.

35 J. Yu, V. A. Smith, P. P. Wang, A. J. Hartemink and E. D. Jarvis, *Bioinformatics*, 2004, **20**, 3594–3603.

36 A. Nair, M. Chetty and P. P. Wangikar, *Neural Information Processing*, Springer International Publishing, 2014, vol. 8834, pp. 446–453.

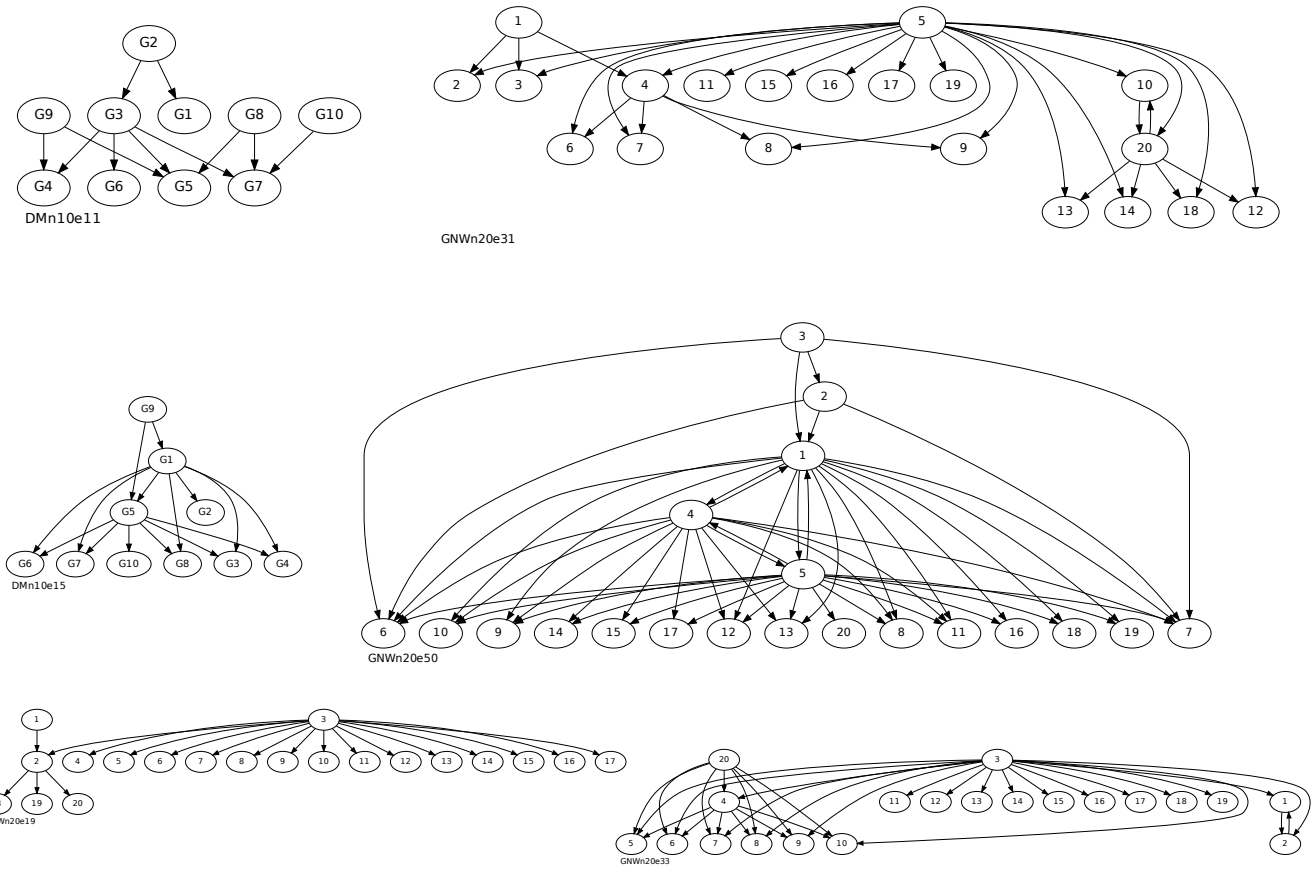


Fig. 3 The six realistic networks used in the study

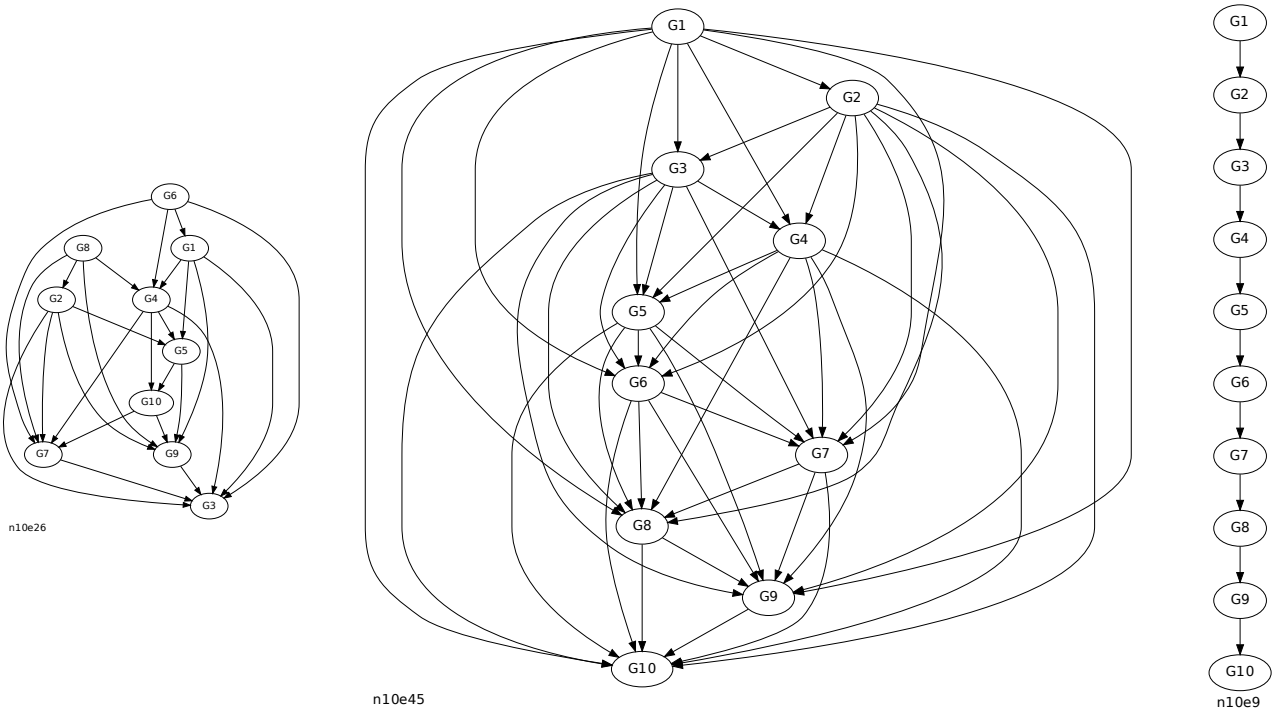


Fig. 4 The three 10-gene *in silico* networks used in the study

