Chapter 7
Network Topology Inference

In contrast to the previous chapter, with its focus on modeling and inference of observed network graphs, in this chapter we consider the problem of network topology inference, wherein the graph or some portion thereof is unobserved and we wish to infer it from measurements. There are a number of variations on this problem; we examine three particular forms in some depth.

7.1 Introduction

In Chapter 3, as part of our study of network mapping, we discussed the process of constructing a network graph representation for a system of interest based on a set of measurements from that system. Recall that this process, as described, was largely informal, with user-specified decisions and rules (presumably informed by expert knowledge) dictating the definition of what should constitute a vertex and an edge in the resulting graph. Such practice is typical – in fact, it is arguably the manner in which the vast majority of network graph representations are created.

As a result of its inherent informality, however, this manner of network graph construction distinctly lacks an element of validation, which can potentially be quite important depending on the context and the intended usage of the network graph. See the discussion in Section 3.7, for example. By 'validation' we have in mind issues like whether the network graph representation is 'accurate,' in terms of capturing some well-defined but unobservable relational structure in the underlying system; what accuracy ideally can be expected, given the information in the available set of measurements; whether there are other similar representations that are equally or almost as accurate; how robust the representation is to small changes in the measurements; and how useful the representation is as a basis for other, perhaps higher-level, purposes.

One natural way to endow the process of network graph construction with a greater sense of formality, and to facilitate a more precise formulation and study of such issues of validation, is to consider the task of constructing a network graph

E.D. Kolaczyk, *Statistical Analysis of Network Data*, Springer Series in Statistics,
DOI 10.1007/978-0-387-88146-1_7, © Springer Science+Business Media, LLC 2009

representation from data as one of statistical inference. To be more precise, suppose that, broadly speaking, we have a set of measurements from a system of interest, such as vertex attributes $\mathbf{x} = (x_1, \ldots, x_{N_v})^T$ or binary indicators $\mathbf{y} = [y_{ij}]$ of certain edges but not others, or both $\mathbf{x}$ and $\mathbf{y}$, and we have a collection $\mathscr{G}$ of potential network graphs G. We might then take as our goal to select an appropriate member of $\mathscr{G}$ that best captures the underlying state of the system, based upon the information in the data as well as any other prior information, using techniques of statistical modeling and inference. That is, we might pose the problem as one of *network topology inference.* In doing so, we position ourselves to potentially bring to bear a collection of accompanying concepts and tools – such as statistical consistency, efficiency, and robustness – on the problem of validation.

Unfortunately, there is at present no single coherent body of formal results on inference problems of this type, and certainly not at the level of generality we have posed. Rather, such problems have been pursued to varying extents in a handful of different areas. And the frameworks developed in these settings naturally take various forms, as dictated by context, with differences driven primarily by the nature of the topology to be inferred and the type of data available.

In this chapter, we will focus on discussing three particular classes of problems that have emerged. Each class of problems is somewhat 'canonical' in nature, being easily posed in more than just a single specific network context. More specifically, in Section 7.2 we consider the problem of inferring whether or not a pair of vertices does or does not have an edge between them (i.e., inference of 'edge' or 'non-edge' status), using measurements that include a subset of vertex pairs whose edge/non-edge status is already observed. This problem is commonly referred to as *link prediction.* Next, in Section 7.3, we discuss the inference of association networks. Here the relation defining edges is taken by definition to be a nontrivial level of association (e.g., correlation) between certain characteristics of the vertices, but is itself unobserved and must be inferred from measurements reflecting these characteristics. Finally, we examine problems of tomographic network inference in Section 7.4. These problems are distinguished by the fact that measurements are available only at vertices that are somehow at the 'perimeter' of the network, and it is necessary to infer the presence or absence of both edges and vertices in the 'interior.'

The ordering of these three classes of problems has a general progression. The first assumes knowledge of all of the vertices and the status of some of the edges/non-edges of the network graph, and seeks to infer the rest of the edges/non-edges. The second starts with no knowledge of edge status anywhere in the network graph, but assumes relevant measurements at all of the vertices and seeks to infer edge status throughout the network using these measurements. The third involves measurements at only a particular subset of vertices, which nevertheless indirectly provide some information useful for inferring the unknown topology of the rest of the network graph. A visual characterization of these three types of problems is shown in Figure 7.1.

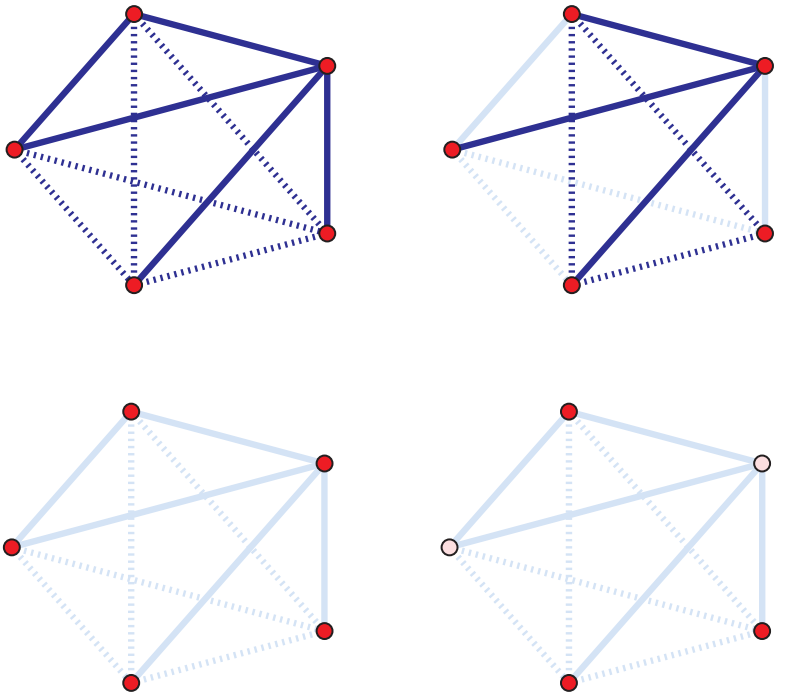

Fig. 7.1 Visual characterization of three types of network topology inference problems, for a toy network graph G. Edges shown in solid; non-edges, dotted. Observed vertices and edges shown in dark (i.e., red and blue, respectively); un-observed vertices and edges, in light (i.e., pink and light blue). Top left: True underlying graph G. Top right: Link prediction. Bottom left: Association graph inference. Bottom right: Tomographic network inference.

7.2 Link Prediction

Let $G = (V, E)$ be an undirected random network graph, and assume for simplicity that all vertices $v \in V$ are known. Recall that $V^{(2)}$ is the set of distinct unordered pairs of vertices. As such, it is (trivially) the union of (i) the set E of edges in G, and (ii) the set $V^{(2)} \setminus E$ of non-edges in G. In this section, we will suppose that among the vertex pairs in $V^{(2)}$, the presence or absence of an edge is observed only for some subset of pairs, say $V^{(2)}_{obs}$. For the remaining pairs, say $V^{(2)}_{miss} = V^{(2)} \setminus V^{(2)}_{obs}$, this information is missing.

