Unsupervised GRN ensemble

Pau Bellot, Philippe Salembier, Ngoc C. Pham, and Patrick E. Meyer

Abstract Inferring gene regulatory networks from expression data is a very challenging problem that has raised the interest of the scientific community. Different algorithms have been proposed to try to solve this issue, but it has been shown that different methods have some particular biases and strengths, and none of them is the best across all types of data and datasets. As a result, the idea of aggregating various network inferences through a consensus mechanism naturally arises. In this chapter, a common framework to standardize already proposed consensus methods is presented, and based on this framework different proposals are introduced and analyzed in two different scenarios: Homogeneous and Heterogeneous. The first scenario reflects situations where the networks to be aggregated are rather similar because they are obtained with inference algorithms working on the same data, whereas the second scenario deals with very diverse networks because various sources of data are used to generate the individual networks. A procedure for combining multiple network inference algorithms is analyzed in a systematic way. The results show that there is a very significant difference between these two scenarios, and that the best way to combine networks in the Heterogeneous scenario is not the most commonly used. We show in particular that aggregation in the Heterogeneous scenario can be very beneficial if the individual networks are combined with our new proposed method ScaleLSum.

Pau Bellot
Centre for Research in Agricultural Genomics (CRAG) e-mail: pau.bellot@cragenomica.es

Philippe Salembier
Universitat Politecnica de Catalunya

Ngoc C. Pham
Bioinformatics and Systems Biology (BioSys) Unit

Patrick E. Meyer
Bioinformatics and Systems Biology (BioSys) Unit e-mail: Patrick.Meyer@ulg.ac.be

1 Introduction

Inferring gene regulatory networks from expression data is a very challenging problem that has seen a continuously rising interest in the last years and presumably in the years to come due to its applications in biomedical and biotechnological research.

Several studies have compared performances of network-inference algorithms [1, 2, 3, 4, 5, 6, 7, 8, 9], reaching the conclusion that none of the methods is the best across all types of data and datasets. Furthermore, the different algorithms have specific biases towards the recovery of different regulation patterns, i.e., Mutual Information (MI) and correlation based algorithms can recover feed-forward loops most reliably, while regression and Bayesian Networks can more accurately recover linear cascades than MI and correlation based algorithms [7, 10].

These observations suggest that different network-inference algorithms have different strengths and weaknesses [11]. Therefore, combining multiple network inference algorithms emerges as a natural idea to infer a more accurate gene regulatory network (GRN), leading to a consensus among *Homogeneous* networks. We use the term homogeneous here to refer to networks obtained from the same experimental data through the use of different algorithms. However, there are other situations where different networks describing the same cells are derived from different original data sets coming from very different technologies such as chip data, microarray, rnaseq and so on. We will refer to this situation as the *Heterogeneous* scenario.

Behind all state of the art consensus network algorithms one can identify a normalization step followed by an aggregation step. The main contributions of this chapter are 1) to systematically analyze the combination of normalization and aggregations strategies, creating in some cases new consensus network algorithms and 2) to evaluate the performances of the algorithms in both the *Homogeneous* and the *Heterogeneous* scenarios, and finally we will also present some alternatives to ensemble GRNs. As will be seen, the conclusions highlight a clear difference in terms of expected performances and algorithm selection in both cases. In order to precisely measure the algorithms performances and to control the degree of homogeneity without depending on any specific network inference algorithm, we rely on a synthetic network generation method presented in this paper. With this approach we can have a full control on the homogeneity and the characteristics of the individual networks.

2 State-of-the-art

GRN inference methods tend to recover indirect regulatory relationships. For example, if gene A regulates gene B and this last one regulates gene C, many algorithms will find a relationship between gene A and gene C even though it is an indirect effect.

Moreover, mixed regulatory interactions represent also a very difficult task. As for another example, if two genes D and E regulates gene F but with opposite effect (one activates the gene, and the other represses it). It is very likely to find any other gene that have a greater similarity to F's gene expression rather than D and E. Therefore, many methods will infer a false relationship instead of true ones. In this sense, [12] made an analysis of different network topologies (motifs) that are commonly inferred incorrectly. Furthermore, [13] pointed that some particular topologies are impossible to be inferred from gene expression data without any further information (knockdowns, existing interactions, etc).

Several network inference methods have been compared in several papers [14, 8, 9, 7]. The main conclusion of [9, 7] is that no single inference method performs optimally across all datasets and each GRN inference method returns a model that is different and has some strengths and biases.

From those observations comes the idea of combining multiple network inference algorithms. This could be a good strategy to infer an accurate and comprehensive GRN thanks to the meta-network algorithms that combine several methods. As a result, this approach is receiving more and more attention from the scientific community.

In order to create a consensus network from initial individual network inference algorithms, several strategies have been used. Assuming that each algorithm provides a score for each edge of the network, the simplest strategy consists of computing the average of the scores across the N individual networks [15].

Other strategies do not rely on the actual edge scores but on their rank defined in an ordered list. The method proposed in [7] is based on rank averaging: If e_{ij} denotes an edge connecting genes i and j and $r_n(e_{ij})$ the rank of the edge for network n, the final rank is computed with:

$$r(e_{ij}) = \sum_{1 \leq n \leq N} r_n(e_{ij}) \tag{1}$$

The consensus network is obtained from the list composed of the sorted values of summed ranks $r(e_{ij})$. This method called *RankSum* is also known as *Borda* count.