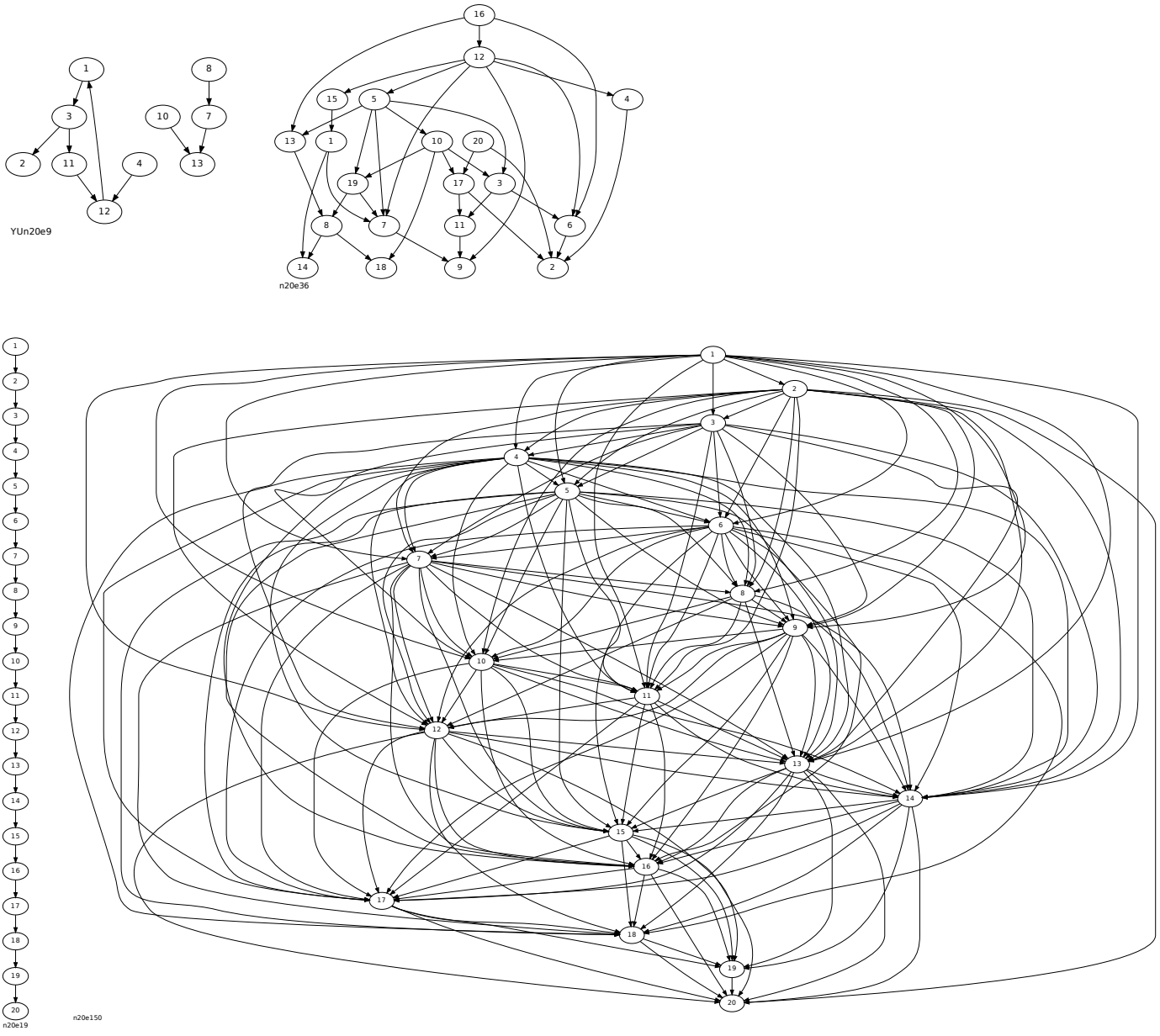


Fig. 5 The four 20-gene *in silico* networks used in the study

Table 5 The networks and the number of data samples used in the study.

Net	Samples	Experiments	Total samples	Samples used	Remarks
IRMA	-	9	142	142	Biological
SOS	50	4	200	200	Biological
DMn10e11	21	4	84	84	Realistic
DMn10e15	21	4	84	84	Realistic
GNWn20e50	21	10	210	210	Realistic
GNWn20e33	21	10	210	210	Realistic
GNWn20e31	21	10	210	210	Realistic
GNWn20e19	21	10	210	210	Realistic
n10e9	1000	1	1000	100	*In silico*
n10e26	1000	1	1000	100	*In silico*
n10e45	1000	1	1000	100	*In silico*
n10e9LS	1000	1	1000	1000	*In silico*
n10e26LS	1000	1	1000	1000	*In silico*
n10e45LS	1000	1	1000	1000	*In silico*
YUn20e9	2000	1	2000	200	*In silico*
n20e19	2000	1	2000	200	*In silico*
n20e36	2000	1	2000	200	*In silico*
n20e150	2000	1	2000	200	*In silico*