Let $\mathbf{Y}$ be the random $N_v \times N_v$ binary adjacency matrix for the full network graph G, and denote by $\mathbf{Y}^{obs}$ and $\mathbf{Y}^{miss}$ the entries of $\mathbf{Y}$ corresponding to the vertex pairs in $V^{(2)}_{obs}$ and $V^{(2)}_{miss}$, respectively. The problem of *link prediction* is to predict the entries in $\mathbf{Y}^{miss}$, given the values $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$ and possibly various vertex attribute variables $\mathbf{X} = \mathbf{x} \in \mathbb{R}^{N_v}$. In other words, we wish to predict whether 'potential edges' between

pairs of vertices in a network graph are present or absent using information provided by a subset of observed edges/non-edges and, if available, vertex attributes.

There are a number of variants of the link prediction problem that have been formulated, arising in settings ranging from information networks (e.g., Liben-Nowell and Kleinberg [261], Popescul and Ungar [314], Taskar, Wong, Abbeel, and Koller [376]), to social networks (e.g., Hoff [200]), to biomolecular networks (e.g., Goldberg and Roth [178], Bader, Chaudhuri, Rothberg, and Chant [16]). Besides differing in context, these variants of the problem can also differ, importantly, in why and how the values in $\mathbf{Y}^{miss}$ are missing. Sometimes there is simply a temporal component to the problem and, for example, edges may be 'missing' only in the sense that they are absent up to a certain point in time and then become present, as in Liben-Nowell and Kleinberg [261] and their study of the growth of the World Wide Web network graph. In many cases, however, all edges and non-edges effectively coincide in time, but the status of potential edges is missing due to issues of sampling. In this latter case the underlying mechanism of missingness can be important.

A common assumption (e.g., Hoff [200], Taskar, Wong, Abbeel, and Koller [376]), and one we shall make here as well, is that the missing information on edge presence/absence is *missing at random*. This assumption means, essentially, that the probability that an edge variable Y_{ij} is observed depends only on the values of those other edge variables observed and not, for example, on its own value.[1] If edge variables Y_{ij} are missing with probability depending upon themselves, this is called *informative missingness*. One example of informative missingness is arguably in the context of certain biological networks, where edges are sometimes derived from databases summarizing the known results in the literature (e.g., gene 1 strongly influences gene 2, protein a has a strong affinity for protein b). Because of the well-documented bias towards 'positive' results in scientific literatures, there is a clear dependency of whether Y_{ij} is observed on whether or not it indicates a true edge. Informative missingness can render methods like those discussed below less effective, and calls for the extension of such methods to include, for example, additional modeling for the mechanism(s) of missingness. We will not pursue such extensions here.

In principle, given an appropriate model for $\mathbf{X}$ and $(\mathbf{Y}^{obs}, \mathbf{Y}^{miss})$, we might aim to jointly predict the elements of $\mathbf{Y}^{miss}$ based on a model for

$$\mathbb{P}\left(\mathbf{Y}^{miss} \,|\, \mathbf{Y}^{obs} = \mathbf{y}^{obs}, \mathbf{X} = \mathbf{x}\right) \quad . \tag{7.1}$$

But serious pursuit of this strategy entails, in part, successfully meeting the various challenges of modeling network graphs already described in Chapter 6, and in addition potentially modeling the missingness in an appropriate fashion. Perhaps not surprisingly, therefore, to date most efforts in this area have instead focused upon the comparatively more manageable task of predicting the variables Y_{ij}^{miss} individually.

[1] See Little and Rubin [263], for example, for a more formal definition and a general introduction to such missing data concepts.

Not only does this approach simplify matters considerably in many ways, but it also appears to in fact be a strong competitor to methods that predict the variables Y_{ij}^{miss} jointly. See Taskar, Wong, Abbeel, and Koller [376], for example. The methods we consider here in this section are designed for the prediction of individual Y_{ij}^{miss}.

7.2.1 Informal Scoring Methods

Scoring methods – methods based on the use of score functions – although somewhat less formal than the model-based methods we will discuss momentarily, are nevertheless popular and can be quite effective. Their essential element, a score function, can also be incorporated into the model-based methods as explanatory variables. Therefore, scoring methods serve as a useful starting point for our discussion.

With scoring methods, for each pair of vertices i and j whose edge status is unknown (i.e., for each potential edge $\{i,j\} \in V_{miss}^{(2)}$), a score $s(i,j)$ is computed. A set of predicted edges may then be returned either by applying a threshold s^* to these scores, for some fixed s^*, or by ordering them and keeping those pairs with the top n^* values, for some fixed n^*.

There are many types of scores that have been proposed in the literature. They generally are designed to assess certain structural characteristics of a network graph $G^{obs} = (V^{obs}, E^{obs})$ associated with $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$. A simple score, inspired by the 'small-world' principle, is negative the shortest-path distance between i and j,

$$s(i,j) = -\,\mathsf{dist}_{G^{obs}}(i,j) \quad . \tag{7.2}$$

The negative sign in (7.2) is present so as to have larger score values indicate vertex pairs more likely to share an edge. There are also a number of scores based on comparison of the observed neighborhoods $\mathcal{N}_i^{obs}$ and $\mathcal{N}_j^{obs}$ of i and j in G^{obs}, including the number of common neighbors

$$s(i,j) = |\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}| \quad , \tag{7.3}$$

a standardized version of this value, called the *Jaccard coefficient*,

$$s(i,j) = \frac{|\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}|}{|\mathcal{N}_i^{obs} \cup \mathcal{N}_j^{obs}|} \quad , \tag{7.4}$$

and a variation on this idea due to Liben-Nowell and Kleinberg [261], extending a more general idea of Adamic and Adar [4], of the form

$$s(i,j) = \sum_{k \in \mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}} \frac{1}{\log |\mathcal{N}_k^{obs}|} \quad . \tag{7.5}$$

This last score in (7.5) has the effect of weighting more heavily those common neighbors of i and j that are themselves not highly connected.

Most of the scores above assess only relatively local structure in G^{obs}. Continuing this line of thought, however, it is easy to conceive of higher-order summaries to use. For example, a variety of methods of scoring may be defined through different spectral characteristics of G^{obs}, analogous to many of those we have seen in Chapter 4. See Liben-Nowell and Kleinberg [261], for example.

Note the relevance that the mechanism of missingness can have here, with respect to the fact that the topology of G^{obs} is assumed to be somehow predictive. For example, if edges are 'missing' only because they can arise in the future, there is the implicit assumption in using the score (7.2) that vertices closer in the original period of observation will be more likely themselves to form a tie in the future, while the neighborhood-based scores (7.3) and (7.4) assume that two vertices originally having more neighbors in common makes the formation of an edge between them later more likely. The impact of different mechanisms of missingness on score function methods has yet to be explored in depth in the literature.

7.2.2 Probabilistic Classification Methods

As an alternative to informal scoring methods, it is appealing to approach link prediction as a classification problem. On the surface at least, to do so is relatively straightforward. The entries of $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$ are taken as training examples of cases of two types (i.e., 0's and 1's), and the goal is to use the information in $\mathbf{y}^{obs}$ and any vertex attributes to build a classifier, which in turn is then used to predict the entries in $\mathbf{Y}^{miss}$. Effectively, we aim to 'learn' an approximation to the rule dictating the presence and absence of edges in the network, and to use that approximation to predict those edges/non-edges we have not yet observed. Although in principle any type of standard classification method can be applied, classifiers based on logistic regression are common and we will build our discussion around this particular choice.