Top$k$Net [10] is based on the observations made in [7], which showed that integration of algorithms with high-diversity outperform the integration of algorithms with low-diversity [7]. First, for each individual network the predictions are ranked, and then the final rank for each edge is the result of applying a rank filter of order k, which returns the kth-greatest value (RankOrderFilter_k) over all the N values:

$$r(e_{ij}) = \text{RankOrderFilter}_k \left\{ r_1(e_{ij}), \ldots, r_N(e_{ij}) \right\} \tag{2}$$

The computation of a rank value in the *RankSum* and the Top$k$Net algorithm can be viewed as a normalization of the score values before their aggregation with the sum or the rank order filter. Based on this observation, we present an analysis of potential normalization and aggregation strategies in the following section.

We argue that this strategy is the most general one, since the networks may come from different kind of data. Furthermore, this strategy can be even applied when the networks are constructed from chip data or even from the literature.

2.1 Systematic consensus analysis

As previously mentioned, the consensus network estimation can be seen as involving two distinct steps. The first one transforms the different network scores, $s_n(e_{ij})$, in order to have a common scale or distribution. This process will be referred to as "Normalization". For example, [7, 10] use *Rank*, whereas [15] does not perform any normalization which can be seen as the *Identity* normalization.

Then, the second step is the "Aggregation" of the N different edge scores into one consensus score for every edge. In [7] and [15] this process is done through the *Sum* process, while [10] uses the rank order filter.

We now extend this idea to propose new algorithms. First, different Normalization options are presented, and then the distinct Aggregation proposals are discussed. Finally, their combination will lead to different consensus network algorithm proposals.

2.1.1 Normalization

Five different normalization techniques will be analyzed. Let us call $t_n(e_{ij})$ the normalized value assigned to edge e_{ij} for the network n.

Identity

The *Identity* does not apply any transformation to the original scores of the inferred network:

$$t_n(e_{ij}) = s_n(e_{ij}) \tag{3}$$

Rank

The *Rank* replaces the numerical score $s_n(e_{ij})$ by their rank $r_n(e_{ij})$ such as the most confident edge receives the highest score. The *Rank* method preserves the ordering of the scores of inferred links but the differences between them are lost:

$$t_n(e_{ij}) = r_n(e_{ij}) \tag{4}$$

Scale

A classical normalization of a random variable involves a transformation to remove the effect of the mean value and to scale it accordingly to its standard deviation. The differences between scores are preserved:

$$t_n(e_{ij}) = \frac{s_n(e_{ij}) - \mu_n}{\sigma_n} \tag{5}$$

where μ_n and σ_n are respectively the mean and standard deviation of the empirical distribution of the inferred scores $s_n(e_{ij})$ for network n. This normalization does not assure a limited range of values.

ScaleL

The previous proposals normalise the network only taking into consideration the score values. *ScaleL* is an extension of the last normalising method. This method takes into account the local context of the scores of gene i and j for computing the normalised score of interaction $t_n(e_{ij})$. In the *ScaleL* method (L stands for *Local*), two local scaled values are initially computed (ζ_i and ζ_j):

$$t_n(e_{ij}) = \sqrt{\zeta_i^2 + \zeta_j^2}, \text{ with } \zeta_i = \frac{s_n(e_{ij}) - \mu_{s_i}}{\sigma_{s_i}}, \text{ and } \zeta_j = \frac{s_n(e_{ij}) - \mu_{s_j}}{\sigma_{s_j}} \tag{6}$$

where μ_{s_i} and σ_{s_i} denote the mean and standard deviation of the empirical distribution of the scores of all edges connected to gene i. They are defined as:

$$\mu_{s_i} = \frac{1}{G}\sum_{l=1}^{G} s_n(e_{il}), \tag{7}$$

$$\sigma_{s_i} = \sqrt{\frac{1}{G-1}\sum_{l=1}^{G}(s_n(e_{il}) - \mu_{s_i})^2} \tag{8}$$

Note that this rule is related to the CLR network inference method [1] which can be interpreted as a normalization strategy as pointed out in [16]. This normalization step highlights a few links per node that stand out among all other scores of the gene. In this way, a ”core” network with the most relevant (and presumably true) links is obtained.

2.1.2 Aggregation

Two different aggregation techniques will be studied. Assume $a(e_{ij})$ denotes the aggregated value of the normalised scores.

Sum

The *Sum* is a simple summation process that is equivalent to the average of the N values of each link:

$$a(e_{ij}) = \sum_{n=1}^{N} t_n(e_{ij}). \tag{9}$$

Median

Finally, the *Median* method assigns the median of the N values:

$$a(e_{ij}) = \text{Median}\left\{t_1(e_{ij}), \ldots, t_N(e_{ij})\right\} \tag{10}$$

This method could be seen as a particularization of the (RankOrderFilter$_k$) with a fixed value of $k = {}^{N}/_{2}$.

2.2 Consensus network algorithms

The combination of four possible normalization strategies with three possible aggregation rules gives rise to 12 different consensus network algorithms. Regarding nomenclature, the algorithms will be referred to by the two names of the two steps, such that each word or abbreviation of the two steps begins with a capital letter.

Note that some of the combinations give a consensus method that has already been published. These methods are listed in Table 1. The other methods have not been reported in the literature and, to our knowledge, they are studied here for the first time.

Table 1: State-of-the-art consensus network algorithms.

Normalization	Aggregation	Name	Reference
Identity	*Sum*	*IdSum*	[15]
Rank	*Sum*	*RankSum*	[7]
Rank	*Median*	*RankMed*	[10]

3 Homogeneous versus Heterogeneous network scenario

The *Homogeneous* scenario reflects the case where the individual networks have been inferred with different algorithms but with the same type of data like gene expression as in [7]. In order to get an estimation of the degree of homogeneity that might be expected in this type of scenario, we have downloaded the networks from the supplemental information of [7]. To measure the homogeneity between networks, we have converted them to vectors and computed the correlation between the networks obtaining a mean correlation of 0.6.