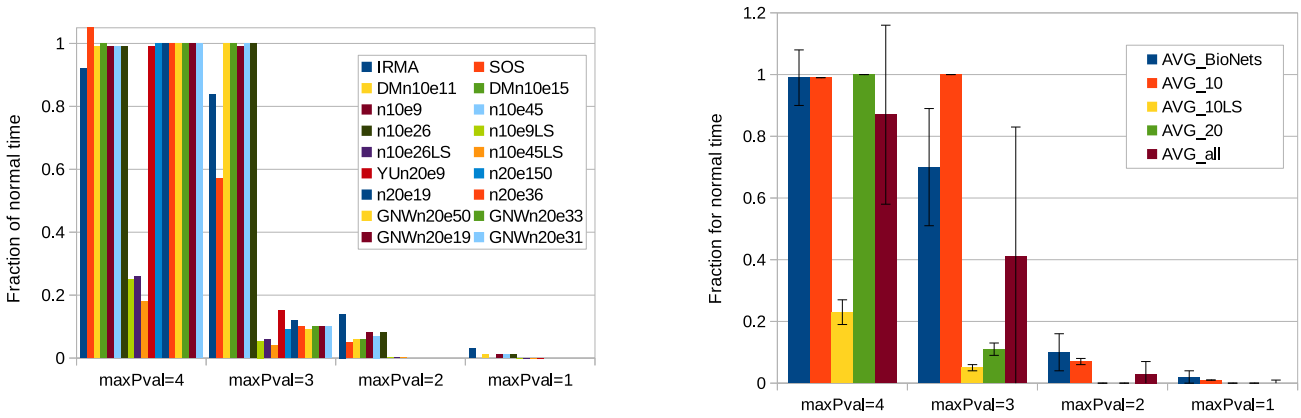


Fig. 8 The time performance improvement with maxP for all networks and their averages for different network types

Table 7 Comparison of theoretical and practical time performance improvement in the maxPiter algorithm for the IRMA network

maxPval	α=0.999			α=0.95		
	genesIter	theoretical	experimental	genesIter	theoretical	experimental
1	3	0.631	0.568	3	0.631	0.628
2	0	0.156	0.135	3	0.756	0.702