7.2.2.1 Logistic Regression Classifiers

Logistic regression classifiers in this context are based on models of the form

$$\log\left[\frac{\mathbb{P}_\beta\left(Y_{ij}=1|\mathbf{Z}_{ij}=\mathbf{z}\right)}{\mathbb{P}_\beta\left(Y_{ij}=0|\mathbf{Z}_{ij}=\mathbf{z}\right)}\right] = \beta^T\mathbf{z}\ , \tag{7.6}$$

where $\mathbf{Z}_{ij}$ is a vector of explanatory variables indexed in the unordered pairs $\{i,j\}$, and β is a vector of regression coefficients, assumed common to all pairs. Standard techniques, based on an efficient iteratively re-weighted least-squares algorithm, may be used to produce maximum likelihood estimates $\hat{\beta}$ of β. Details may be

found in Hastie, Tibshirani, and Friedman [194, Ch. 4.4], for example. Potential edges $\{i, j\}$ are then classified as being present or absent according to whether or not the estimated classification probabilities

$$\mathbb{P}_{\hat{\beta}}\left(Y_{ij}^{miss} = 1 \,|\, \mathbf{Z}_{ij} = \mathbf{z}\right) = \frac{\exp\left(\hat{\beta}^T \mathbf{z}\right)}{1 + \exp\left(\hat{\beta}^T \mathbf{z}\right)} \tag{7.7}$$

exceed a threshold (e.g., 0.5). If more than one model is entertained, standard techniques for model selection may be used to select an appropriate model. See Hastie, Tibshirani, and Friedman [194, Ch. 7], for example, for a description of such techniques in general regression and classification problems.

The $\mathbf{Z}_{ij}$ here are vectors of functions of the form

$$\mathbf{Z}_{ij} = \left(g_1(\mathbf{Y}_{(-ij)}^{obs}, \mathbf{X}), \ldots, g_K(\mathbf{Y}_{(-ij)}^{obs}, \mathbf{X})\right)^T \quad , \tag{7.8}$$

where $\mathbf{Y}_{(-ij)}^{obs}$ is defined to be all elements of $\mathbf{Y}^{obs}$ other than Y_{ij}. Note that the exclusion of Y_{ij} is only relevant to the training stage of our classification process (i.e., when the pair $\{i, j\}$ is among those observed). Such functions should be chosen to encode useful predictive information in $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$ and $\mathbf{X} = \mathbf{x}$. For example, they may extract the vertex-specific attributes x_i and x_j from $\mathbf{x}$, or simple transformations thereof. Or they may summarize structural information in a graph G^{obs} derived from $\mathbf{y}^{obs}$, in the manner of the scores described in Section 7.2.1. Or, finally, they may combine information in $\mathbf{y}_{(-ij)}^{obs}$ and $\mathbf{x}$, in analogy to (6.41). We provide some specific examples below in the illustration of Section 7.2.3.

Two issues are important to note in regard to the procedure just described. First, the standard logistic regression framework assumes that the observations in the training data are independent, conditional on the explanatory variables. However, as we have already discussed in Chapter 6, the Y_{ij} can typically be expected to be dependent upon each other in a nontrivial fashion, beyond what is specified in a simple logistic model. In fact, the use of logistic regression is essentially equivalent to the pseudo-likelihood approach associated with (6.40). However, there is no formal work to date exploring the implications on prediction accuracy of ignoring possible dependencies in this manner. Second, as with the informal scoring methods of Section 7.2.1, the nature of the underlying mechanism of missingness is relevant. The use of classification in the link prediction problem is reminiscent of cross-validation, and the assumption that data are missing at random is fundamental to the predictive accuracy of cross-validation procedures.[2] Therefore, presumably, if the missingness

[2] *Cross-validation* is a simple and general method for estimating prediction error in regression and classification problems. It seeks to mimic the ideal prediction setup, where we have available separate training and testing datasets of independent and identically distributed cases, in the context where we cannot or do not wish to separate the data in such fashion. It does so by dividing the data into K subsets – called 'folds' – and alternately using one fold to validate predictions made by a model fit to the other folds. The prediction errors on the K folds can then be used to produce an estimate of the overall prediction error of the underlying methodology, which in turn can be used

underlying $\mathbf{Y}^{miss}$ is not at random, the accuracy of the classification approach will suffer, although we might hope that the classification probabilities still provide a useful ordering of potential edges.

The issue of missingness is a topic on which, in the particular context of link prediction, little work has been done to date. However, the issue of dependence among the Y_{ij} can in principle be addressed by appropriate use of additional model structure, as we describe next.

7.2.2.2 Logistic Regression with Latent Variables

The use of latent variables is an intuitively appealing way to indirectly model unobserved factors driving the formation of network structure. We describe a very general class of latent variable models for this purpose, due to Hoff [200, 201], that extends the logistic regression framework above.

Let $\mathbf{M}$ be an unknown random, symmetric $N_v \times N_v$ matrix of the form

$$\mathbf{M} \;=\; \mathbf{U}^T \Lambda \mathbf{U} + \mathbf{E} \;\;, \tag{7.9}$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_{N_v}]$ is a random orthonormal matrix, Λ is an $N_v \times N_v$ random diagonal matrix, and $\mathbf{E} = [\varepsilon_{ij}]$ is a matrix of independent and identically distributed noise variables, with $\varepsilon_{ij} = \varepsilon_{ji}$, so as to enforce symmetry. Each element of $\mathbf{M}$ therefore has the form

$$M_{ij} \,|\, \mathbf{U}, \Lambda, \mathbf{E} = \mathbf{u}_i^T \Lambda \mathbf{u}_j + \varepsilon_{ij} \;\;. \tag{7.10}$$

Now, augment each observed explanatory variable $\mathbf{Z}_{ij}$ by the unobserved variable M_{ij}, so that (7.6) becomes

$$\log \left[\frac{\mathbb{P}_\beta \left(Y_{ij} = 1 | \mathbf{Z}_{ij} = \mathbf{z}, M_{ij} = m \right)}{\mathbb{P}_\beta \left(Y_{ij} = 0 | \mathbf{Z}_{ij} = \mathbf{z}, M_{ij} = m \right)} \right] = \beta^T \mathbf{z} + m \;\;. \tag{7.11}$$

Although the Y_{ij} are still assumed conditionally independent, given the $\mathbf{Z}_{ij}$ and M_{ij}, they are now conditionally *dependent* given only the $\mathbf{Z}_{ij}$, in contrast to the model in (7.6). This property follows as a result of the hierarchical nature of this model. The latent variable matrix $\mathbf{M}$ is intended to capture effects of network structural characteristics or processes not already described by the observed explanatory variables $\mathbf{Z}_{ij}$. Hoff [201] shows that the form of $\mathbf{M}$ in (7.9) includes other previously proposed latent variable models, such as those aimed at capturing effects of homophily (e.g., Hoff, Raftery, and Handcock [202]) or stochastic notions of equivalence (e.g., Nowicki and Snijders [304]).