On the other hand, it is possible to reconstruct GRN from different kind data. As an example a physical, regulatory network is one where edges represent a physical interaction between a TF and a TG as detected in chip assays or predicted using sequence-based DNA binding models (regulatory motifs). However such edges may not necessarily lead to functional changes in gene expression. In contrast, a functional regulatory network is one where edges between TF's and their targets are supported by functional changes in the gene expression [17], but as we already stated these relationships might be indirect.

From those observations comes the idea of combining both physical and functional evidence among others to further improve the inferred network. However, the nature of such networks are very different, and the recovered links are very different for each data. We thus refer to this scenario as *Heterogeneous* scenario. This situation reflects the case where the individual networks have been inferred from very different data types, and the consequently individual networks are very different, hardly having any edge in common. In [15] the authors tackle this problem using both supervised and unsupervised methods to predict regulatory edges by integrating datasets as input features. The unsupervised method consists on averaging of the links across different dataset networks. The final list is computed by sorting these score decreasingly.

To get an estimation of the expected correlation in this scenario, we have downloaded the networks from supplemental material of [15] and converted them to vectors. Afterwards, we have computed the correlation between the networks obtaining a mean correlation of 0.06.

Table 2: Synthetic networks used in this study and their characteristics.

Network	Name	Topology	Experiments	Genes	Edges
$Rogers_{1000}$	R1	Power-law tail topology	1000	1000	1350
$SynTReN_{300}$	S1	E. coli	800	300	468
$SynTReN_{1000}$	S2	E. coli	1000	1000	4695
GNW_{1565}	G1	E. coli	1565	1565	7264
GNW_{2000}	G2	Yeast	2000	2000	10392

4 Data

In this section, we present the data that belong to either homogeneous or heterogeneous group in order to test the methods described in the previous section.

4.1 Synthetic network generation

The different consensus methods are meant to be used for the integration of networks obtained through the use of real inference algorithms. However, first, we use synthetically generated networks. This allows providing a first approach and estimation of the performance of the different consensus network alternatives. Creating the individual networks in a synthetic manner allows us to control the degree of homogeneity between networks. Moreover, more importantly, we do not depend on any specific network inference algorithm. So, instead of relying on real network inference algorithms, we rely on a subsampling strategy applied on a network (True_{Net}). We use the networks proposed in [9], whose characteristics are detailed in Table 2.

Homogeneous scenario

The N individual networks are very similar and have many edges in common. To create the dataset corresponding to this scenario, a unique network is first created by a subsampling of True_{Net}. Then, this network is altered N different times by introducing hard errors (false positives and negatives) and by adding a Gaussian noise to the scores associated to all edges. The alteration parameters are chosen so that the homogeneity of the resulting networks is similar to the one obtained when various inference algorithms are applied to the same data.

Heteregeneous scenario

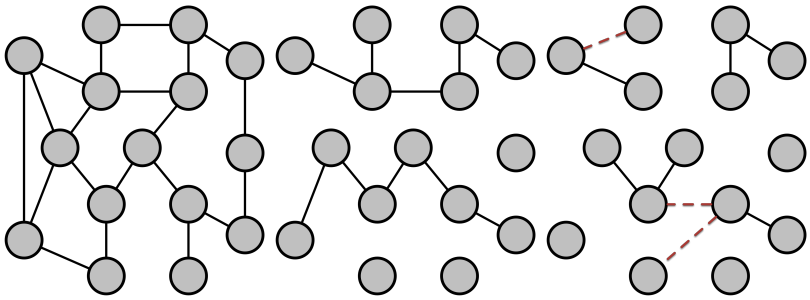
In this case, the N individual networks are very different and hardly have any edge in common. To reflect this situation, N different networks are generated through various independent subsampling of True_{Net}. As previously, these networks are then altered N different times with the introduction of hard errors (false positives and false negatives), and then with soft errors through the addition of noise.

4.1.1 Subsampling strategy

The networks of the two scenarios are generated with the Algorithm 1, which is illustrated with a toy example in Figure 1. A toy True_{Net} is shown in Figure 1a. It has 15 genes (illustrated with circles) and 20 edges (illustrated with lines). The subsampling step of Algorithm 1 selects randomly τ% edges of the True_{Net} to get ν_i. In Figure 1b a particular case of ν_i is shown, in this case $\tau = 50$ so ν_i has 10 edges. Then, m% of errors (both false positives and false negatives) are introduced. Following the example and with $m = 30$, this will introduce 3 false negatives and 3 false positives to obtain η_i. The resulting network is presented in Figure 1c, where the dashed lines represent the false positives. The following steps of Algorithm 1 are meant to generate realistic networks. First a noise is added to the network, after this process a constant value is added to shift the scores values in order to ensure non-negative scores in the networks. This step introduces more false positives with a low confidence (if δ is small) and also introduces variability of the scores in the true "recovered" edges.

Input: True_{Net}, *Heterogeneous*, N, τ, m
Output: N individual synthetic networks $\left(\{\eta_n\}_{n=1}^{N}\right)$
$n \leftarrow 1$;
while $n \leq N$ **do**
  if *Heterogeneous* **then**
    $\nu_n \leftarrow$ Subsample τ % edges from the True_{Net};
  else
    if *n==1* **then**
      $\nu_1 \leftarrow$ Subsample τ % edges from the True_{Net};
    else
      $\nu_n \leftarrow \nu_1$;
    end
  end
  $\eta_n \leftarrow$ Introduce m % errors in the network ν_n;
  Add noise to η_n;
  $n \leftarrow n+1$
end

Algorithm 1: Generate N individual synthetic networks.