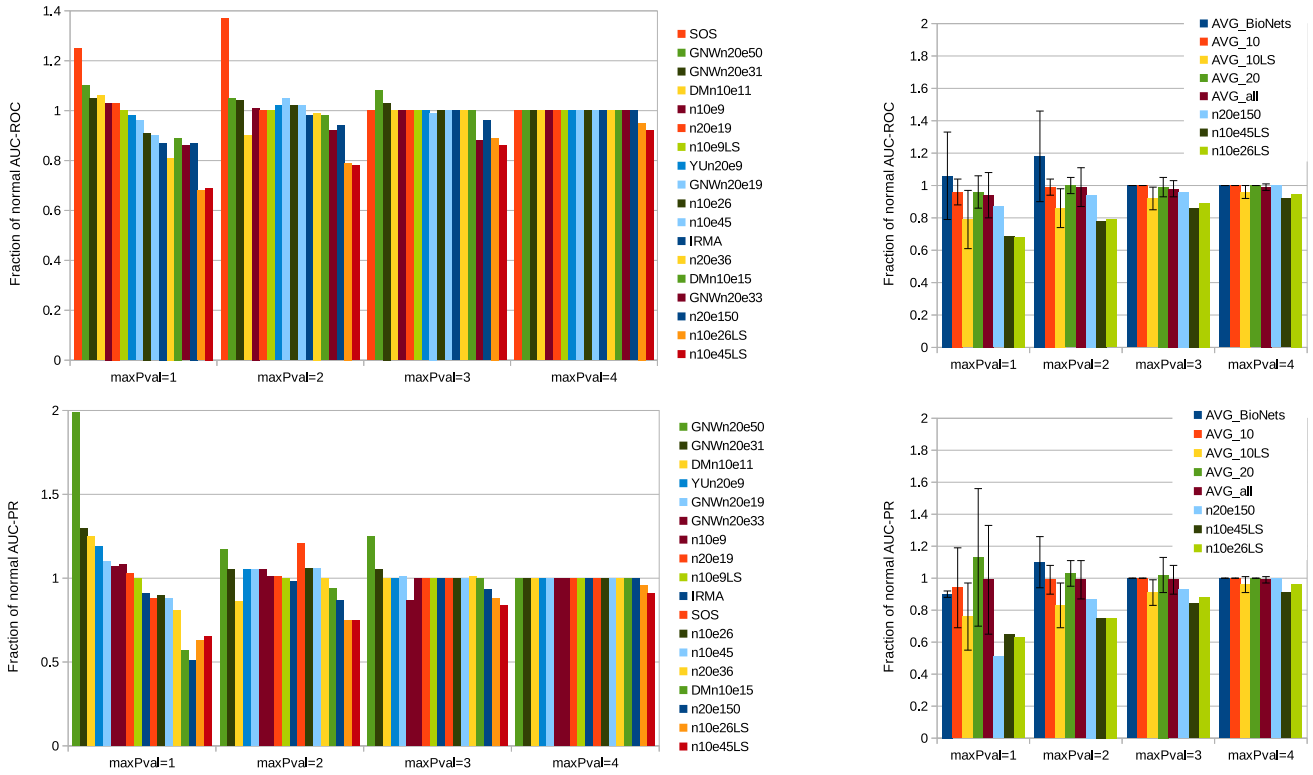


Fig. 9 The inference performance improvement for AUC-ROC and AUC-PR of all networks and their averages for different network types, with maxP

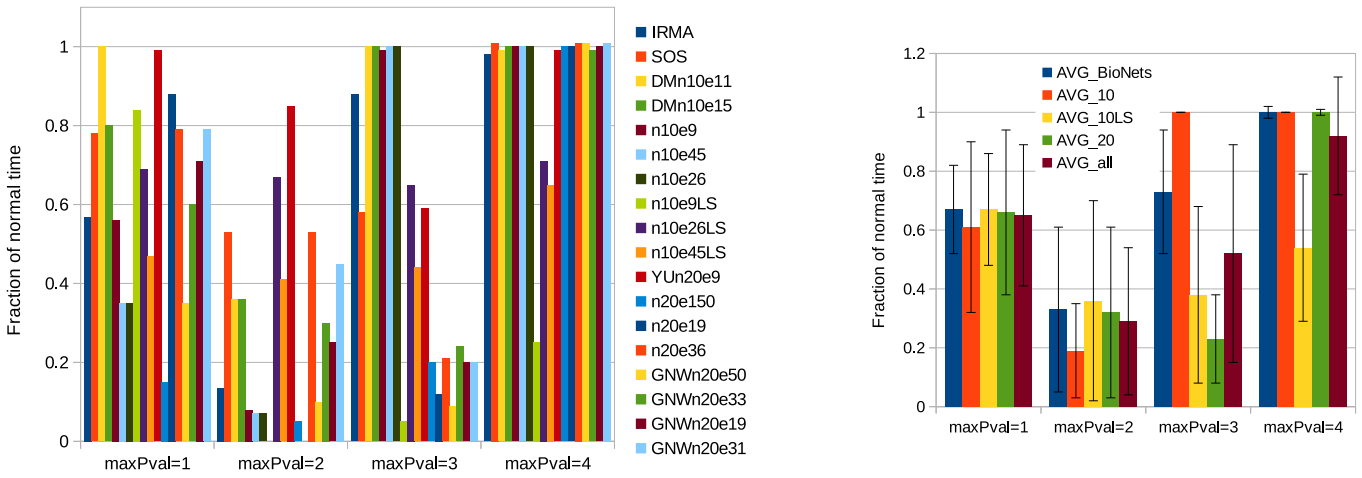


Fig. 10 The time performance improvement with maxPiter algorithm of all networks and their averages for different types of networks