In order to produce statistical predictions using this model it is necessary to additionally specify distributions for $\mathbf{U}$, Λ, and $\mathbf{E}$. If a Bayesian approach to inference is to be used, which is natural here, a prior distribution for β is also needed. An intuitive choice for modeling $\mathbf{U}$ is the uniform distribution on the space of all $N_v \times N_v$

for tasks like variable selection and setting tuning parameters. For additional details, see Hastie, Tibshirani, and Friedman [194, Ch. 7.10], for example.

orthonormal matrices. For the other variables, multivariate Gaussian distributions are convenient, due to the manner in which they facilitate MCMC sampling of the relevant posterior distribution through conjugacy properties. See Hoff [200, 201], for example. In predicting the values of elements in $\mathbf{Y}^{miss}$, it is natural to compare the values

$$\mathbb{E}\left(\frac{\exp\{\beta^T \mathbf{Z}_{ij} + M_{ij}\}}{1+\exp\{\beta^T \mathbf{Z}_{ij} + M_{ij}\}} \,\middle|\, \mathbf{Y}^{obs} = \mathbf{y}^{obs}, \mathbf{Z}_{ij} = \mathbf{z} \right) \tag{7.12}$$

to a given threshold, in analogy to our use of (7.7) previously. These values may be approximated numerically to any desired accuracy by the corresponding sample average of draws from the posterior indicated.

Note that the use of MCMC sampling noticeably increases the computational cost over the standard logistic regression method, which restricts the utility of this approach to networks of small to moderate size. This cost is primarily driven by the need to generate draws of the N_v^2 unobserved variables $\{U_{ij}\}$ in the process of sampling to approximate (7.12). But a significant reduction in cost can be obtained by specifying that $\mathbf{U}$ have only K non-zero column vectors, for some $K \ll N_v$, which constrains $\mathbf{M}$ to have an effective rank of K. Hoff [200, 201] has found values of K as small as 2 or 3 to work well in practice, which is of some interest in its own right.

7.2.3 Case Study: Predicting Lawyer Collaboration

Recall the network of collaborative working relationships among lawyers analyzed in Section 6.5.4. We will use the same data to illustrate the relative performance of some of the link prediction methods we have discussed through the device of a cross-validation study.

The set of $36(36-1)/2 = 630$ distinct vertex pairs in the network graph in Figure 6.7 were randomly divided into five separate subsets. For each subset in turn, the presence or absence of its 126 potential edges were taken to constitute the variables in $\mathbf{Y}^{miss}$, while the status of potential edges in the remaining four subsets was considered observed and defined $\mathbf{Y}^{obs}$. For each link prediction method examined here, predictions were generated for the elements of $\mathbf{Y}^{miss}$, using the information in $\mathbf{Y}^{obs}$ and, in some cases, the vertex attribute information on seniority, practice, gender, and office.

Four different methods were implemented for this illustration. Method 1 is the standard logistic regression in (7.6), where the explanatory variables for Y_{ij} were defined, in analogy to those in Section 6.5.4, to be

$$\begin{aligned}
Z_{ij}^{(1)} &= \text{seniority}_i + \text{seniority}_j \\
Z_{ij}^{(2)} &= \text{practice}_i + \text{practice}_j \\
Z_{ij}^{(3)} &= \text{Indicator of } \{\text{practice}_i = \text{practice}_j\} \\
Z_{ij}^{(4)} &= \text{Indicator of } \{\text{gender}_i = \text{gender}_j\}
\end{aligned}$$

$$Z_{ij}^{(5)} = \text{Indicator of } \{\text{office}_i = \text{office}_j\} \ .$$

Method 2 is also standard logistic regression, but with the above set of explanatory variables augmented by the additional variable

$$Z_{ij}^{(6)} = |\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}| \ . \tag{7.13}$$

This variable counts the number of neighbors common to vertices i and j in the graph G^{obs} induced by the vertex set V and the observations in $\mathbf{Y}^{obs}$. Method 3 is an informal scoring method, in which predictions were made based on the ranking of potential edges according to the number of common neighbors of their incident edges, as in (7.3), that is, based on their scores $s(i, j) = Z_{ij}^{(6)}$. Finally, method 4 implements the model of Hoff[3] in the form described above, using the same set of explanatory variables $\mathbf{Z}_{ij}$ as in method 1. The number of nontrivial columns in $\mathbf{U}$ was set to be $K = 3$.

Figure 7.2 shows receiver operating characteristic (ROC) curves[4] summarizing the predictive performance of the four methods as a function of an underlying threshold value. For methods 1, 2, and 4, this threshold is simply a value t ranging from 0 to 1, while for method 3, the threshold is the number n^* of vertex pairs predicted as having an edge. Each curve is an average over the five cross-validation folds. Since the network density was slightly under 0.2, in each fold the corresponding $\mathbf{Y}^{miss}$ had roughly 25 true edges and 100 true non-edges to be predicted.

All four methods can be seen to have performed noticeably better than random guessing (indicated by the gray line). The worst performance is observed for method 1, based on logistic regression using vertex attributes alone. The other three methods are quite similar in their performance, with the scoring method doing arguably just slightly worse than the other two. Comparison of the average area under the curve[5] for each of the four methods, as shown in Table 7.1, confirms this ordering. The improvement of methods 2, 3, and 4 over method 1 is indicative of the benefit derived from incorporating information on network structure. The fact that method 3 uses only network structure, whereas methods 2 and 4 use both network structure and vertex attributes, suggests that network structure alone is the more powerful explanatory factor here. This is confirmed by fitting models 2 and 4 without vertex attributes, which yields curves similar to that of method 3, though we do not show the results here.

---

[3] Computations were performed using the R package *eigenmodel*, available at *http://cran.r-project.org*, through the Comprehensive R Archive Network.

[4] An ROC curve relates the fraction of true positive predictions (y-axis) to the fraction of false positive predictions (x-axis) made by a binary classifier, as a parameter controlling the discrimination between positives and negatives is varied. Classifiers with curves pushing more into the upper left-hand corner are generally considered more desirable.

[5] The area under the curve (AUC) is a standard method for comparing ROC curves. It is maximized at 1 in the case of a perfect classifier, and is expected to be 0.5 in the case of random guessing.

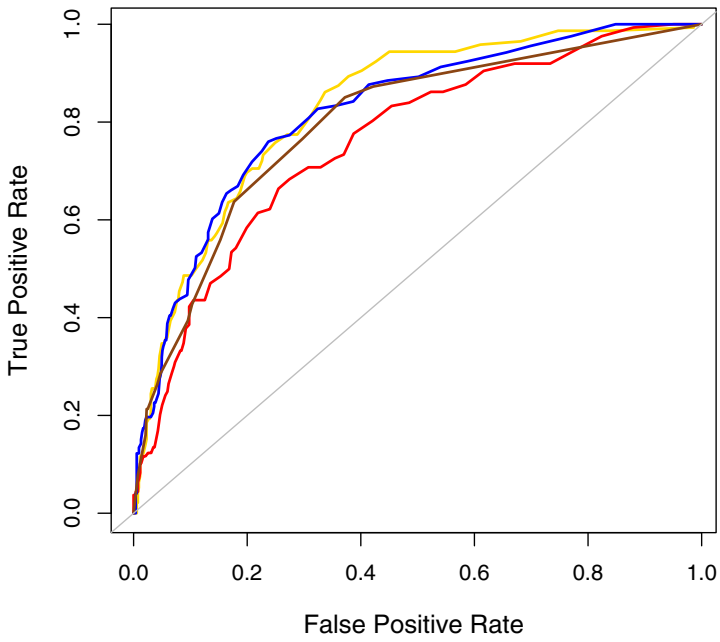

Fig. 7.2 ROC curves summarizing the capabilities to predict collaborative working relationships in the lawyer dataset of Section 6.5.4, for method 1 (red), based on logistic regression, with the explanatory variables $Z^{(1)}$ through $Z^{(5)}$, method 2 (blue), which is method 1 augmented with the variable $Z^{(6)}$, method 3 (brown), an informal scoring method based on scores $s(i, j) = Z_{ij}^{(6)}$, and method 4 (yellow), the method of Hoff [200, 201], using the same variables as in method 1.