(a) True_{Net} (b) Subsampled network ν_n (c) ν_n with errors

Fig. 1: Toy example illustration of the first steps of Algorithm 1.

4.2 DREAM5 Data

In a second step, we will use the predictions of 35 real GRN inference algorithms on the first simulated dataset of DREAM5 [7]. We use the inferred networks by the participants of the challenge. These networks are obtained from the same insilico data with different approaches. So we can consider it in the *Homogeneous* scenario. However, these networks present a particularity, the different algorithms presents a high variability in their performances (see Figure 3).

4.3 Real Data

Finally, the proposed consensus procedures are tested on public data sets of well-studied model organisms. With this study, we will improve our understanding of the algorithms seeing their performances when dealing with real data.

In this case, the metrics are computed according to a partial Gold standard (GS). This standard is generated by collecting all curated interactions for a particular organism. Therefore, this leads to a partial knowledge of the network.

All the collected interactions are treated as true positives, moreover, all predicted interactions between genes that are not documented in the curated database are treated as false positives. Such evaluation tends to overestimate the False Positive Ratio (FPR), as most genes probably interact with much more TF's than currently documented ones. Moreover, predictions for transcription factors and genes that are not part of the GS, i.e., for which no experimentally supported interactions exist, are ignored in the evaluation. This evaluation reward methods that tend to reproduce current knowledge and penalises those that could find new results [18].

In particular, we use two real data. The first one is chosen to represent the *Homogeneous* scenario and the second one the *Heteregeneous* scenario.

4.3.1 Escherichia coli

We have selected Escherichia coli (E. coli), a very well known bacteria, since predictions can be validated against the GS from RegulonDB database. The RegulonDB database [19, 20] contains the largest and best-known information on transcriptional regulation in E. coli. Thus, it has been used as a GS to evaluate the accuracy of the variously constructed networks.

We use different E. coli datasets:

1. *Ecoli1*: Many Microbe Microarrays Database (M3D) [21] which contains 907 microarrays measured under 466 experimental conditions using Affymetrix GeneChip E. coli Genome arrays.
2. *Ecoli2*: Expression data from laboratory evolution of Escherichia coli on lactate or glycerol [22] (GSE33147), which contains 96 microarrays of laboratory adaptive evolution experiments using Affymetrix E. coli Antisense Genome Arrays.
3. *Ecoli3*: Expression data from [23, 24] which contains 217 arrays that measures the transcriptional response of E. coli to different perturbations and stresses, such as drug treatments, UV treatments and heat shock.

Note that this data will be used to infer the networks with the GRN methods that are presented in [25]. Then, we use the different consensus strategies to integrate them.

4.3.2 Drosophila melanogaster

The fruit fly Drosophila melanogaster provides an ideal model organism for the inference and study of functional regulatory networks in multicellular organisms. There exists a rich literature about regulatory relationships, which have resulted in small, but high-quality networks of known regulatory interactions such as REDfly [26].

We have selected the data used in [15], since it provides a Heterogeneous scenario with networks of the same organism that comes from different kind of data. There is a total of six networks that comes from both functional and physical regulatory interactions.

5 Results

As mentioned before, the performances of the various algorithms are benchmarked with both real and synthetic networks.

In [9] have proposed to evaluate only the best $x\%$ of the total predictions. In [25], it is analysed how the $\text{AUPR}_{x\%}$ behaves as a function of x, and proposed to use the value of $\text{AUPR}_{5\%}$ as our metric.

As previously, we use AUPR_5 of the consensus network. But we are comparing very different data and situations, presenting different sizes and topological properties. Moreover, using AUPR_5 it is not possible to know if the consensus method is improving or not from the individual networks. Therefore, we propose to normalise the AUPR_5 with respect to the mean AUPR_5 of the individual networks ($\mu_{\text{AUPR}_{5;ind}}$):

$$\text{AUPR}_{5;norm} = \frac{\text{AUPR}_5}{\mu_{\text{AUPR}_{5;ind}}} \tag{11}$$

With this approach, we can compare the different networks. And therefore, it is possible to know if the consensus method is improving on the average network (if $\text{AUPR}_{5;norm}$ is greater than 1).

5.1 Results on Synthetic networks

In the case of synthetic networks, the individual networks are generated with Algorithm 1 with the parameters being specified in Table 3. Using these values, the mean correlation between the networks in the *Homogeneous* case is 0.66, and the average correlation of the *Heterogeneous* case is 0.003. On this experimental setup, we generate the individual networks for each one of the networks on Table 2, and this procedure is repeated ten times in order to have different runs of Algorithm 1 and therefore different pools of individual networks.

Table 3: Algorithm 1 parameters to generate the experimental setup for synthetic networks.

Parameter	Value
Number of individual networks (N)	10
Subsampling (τ) %	15
Introduced errors (m) %	20

Figure 2 presents the boxplots of $\text{AUPR}_{5;\text{norm}}$ of different consensus algorithms across all networks. Each box represents the statistics of a method, at the *Homogeneous* scenario and *Heterogeneous* scenario.

In the *Heterogeneous* case, we observe bigger differences between different consensus proposals. By observing the figure we can confirm that the *IdSum* method that was used in [15] in a *Heterogeneous* scenario is a good choice for this case. However, using the *Rank* as normalization step and *Sum* as aggregation step provides even better results.