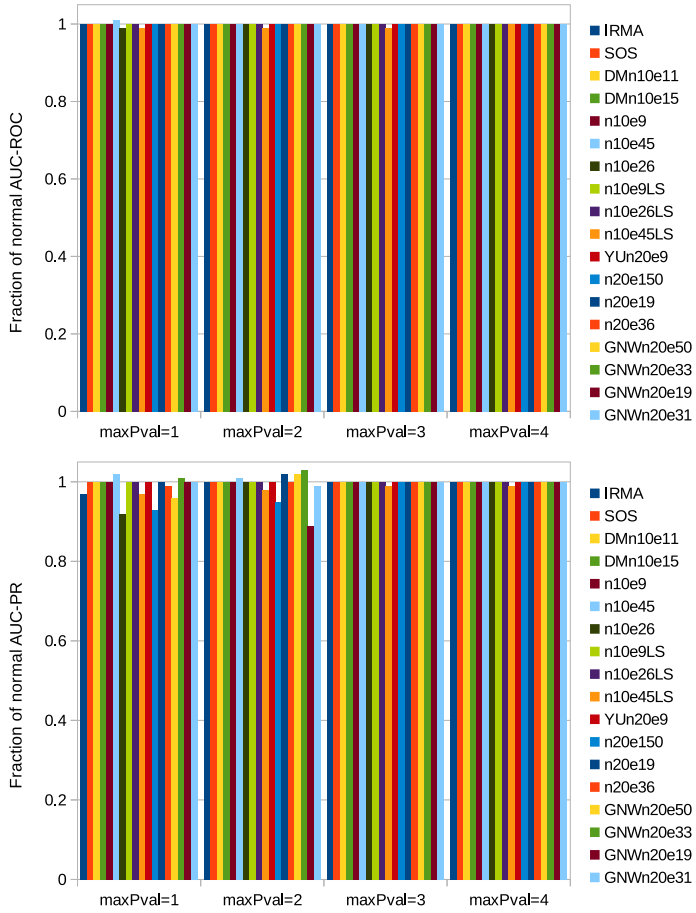


Fig. 11 The inference performance improvement for AUC-ROC and AUC-PR of all networks and their averages for different types of networks, with maxPiter algorithm

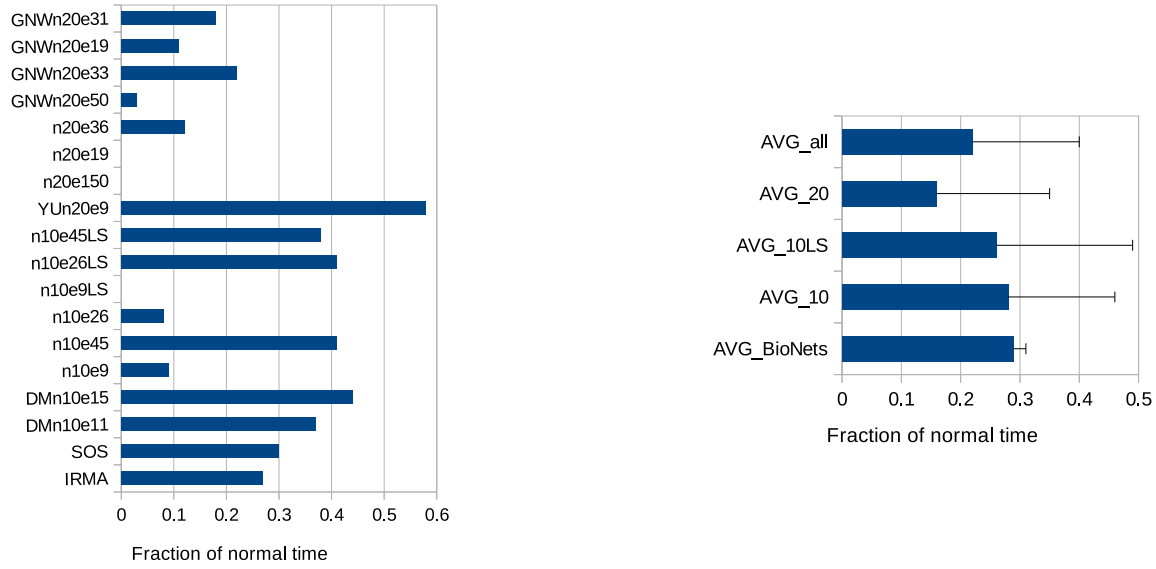


Fig. 12 The time performance improvement with maxPincrement algorithm of all networks and their averages for different types of networks