	Method 1	Method 2	Method 3	Method 4
AUC	0.7638	0.8229	0.7963	0.8204

Table 7.1 Values of the area under the curve (AUC) for the ROC curves in Figure 7.2, averaged over five folds.

7.3 Inference of Association Networks

In Chapter 3 we saw that often the rule used for defining edges in the network graph representation of a given data set is that there be a sufficient level of 'association' between certain attributes of the two incident vertices. Such *association networks*

are found in many domains and include networks of citation patterns across scientific articles, networks of actors co-starring in movies, and networks of regulatory influence among genes. Frequently the rules defining edges in such networks are specified without statistics necessarily playing an explicit role. For example, recall the case study of Section 3.5.1, where the network of 'Science' shown in Figure 3.6 derives from use of the Jaccard measure of similarity in (3.1) and a combination of *ad hoc* thresholding and expert-guided clustering rules applied thereto.

While such rule-based approaches are appropriate in many contexts, in other contexts, where issues of sampling or measurement error are of potential concern, it may be necessary to incorporate statistical principles and methods into the process of constructing an association network graph. Generally speaking, we may suppose we have a collection of elements represented as vertices $v \in V$. Furthermore, suppose that each such vertex v has corresponding to it a vector $\mathbf{x} \in \mathbb{R}^m$ of observed vertex attributes, yielding a collection $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N_v}\}$ of attribute vectors. Let $\mathsf{sim}(i, j)$ be a user-specified quantification of the inherent similarity between the pair of vertices $i, j \in V$, and assume that it is accompanied by a corresponding notion of what value(s) of $\mathsf{sim}(i, j)$ constitute a 'non-trivial' level of association between i and j. We are interested here in the case where sim is not itself directly observable, but nevertheless the attributes $\{\mathbf{x}_i\}$ contain sufficiently useful information to make inference on sim conceivable.

Example 7.1 (Inference of Gene Regulation from Microarray Data). Regulatory interactions among genes are fundamental to the workings of biological organisms. Networks are a popular and useful tool for describing the patterns of known regulatory interactions, as we have already observed in the discussions surrounding Figures 1.3 and 3.2. Recall that, in this context, vertices represent genes and edges typically represent some notion of a regulatory relationship (e.g., activation or repression). Unfortunately, it is infeasible to construct such networks on a large-scale basis using traditional experimental techniques. But with the modern wealth of high-throughput genomic measurements, it is appealing to approach the construction of gene regulatory networks as a problem of network inference, given measurements sufficiently reflective of gene regulatory activity. See Gardner and Faith [166], for example, for an overview of various versions of this problem and some of the many approaches proposed for its solution.

Genes are sets of segments of DNA that encode information necessary to the proper functioning of a cell. The central dogma of biology says that such information is utilized in the 'expression' of genes, whereby biochemical products, in the form of RNA or proteins, are created. The 'regulation' of a gene refers to the control of its expression. One important stage at which such control is exerted is at the level of transcription, wherein DNA is copied to RNA. A gene that plays a role in controlling gene expression at this stage is called a 'transcription factor' (TF), and the genes that are controlled by it, gene 'targets.' The problem of inferring regulatory interactions among genes in this context refers to the identification of TF/target gene pairs.

The relative levels of RNA expression of genes in a cell, under a given set of conditions, can be measured efficiently on a genome-wide scale using microarray technologies. Microarray measurements are therefore a common type of data used in methods of inference of gene regulatory interactions. In particular, for each gene i, the vertex attribute vector $\mathbf{x}_i \in \mathbb{R}^m$ typically consists of RNA relative expression levels measured for that gene over a compendium of m experiments. We provide further details later in the case study of Section 7.3.4. Given such measurements, various techniques are then used in this context to summarize their similarity for each TF/target pair. We will describe a number of such techniques in additional examples below and in the case study. □

There are of course countless choices for sim in practice. In this section we will concentrate on the use of certain basic linear measures of correlation between univariate random variables $X \in \mathbb{R}$. This choice is both popular in practice and enables us to clearly highlight a number of the fundamental issues that arise in this problem – regardless of how sim is defined. Some comments on alternative choices of similarity may be found in Section 7.5.

7.3.1 Correlation Networks

Let $X \in \mathbb{R}$ be a (continuous) random variable of interest corresponding to the vertices in V. A standard measure of similarity between vertex pairs is $\mathsf{sim}(i,j) = \rho_{ij}$, where

$$\rho_{ij} = \mathsf{corr}_{X_i,X_j} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \tag{7.14}$$

is the *Pearson product-moment correlation* between X_i and X_j, expressed in terms of the entries of the covariance matrix $\Sigma = \{\sigma_{ij}\}$ of the random vector $(X_1,\ldots,X_{N_v})^T$ of vertex attributes. Given this choice, a natural criterion defining association between i and j is that ρ_{ij} be non-zero. The corresponding association graph G is then just the graph (V,E) with edge set

$$E = \left\{ \{i,j\} \in V^{(2)} : \rho_{ij} \neq 0 \right\} \ . \tag{7.15}$$

Therefore, given a set of observations of the X_i, the task of inferring this association network graph can be taken as equivalent to that of inferring the set of non-zero correlations.

This task can be approached through testing the hypotheses

$$H_0 : \rho_{ij} = 0 \quad \text{versus} \quad H_1 : \rho_{ij} \neq 0 \ . \tag{7.16}$$

However, there are at least three important issues to be faced in doing so. First, there is the choice of test statistic to be made. Second, given a test statistic, an appropriate null distribution must be determined for the evaluation of statistical significance.

And third, there is the fact that a large number of tests are to be conducted simultaneously (i.e., for all $N_v(N_v-1)/2$ potential edges), which implies that the problem of multiple testing must be addressed.

Suppose that for each vertex $i \in V$, we have n independent observations $x_{i1},\ldots,x_{in}$ of X_i. As test statistics, the *empirical correlations*

$$\hat{\rho}_{ij} = \frac{\hat{\sigma}_{ij}}{\sqrt{\hat{\sigma}_{ii}\hat{\sigma}_{jj}}} \quad , \tag{7.17}$$

are a common choice, where the $\hat{\sigma}_{ij}$ are the entries of the unbiased covariance matrix estimate

$$\hat{\Sigma} = \frac{1}{n-1}\left(\mathbf{X}-\bar{\mathbf{X}}\right)^T\left(\mathbf{X}-\bar{\mathbf{X}}\right) \quad , \tag{7.18}$$

with $\mathbf{X}$ here denoting the $n \times N_v$ matrix of observations x_{ik} and $\bar{\mathbf{X}}$, the matrix whose columns contain copies of the sample means $\bar{x}_i$. If the pair of variables (X_i, X_j) has a bivariate Gaussian distribution, the density of $\hat{\rho}_{ij}$ under $H_0 : \rho_{ij} = 0$ has a concise closed-form expression, but it is somewhat complicated and requires numerical integration or tables to produce p-values. See Hotelling [206]. Therefore, transformed versions of the sample correlation $\hat{\rho}_{ij}$ may be preferable.