On the other hand, the *homogeneous* scenario, it can be concluded that consensus network algorithms results in improving the inference compared to the average individual networks as the $\text{AUPR}_{5;norm}$ is around 4 for all consensus methods. This conclusion is in line with previous publications such as [10, 15]. However, there are

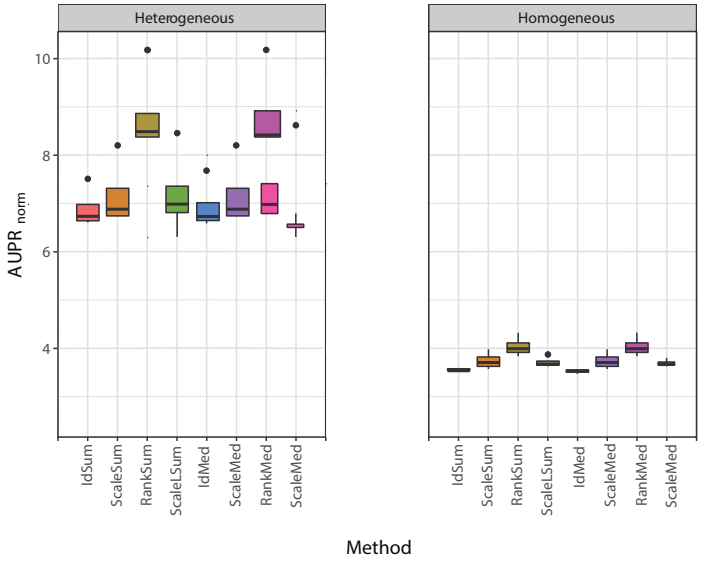


Fig. 2: Boxplots of $AUPR_{5;norm}$ performance of consensus methods on synthetic generated networks.

few differences between various algorithms. This case almost shows no significant differences between methods. Therefore, we think that the best option is to use the *RankSum* [7] or *IdSum* [15], as they are already part of the state-of-the-art and have a simpler normalization and aggregation methods.

This study shows how in both scenarios combining the results of multiple inference methods is a good strategy for improving the individual results. However, in the *Heterogeneous* case, it seems that there is still room for improvement. In average, the results improve by 8.5 outperforming the *Homogeneous*' results, which has an average $AUPR_{5;norm}$ of 3.7.

Since the individual results are synthetically generated, the obtained results should be interpreted as an estimation of the potential of consensus methods in the two different scenarios. Therefore, in the following subsections, we will study the different consensus methods on more real datasets.

5.2 Results on DREAM5

In this subsection, we have integrated the predictions of all 35 GRN predictions on DREAM5 to construct community networks with the different approaches.

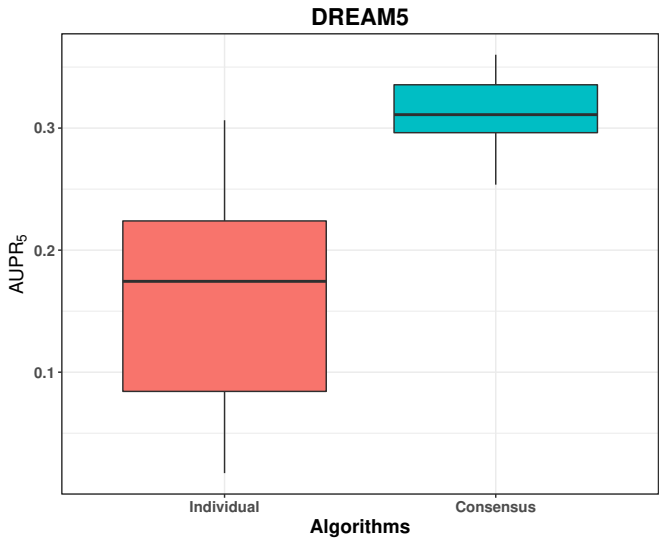


Fig. 3: Boxplots of AUPR_5 performance of individual networks and consensus methods on DREAM5.

The different consensus networks obtain in average better performances than the 35 applied inference methods, which shows that the community network is consistently as good or better than the top individual methods (Figure 3). Some of the top-performing methods are competitive with some of the consensus methods. However, as we have seen in the previous chapter the performance of individual methods does not generalise across different networks. Moreover, it is difficult to know in a real situation which one of the GRN algorithms has the best performance. Furthermore, in Figure 3, we can see how the mean value of AUPR_5 of individual networks is 0.17, while of the consensus network is 0.31. Obtaining a $\text{AUPR}_{5;norm} = 1.78$, which is a value more modest than the ones obtained on Synthetic generated networks.

Finally, the values of $\text{AUPR}_{5;norm}$ for each individual consensus method are shown in Figure 4. In this case, our proposal *ScaleLWsum* achieves the best consensus with a value of $\text{AUPR}_{5;norm} = 2.24$. We think that a big part of this good integration is obtained thanks to the normalization step since *ScaleLSum* also almost obtain the same performance.

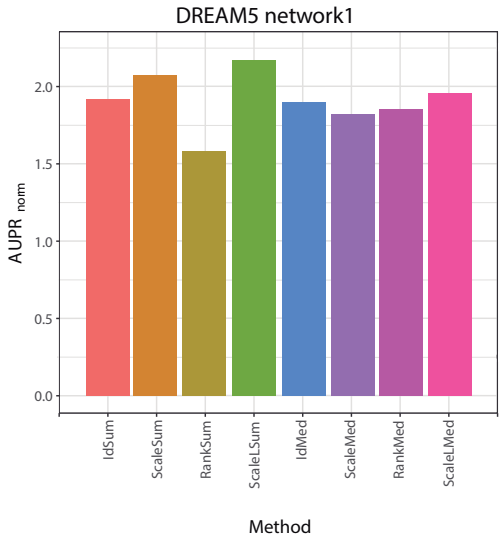


Fig. 4: Boxplots of $AUPR_5$ performance of individual networks and consensus methods on DREAM5.