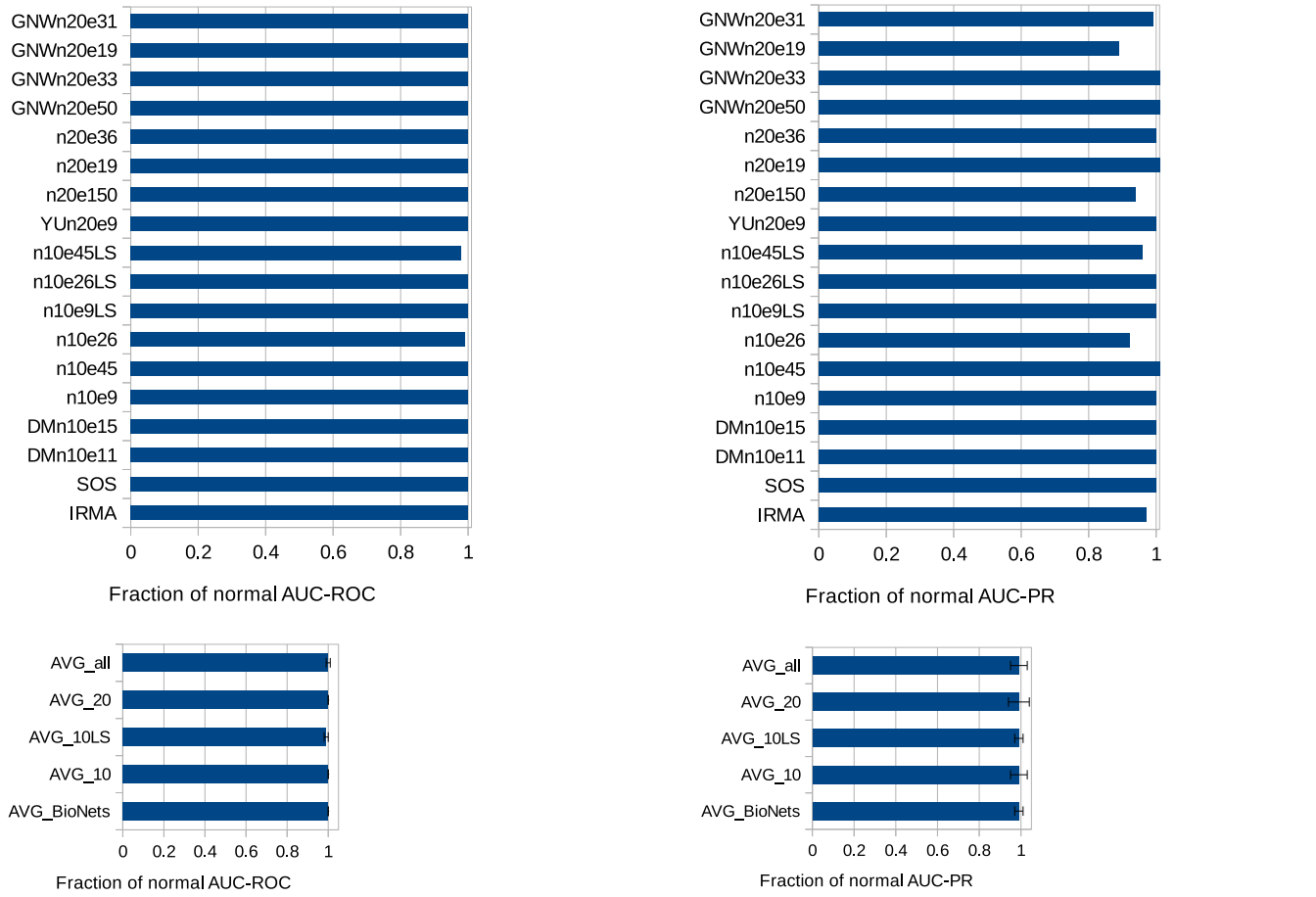


Fig. 13 The inference performance improvement for AUC-ROC and AUC-PR of all networks and their averages for different types of networks, with maxPincrement