For example, under the same conditions as above, the statistic

$$z_{ij} = \frac{\hat{\rho}_{ij}\sqrt{n-2}}{\sqrt{1-\hat{\rho}_{ij}^2}} \tag{7.19}$$

can be shown to have a t-distribution on $n-2$ degrees of freedom. Furthermore, under the null H_0, this distribution is relatively robust to departures of the X_i from Gaussianity. Also commonly used is Fisher's transformation,

$$z_{ij} = \tanh^{-1}(\hat{\rho}_{ij}) = \frac{1}{2}\log\left[\frac{(1+\hat{\rho}_{ij})}{(1-\hat{\rho}_{ij})}\right] \quad . \tag{7.20}$$

In this case, again for bivariate Gaussian pairs, the distribution of z_{ij} does not have a simple exact form. But under H_0 this distribution is well approximated by that of a Gaussian random variable with mean zero and variance $1/(n-3)$. The justification for this approximation is asymptotic in n, but has been observed to be quite accurate even for only moderately large n.

Example 7.2 (Correlation Among Gene Expression Levels). Recall our discussion of gene regulatory network inference in Example 7.1. In Section 7.3.4, we will examine the use of microarray measurements for inference of gene regulatory interactions on a large scale, using data for the bacteria *Escherichia coli (E. coli)*. Here we use a small subset of that data to illustrate our above discussion of correlations.

Figure 7.3 shows a plot of microarray measurements across $n = 445$ experiments for two TFs, *tyrR* and *lrp*, and a potential target gene, *aroG*. It is known that *aroG* is regulated by *tyrR* but not *lrp*. Also shown in each plot is a least-squares regression line and the empirical correlation $\hat{\rho}_{ij}$ in (7.17). The correlation of $\hat{\rho} = 0.85$ between

aroG and *lrp* is decidedly stronger than that between *aroG* and *tyrR*, at $\hat{\rho} = 0.43$. These values lead to Fisher z-scores (7.20) of $z = 1.2562$ and 0.4599, respectively, and comparison to a Gaussian distribution with mean zero and variance $1/(445-3) \approx 0.0023$ yields p-values of 4.27×10^{-152} and 7.69×10^{-22}, respectively. These results therefore suggest that, based on the use of correlation, the expression levels of the gene *aroG* are strongly associated with those of both *tyrR* and *lrp*. □

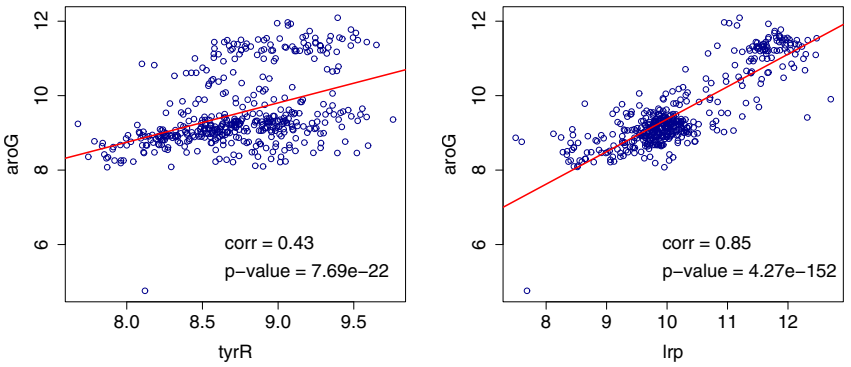

Fig. 7.3 Scatterplots of microarray measurements, in units of log-relative RNA expression levels, for the gene *aroG* and the transcription factors *tyrR* (left) and *lrp* (right), in the organism *E. coli*.

Alternatively, permutation methods may be used to construct an appropriate null distribution. In this case, it can still be advantageous to use a transformed version of the empirical correlations $\hat{\rho}_{ij}$, such as the z_{ij} in (7.19) or (7.20). For each pair of vertices $i, j \in V$, the labels "i" and "j" are randomly permuted on the $2n$ variables $(x_{i1}, \ldots, x_{in}, x_{j1}, \ldots, x_{jn})$ some large number B times. The value $z_{ij}^{(b)}$ is then computed on the 'new' data each time, for $b = 1, \ldots, B$. The significance of the original test statistic z_{ij} may then be assessed through comparison with the distribution of the B values $\{z_{ij}^{(b)}\}$. Note, however, that this approach can become computationally intensive for large N_v.

Given a choice of test statistic and the ability to compute p-values, false discovery rate (FDR) principles may be used to address the problem of multiple testing. For example, using the original method proposed by Benjamini and Hochberg [33], we would sort the p-values from our $N = N_v(N_v - 1)/2$ tests, yielding a sequence $p_{(1)} \leq p_{(2)} \leq \ldots \leq p_{(N)}$, and reject the null hypothesis for all potential edges for which

$$p_{(k)} \leq (k/N)\,\gamma \ . \tag{7.21}$$

Ideally, this approach will control the FDR for edges in the network graph at the level γ (i.e., it guarantees that $\mathrm{FDR} \leq \gamma$, where FDR is defined in (2.24)). Alternatively, we could employ the method of Storey [370] and declare edges to be present

using a particular q-value, say q^*. Our expectation is then that only q^*N of those edges will be included erroneously.

In implementing and interpreting such FDR procedures in this manner to construct an association network graph, two points should be kept in mind. First, it is assumed that the p-values serving as input to the procedure are calculated correctly; that is, that they are calculated under an accurate representation of the null distribution. So issues of robustness, like those discussed above, are relevant. Second, for the standard FDR procedures, like those two just mentioned, the stated levels of control are most valid when the tests are independent. Under dependency of tests, the actual control levels can vary from those stated, although how much they vary depends upon both the extent and type of dependency involved.

In our present setting, dependency clearly will be found among the N tests, given the all-pairs nature of the comparisons being made. If the dependency among the test statistics is essentially positive in nature, work of Benjamini and Yekutieli [34] suggests that the control guarantee offered by the standard FDR procedure for independent tests still continues to hold. In addition, these authors offer a simple extension of FDR for the case of more general types of dependency, which in our context entails replacing the value γ in (7.21) with $\gamma/\sum_{k=1}^{N}(1/k)$. Alternatively, we could use more sophisticated, adaptive FDR procedures, which try to estimate a certain parameter(s) reflective of the dependency and use the estimate to adjust the FDR. See, for example, Efron [130] and other references therein.

7.3.2 Partial Correlation Networks

The oft-cited dictum *'correlation does not imply causation'* should be kept in mind when constructing association networks based on Pearson's correlation and the like. Two vertices $i, j \in V$ may have highly correlated attributes X_i and X_j because the vertices somehow strongly 'influence' each other in a direct fashion. Alternatively, however, their correlation may be high primarily because, for example, they each are strongly influenced by a third vertex, say $k \in V$, and hence X_i and X_j are each highly correlated with X_k. The extent to which this issue is problematic or not will of course depend in no small part on the intended usage of G. But if it is felt desirable to construct a graph G where the inferred edges are more reflective of direct influence among vertices, rather than indirect influence, the notion of *partial correlation* becomes relevant.