5.3 Results on Real Data

To finish the analysis of consensus methods and their limits, we will evaluate them on the real data described in subsection 4.3.

With the three different E.coli microarray data, we have inferred GRN with the different methods described in [25]. Afterwards, we have integrated these networks with the different consensus proposals and evaluated them. Figure 5 shows the $AUPR_{5;norm}$ obtained by the different consensus proposals for each one of the datasets. The mean $AUPR_{5;norm}$ of the three values is marked with a black diamond.

Figure 5 confirms the conclusions reached in the previous subsection. *ScaleLWsum* is the best consesus method while *RankWsum* even obtains a worst result than the average individual method, $AUPR_{5;norm} < 1$, in Ecoli3 dataset.

The previous results reflect that in a *Homogeneous* scenario *ScaleLWsum* is the best alternative to the studied consensus methods. In order to have a larger picture, we evaluate the performance of consensus methods in a real *Heteregeneous* scenario with FlyNet data shown in Figure 6. Observing the figure we can confirm that the *IdSum* method that was used in the original work [15] is the best choice for this case.

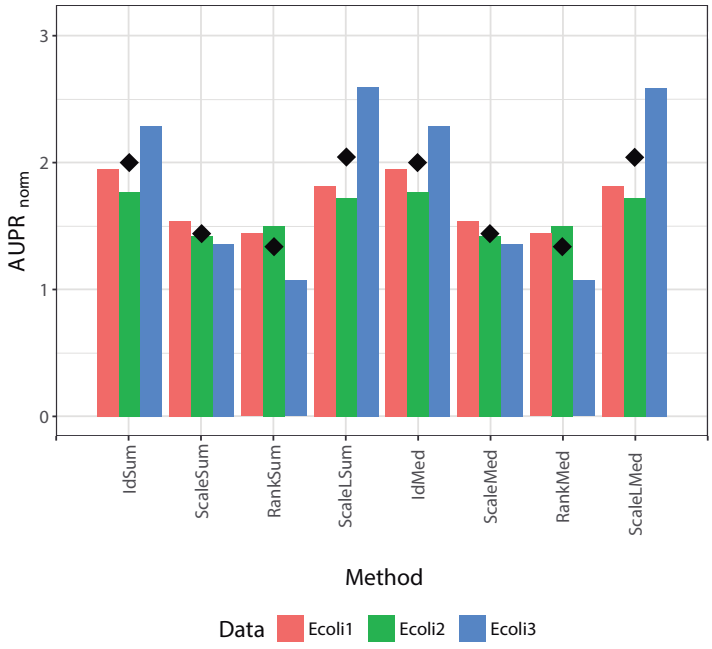


Fig. 5: $AUPR_{5;norm}$ performance of consensus methods on E.coli datasets.

6 Other unsupervised GRN ensemble options

In this chapter, we have presented different approaches to generate a consensus network to try to improve the inferred network. In this section, we present other approaches that does not fit in the presentd framework of normalization and aggregation.

6.1 Assembling pairwise matrices

In [27] a meta-analysis approach for inferring GRN from multiple studies is presented. The method is adapted to methods based on pairwise measures such as correlation or mutual information and consists of two steps: aggregating matrices of the pairwise measures from every dataset followed by extracting the network from the meta-matrix.

The proposed method aggregates mutual information matrices rather than data or networks. The idea behind assembling pairwise matrices is that, although expression data typically shows high variability due to differences in technology, samples, labels, etc., pairwise dependency measures between genes should be much less variant

(i.e. dependent variables, such as a regulating variable and its regulated counterpart, should remain dependent in every platform/experiment/dataset even if ranges of values differ significantly). Thus, to infer a network from various expression data, the approach consists in combining mutual information matrices (MIMs) estimated independently from every single dataset. And then, a GRN is created from inferring the aggregated Mutual Information Matrix using one of the Information-theoretic based GRN inference methods. We refer to the interested reader to [28] for a review of such GRN MIM based methods.

6.2 Topological ensemble

In [29] a post-processing algorithm called Netter is proposed. It changes the rank of the predicted edges of the inferred network. It tries to improve the structural properties (based on graphlets [30]) of the final network to resemble those typically found in a gene regulatory network.

The algorithm reorders only the top x links. Each ranking has an assigned cost and using simulated annealing [31] this cost is minimised several times, obtaining