In words, the partial correlation of attributes X_i and X_j of vertices $i, j \in V$, defined with respect to the attributes $X_{k_1}, \ldots, X_{k_m}$ of vertices $k_1, \ldots, k_m \in V \setminus \{i, j\}$, is the correlation between X_i and X_j left over after adjusting for those effects of $X_{k_1}, \ldots, X_{k_m}$ common to both. Formally, letting $S_m = \{k_1, \ldots, k_m\}$, we define the partial correlation of X_i and X_j, adjusting for $\mathbf{X}_{S_m} = (X_{k_1}, \ldots, X_{k_m})^T$, as

$$\rho_{ij|S_m} = \frac{\sigma_{ij|S_m}}{\sqrt{\sigma_{ii|S_m}\sigma_{jj|S_m}}} \quad . \tag{7.22}$$

Here $\sigma_{ii|S_m}, \sigma_{jj|S_m}$, and $\sigma_{ij|S_m} = \sigma_{ji|S_m}$ are the diagonal and off-diagonal elements, respectively, of the 2×2 partial covariance matrix

$$\Sigma_{11|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \quad , \tag{7.23}$$

where Σ_{11}, Σ_{22}, and $\Sigma_{12} = \Sigma_{21}^T$ are defined through the partitioned covariance matrix

$$\mathsf{Cov}\begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \end{pmatrix} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \quad , \tag{7.24}$$

for $\mathbf{W}_1 = (X_i, X_j)^T$ and $\mathbf{W}_2 = \mathbf{X}_{S_m}$. Note that in the case of $m = 0$, the partial correlation in (7.22) reduces to the Pearson correlation in (7.14). If $(X_i, X_j, X_{k_1}, \ldots, X_{k_m})^T$ has a multivariate Gaussian distribution, then $\rho_{ij|S_m} = 0$ if and only if X_i and X_j are independent conditional on $\mathbf{X}_{S_m}$. For more general distributions, however, zero partial correlation will not necessarily imply independence (the converse, of course, is still true).

Partial correlations can be used in various ways for defining an association network graph G, with respect to vertex attributes $X_1, \ldots, X_{N_v}$. For example, for a given choice of m, we may dictate that an edge be present only when there is correlation between X_i and X_j regardless of which m other vertices are conditioned upon. That is,

$$E = \left\{ \{i, j\} \in V^{(2)} \,:\, \rho_{ij|S_m} \neq 0, \text{ for all } S_m \in V_{\setminus\{i,j\}}^{(m)} \right\} \quad , \tag{7.25}$$

where $V_{\setminus\{i,j\}}^{(m)}$ is the collection of all unordered subsets of m (distinct) vertices from $V \setminus \{i, j\}$. Other choices are clearly possible as well.

Under the definition of edges in (7.25), the problem of determining the presence or absence of a potential edge $\{i, j\}$ in G can be represented as one of testing

$$H_0 : \rho_{ij|S_m} = 0 \quad \text{for some} \quad S_m \in V_{\setminus\{i,j\}}^{(m)}$$

versus

$$H_1 : \rho_{ij|S_m} \neq 0 \quad \text{for all} \quad S_m \in V_{\setminus\{i,j\}}^{(m)} \quad . \tag{7.26}$$

In order to infer an association network graph in this context, given measurements $x_{i1}, \ldots, x_{in}$ for each vertex $i \in V$, we must again select a test statistic, construct an appropriate null distribution, and adjust for multiple testing, as above.

With respect to the issue of a test statistic, it is useful to first note that the analogue of the empirical correlation coefficient $\hat{\rho}_{ij}$ in (7.17) is the empirical partial correlation coefficient, $\hat{\rho}_{ij|S_m}$. An algorithmic definition of this value is that it is the result of (i) performing separate multiple linear regressions of the observations of X_i and X_j, respectively, on the observed values of $\mathbf{X}_{S_m}$, and then (ii) computing the empirical correlation (7.17) between the two resulting sets of residuals. In practice, however, when computing many such coefficients, it is more computationally efficient to use recursive expressions, which relate each m-th order empirical partial correlation coefficient to three $(m-1)$-th order coefficients. See, for example, Anderson [11, Ch. 2.5.3]. Exploiting redundancies in these recursions, all m-th or-

der coefficients for a given pair $i, j \in V$ may be computed in $O(m^3)$ time from the $(m-1)$-th order coefficients.

In approaching the testing problem in (7.26), it can be convenient to consider it as a collection of smaller testing sub-problems of the form

$$H_0' : \rho_{ij|S_m} = 0 \quad \text{versus} \quad H_1' : \rho_{ij|S_m} \neq 0 \ , \tag{7.27}$$

for which $\hat{\rho}_{ij|S_m}$ is a natural test statistic. Under the assumption that the joint distribution of $(X_i, X_j, X_{k_1}, \ldots, X_{k_m})^T$ is a multivariate Gaussian distribution, the empirical partial correlation coefficient $\hat{\rho}_{ij|S_m}$ has a null distribution that is known but, again, whose functional form is somewhat complicated, requiring numerical integration or tables for practical usage (Hotelling [206]). Fisher's transformation

$$z_{ij|S_m} = \frac{1}{2} \log \left[\frac{(1+\hat{\rho}_{ij|S_m})}{(1-\hat{\rho}_{ij|S_m})} \right] \tag{7.28}$$

is therefore often used instead as a test statistic. It has an asymptotically Gaussian null distribution, with mean 0 and variance $1/(n-m-3)$.

Example 7.3 (Partial Correlation Among Gene Expression Levels). Recall the microarray data shown in Figure 7.3 of Example 7.2. It is known that a TF can actually be a target of another TF. And so direct correlation between measurements of a TF and a gene target may actually just be a reflection of the regulation of that TF by another TF. Partial correlation provides a tool for attempting to sort out such confounding among variables.

In Figure 7.4 are shown scatterplots of the data for the gene target *aroG* against that of the TFs *tyrR* and *lrp*, each adjusted for the effects of the other. More precisely, the measurements of *aroG* were regressed upon those of *lrp*, and similarly those of *tyrR* were regressed upon those of *lrp*, and the residuals from the two regressions were plotted. The analogous analysis was done with the roles of *tyrR* and *lrp* reversed. We can see that the strength of the association between *aroG* and *lrp* is little changed, as measured now by the empirical partial correlation coefficients, while that between *aroG* and *tyrR* has dropped by nearly a third. Note that the association between *aroG* and *tyrR* would now be judged not to be at all significant at any reasonable level, based on the p-value 0.92, which derives from comparison of the Fisher z-value in (7.28) to a Gaussian distribution with mean zero and variance $1/(445-1-3) \approx 0.0023$.

Also shown in Figure 7.4 are similar scatterplots based on adjusting for the effects of $m = 152$ TFs rather than just one. So, for example, in comparing *aroG* and *tyrR* we adjust for the effects of not only *lrp* but also 151 other TFs in *E. coli*. Similarly, in comparing *aroG* and *lrp* we adjust for *tyrR* and the same 151 other TFs. In both cases, we adjust by regressing the expression levels of *aroG* and the TF of interest on those of the other 152 TFs. From the plots we can see that there is moderately strong evidence of association left in the residuals, as indicated by the p-values corresponding to the empirical partial correlation coefficients. Again the z-value in (7.28) was used, but now with a variance of $1/(445-152-3) \approx 0.00345$