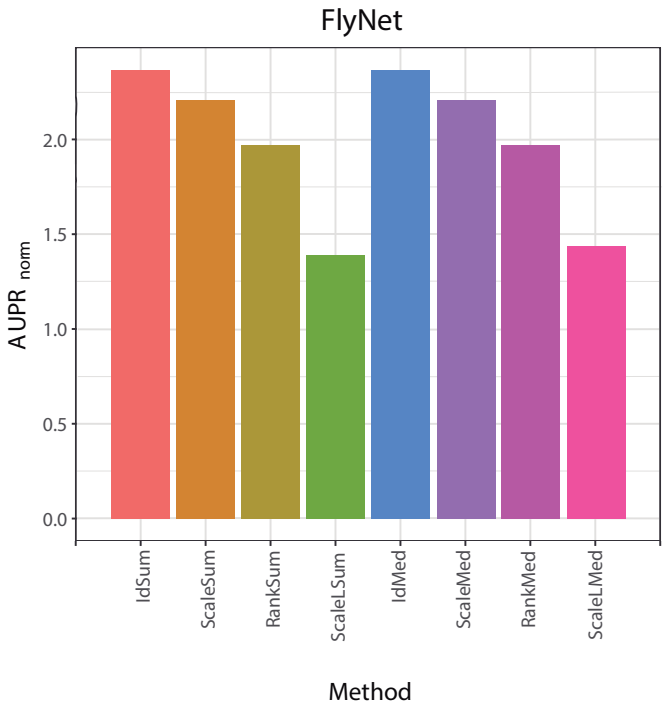


Fig. 6: $AUPR_{5;norm}$ performance of consensus methods on FlyNet.

different re-ranked lists. These lists are averaged to get the final output ranking. The cost function penalises the modification of the original ranking and rewards better structural properties. [29] proposes to use the frequency of graphlet G4 among the graphlets of four nodes (see Figure 7).

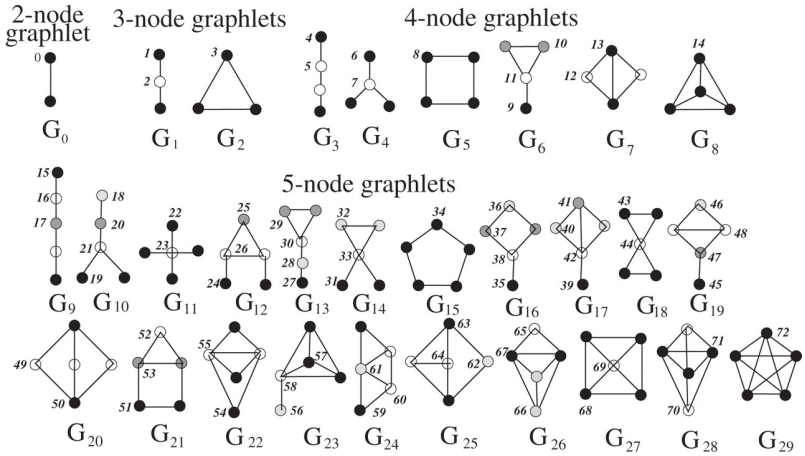


Fig. 7: The 73 automorphism orbits for the graphlets up to 5 nodes. In a particular graphlet G_i, $i \in \{0,1,\dots 29\}$, nodes that belongs to the same orbit have the same gray color. Figure taken from [30].

In [7], it is shown that some algorithms perform better than others. So, it may have sense to integrate only in the consensus the best-performing methods. The *WeightedSum* is based on this idea and implements it giving methods with a better performance a higher associated weight (w_n).

$$a(e_{ij}) = \sum_{n=1}^{N} w_n \cdot t_n(e_{ij}). \tag{12}$$

These weights can be learned in a supervised manner as in [15]. The authors propose a logistic regression-based binary classifier, where the class label represents the presence or absence of an edge.

[25] proposed to estimate these weights in an unsupervised manner, through an strategy where weights are proportional to the "topological quality" of the networks. The "topological quality" is measured by the relative frequency of some graphlets as compared to other graphlets. It concluded that the proposal *ScaleLWsum* seems to be a valuable choice and the best in some of the analysed real data.

7 Conclusions

In this chapter, we have proposed a framework for combining and integrating different inferred networks. It has been defined as a two-step process, consisting in a normalization strategy followed by an aggregation technique. We studied two different scenarios of practical interest: *Homogeneous* (a situation where various network inference algorithms are used on the same data) and *Heterogeneous* (a situation where various sources of data are used to generate the individual networks), with a controlled synthetic experimental setup. The results show how in a *Homogeneous* scenario combining individual networks generally outperforms the mean individual network, and that the different analysed algorithms do not present significant differences. However, in a *Heterogeneous* scenario the differences are very significant, and the potential win is much bigger. The choice of the proper normalization and aggregation steps allows very large improvements to be obtained.

Finally, we have studied how those results compare in different kinds of data when dealing with *Homogeneous* networks with very different performances and moreover in a real *Homogeneous* and *Heteregeneous* scenario. We have concluded that in this case, the improvement over the average individual network is smaller than the synthetic study. However, the increase in performance seems to be still attractive.

We can conclude that *IdSum* looks like the method of choice because it is simple and robust, always being one of the top performers, especially when all networks have a common scaling like in the FlyNet example (where all networks are 0-1. However, if the networks have a very different range of weights, then scaling becomes necessary to not overweight some networks. and in that case, *ScaleLSum* seems to be the best performing strategy.

References

1. Jeremiah J Faith, Boris Hayete, Joshua T Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J Collins, and Timothy S Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biol*, 5(1):e8, 2007.
2. Patrick E Meyer, Kevin Kontos, Frederic Lafitte, and Gianluca Bontempi. Information-theoretic inference of large transcriptional regulatory networks. *EURASIP journal on bioinformatics and systems biology*, 2007, 2007.
3. Patrick Meyer, Kevin Kontos, and Gianluca Bontempi. Biological network inference using redundancy analysis. *Bioinformatics Research and Development*, pages 16–27, 2007.
4. Patrick E Meyer, Daniel Marbach, Sushmita Roy, and Manolis Kellis. Information-theoretic inference of gene networks using backward elimination. In *BIOCOMP*, pages 700–705, 2010.
5. Gökmen Altay and Frank Emmert-Streib. Inferring the conservative causal core of gene regulatory networks. *BMC Systems Biology*, 4(1):132, 2010.
6. Gökmen Altay and Frank Emmert-Streib. Revealing differences in gene network inference algorithms on the network level by ensemble methods. *Bioinformatics*, 26(14):1738–1744, 2010.

7. Daniel Marbach, James C Costello, Robert Küffner, Nicole M Vega, Robert J Prill, Diogo M Camacho, Kyle R Allison, Manolis Kellis, James J Collins, Gustavo Stolovitzky, et al. Wisdom of crowds for robust gene network inference. *Nature methods*, 9(8):796–804, 2012.
8. Stefan R Maetschke, Piyush B Madhamshettiwar, Melissa J Davis, and Mark A Ragan. Supervised, semi-supervised and unsupervised inference of gene regulatory networks. *Briefings in bioinformatics*, page bbt034, 2013.
9. Pau Bellot, Catharina Olsen, Philippe Salembier, Albert Oliveras-Vergés, and Patrick E Meyer. Netbenchmark: a bioconductor package for reproducible benchmarks of gene regulatory network inference. *BMC bioinformatics*, 16(1):312, 2015.
10. Takeshi Hase, Samik Ghosh, Ryota Yamanaka, and Hiroaki Kitano. Harnessing diversity towards the reconstructing of large scale gene regulatory networks. *PLoS Comput Biol*, 9(11):e1003361, 11 2013.
11. Daniel Marbach, Robert J Prill, Thomas Schaffter, Claudio Mattiussi, Dario Floreano, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291, 2010.
12. Schaffter Thomas, Daniel Marbach, and Dario Floreano. GeneNetWeaver: In silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27(16):2263–2270, 2011.
13. Arun Krishnan, Alessandro Giuliani, and Masaru Tomita. Indeterminacy of reverse engineering of gene regulatory networks: The curse of gene elasticity. *PLoS ONE*, 2(6), 2007.
14. Frank Emmert-Streib, Galina V. Glazko, Gökmen Altay, and Ricardo de Matos Simoes. Statistical inference and reverse engineering of gene regulatory networks from observational expression data, 2012.
15. Daniel Marbach, Sushmita Roy, Ferhat Ay, Patrick E Meyer, Rogerio Candeias, Tamer Kahveci, Christopher A Bristow, and Manolis Kellis. Predictive regulatory models in drosophila melanogaster by integrative inference of transcriptional networks. *Genome research*, 22(7):1334–1349, 2012.
16. Pau Bellot and Patrick E Meyer. Efficient combination of pairwise feature networks. In *NCW2014 ECML*, 2014.
17. Andrew P Capaldi, Tommy Kaplan, Ying Liu, Naomi Habib, Aviv Regev, Nir Friedman, and Erin K O'Shea. Structure and function of a transcriptional network activated by the MAPK Hog1. *Nature genetics*, 40(11):1300–6, 2008.
18. Riet De Smet and Kathleen Marchal. Advantages and limitations of current network inference methods. *Nat Rev Microbiol*, 8(10):717–29, 2010.
19. S Gama-Castro, H Salgado, and M Peralta-Gil. RegulonDB version 7.0: transcriptional regulation of Escherichia coli K-12 integrated within genetic sensory response units (Gensor Units). *Nucleic acids*, 2011.
20. H Salgado, I Martínez-Flores, and A Lopez-Fuentes. Extracting regulatory networks of Escherichia coli from RegulonDB. *Networks: Methods and . . .* , 2012.
21. JJ Faith, ME Driscoll, and VA Fusaro. Many Microbe Microarrays Database: uniformly normalized Affymetrix compendia with structured experimental metadata. *Nucleic acids*, 2008.
22. SS Fong, AR Joyce, and BØ Palsson. Parallel adaptive evolution cultures of Escherichia coli lead to convergent growth phenotypes with different gene expression states. *Genome research*, 2005.
23. DP Sangurdekar and F Srienc. A classification based framework for quantitative description of large-scale microarray data. *Genome*, 2006.
24. G Xiao, X Wang, and AB Khodursky. Modeling three-dimensional chromosome structures using gene expression data. *Journal of the American*, 2011.
25. Pau Bellot. *Study of gene regulatory networks inference methods from gene expression data*. PhD thesis, Universitat Politècnica de Catalunya, 2017.
26. MS Halfon, SM Gallo, and CM Bergman. REDfly 2.0: an integrated database of cis-regulatory modules and transcription factor binding sites in Drosophila. *Nucleic acids research*, 2008.
27. Ngoc C. Pham, Benjamin Haibe-Kains, Pau Bellot, Gianluca Bontempi, and Patrick E. Meyer. Study of Meta-Analysis Strategies for Network Inference using Information-Theoretic Approaches. *Biological Knowledge Discovery and Data Mining*, 2016.

28. Patrick E Meyer, Catharina Olsen, and Gianluca Bontempi. Transcriptional network inference based on information theory. *Applied statistics for network biology: methods in systems biology. Weinheim, Wiley-Blackwell*, pages 67–89, 2011.
29. Joeri Ruyssinck, Piet Demeester, Tom Dhaene, and Yvan Saeys. Netter: re-ranking gene network inference predictions using structural network properties. *BMC Bioinformatics*, 17(1):76, 2016.
30. Nataša Pržulj. Biological network comparison using graphlet degree distribution. In *Bioinformatics*, volume 23, 2007.
31. Chii-Ruey Hwang. Simulated annealing: theory and applications. *Acta Applicandae Mathematicae*, 12:108–111, 1988.