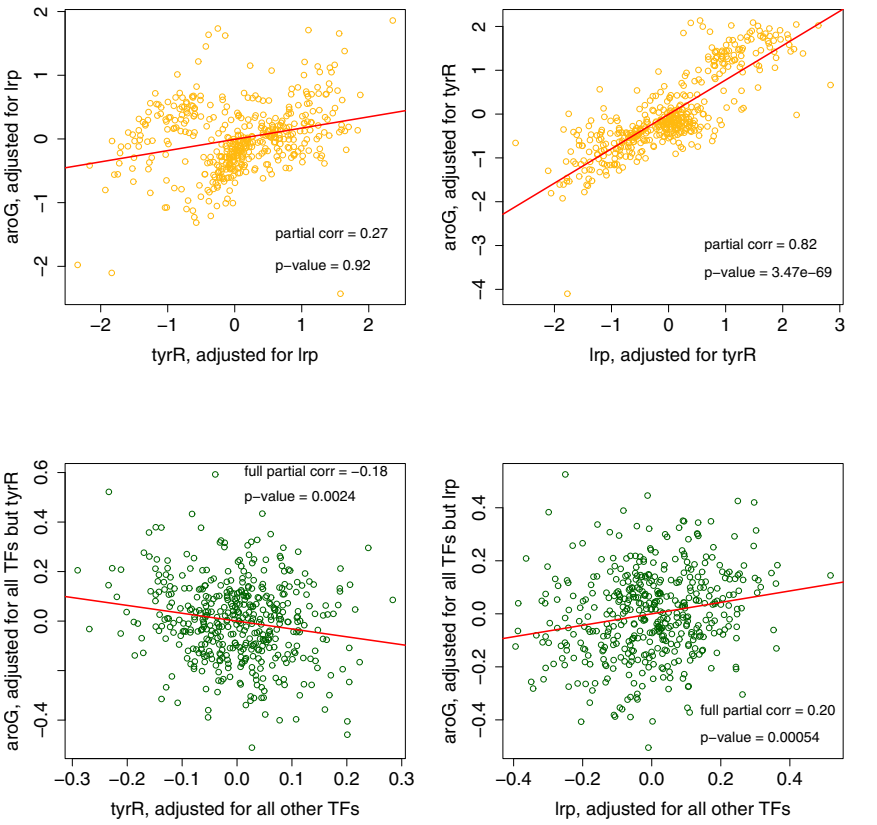

Fig. 7.4 Scatterplots of microarray measurements, in units of log-relative RNA expression levels, for the gene *aroG* and the transcription factors *tyrR* (left column) and *lrp* (right column). Top row (yellow) shows the data adjusted for *lrp* (left) and *tyrR* (right); bottom row (green) shows the data adjusted for all other transcription factors.

for the corresponding Gaussian under the null hypothesis. These results suggest that, beyond the effects of the other TFs, in each case the expression levels of *aroG* have a non-trivial association with those of the remaining TF (i.e., *tyrR* or *lrp*, respectively). Interestingly, the sign of the association between *aroG* and *tyrR* has in principle changed, from positive to negative, although recall that in the former case the p-value did not suggest a partial correlation significantly different from zero. This negative association suggests a repressive role of *tyrR* in regulating *aroG*. □

A test of (7.26) may be constructed from the tests of the sub-problems (7.27) through aggregation. For example, Wille et al. [406] suggest combining the information in p-values $p_{ij|S_m}$, over all S_m by defining

$$p_{ij,max} = \max \left\{ p_{ij|S_m} : S_m \in V^{(m)}_{\setminus\{i,j\}} \right\} \tag{7.29}$$

to be the p-value for the test in (7.26), where the $p_{ij|S_m}$ are the p-values for the tests of (7.27). Given the full collection of such p-values $\{p_{ij,max}\}$, over all potential edges $\{i, j\}$, an FDR procedure may be applied to this collection to choose an appropriate testing threshold, analogous to the manner described above. See Wille and Bühlmann [405] for justification.

Note that some caution is called for in choosing the order(s) m defining the underlying association network graph. For example, if $n \leq m$ the empirical partial correlation coefficients $\hat{\rho}_{ij|S_m}$ are not well defined. And even when $n > m$, these values are likely to be poor estimates of $\rho_{ij|S_m}$ if m is still large with respect to n. In addition, with increasing m the computational costs grow exponentially fast. Citing a worst case total cost of $O(n^{m+2})$ if coefficients are to be computed for all vertex pairs, de la Fuente, Bing, Hoeschele, and Mendes [110] advocate using only partial correlation coefficients up to order $m = 2$ in the context of inference of biochemical networks.

7.3.3 Gaussian Graphical Model Networks

A special – and, indeed, popular – case of the use of partial correlation coefficients is when $m = N_v - 2$ and the attributes are assumed to have a multivariate Gaussian joint distribution. Here the partial correlation between attributes of two vertices is defined conditional upon the attribute information at *all* other vertices. Denoting these coefficients as $\rho_{ij|V\setminus\{i,j\}}$, under the Gaussian assumption the vertices $i, j \in V$ have partial correlation $\rho_{ij|V\setminus\{i,j\}} = 0$ if and only if X_i and X_j are conditionally independent given all of the other attributes. The graph $G = (V, E)$ with edge set

$$E = \left\{ \{i, j\} \in V^{(2)} : \rho_{ij|V\setminus\{i,j\}} \neq 0 \right\} \tag{7.30}$$

is called a *conditional independence graph*.[6] The overall model, combining the multivariate Gaussian distribution with the graph G, is called a *Gaussian graphical model.* This model is one example of the more general class of graphical models, a point we will revisit briefly in Chapter 10.

A useful result in the context of Gaussian graphical models is that the partial correlation coefficients may be expressed in the form

$$\rho_{ij|V\setminus\{i,j\}} = \frac{-\omega_{ij}}{\sqrt{\omega_{ii}\omega_{jj}}} \ , \tag{7.31}$$

[6] Note that it is actually the non-edges in G that correspond to conditional independence, and the edges, to conditional dependence.

where ω_{ij} is the (i,j)-th entry of $\Omega = \Sigma^{-1}$, the inverse of the covariance matrix Σ of the vector $(X_1,\ldots,X_{N_v})^T$ of vertex attributes. See Lauritzen [248, Ch. 5.1] or Whittaker [403, Ch. 5.8], for example. The matrix Ω is known as the *concentration* or *precision* matrix, and its non-zero off-diagonal entries, occuring as they do in the numerator of (7.31), are linked in one-to-one correspondence with the edges in G. As a result, G also is sometimes referred to as a *concentration graph*.

The problem of inferring G from data is known in this context as the 'covariance selection problem' (Dempster [115]). A standard approach to this problem employs a recursive, likelihood-based procedure that effectively tests the hypotheses

$$H_0 : \rho_{ij|V\setminus\{i,j\}} = 0 \quad \text{versus} \quad H_1 : \rho_{ij|V\setminus\{i,j\}} \neq 0 \ , \tag{7.32}$$

using the empirical partial correlations $\hat{\rho}_{ij|V\setminus\{i,j\}}$, defined as in (7.31), but with the entries of $\hat{\Omega} = \hat{\Sigma}^{-1}$ in place of those of Ω, where $\hat{\Sigma}$ is as in (7.18). Starting with the complete graph on N_v vertices as an initial estimate $\hat{G}^{(0)}$, and a pre-specified significance level, say α, edges corresponding to those vertex pairs $\{i,j\}$ for which the null hypothesis in (7.32) is not rejected at level α are removed. The resulting graph, say $\hat{G}^{(1)}$, now summarizes a set of conditional (in)dependencies. Taking these now as known constraints, the procedure is repeated, yielding an estimate $\hat{G}^{(2)}$, and so forth, until no further edges are removed. See Whittaker [403, Ch. 6] or Lauritzen [248, Ch. 5], for details.

For large-scale network graphs, this approach presents a number of problems. First, it is computationally intensive and really only feasible for graphs of quite modest size. Second, there is no attention paid to multiple testing. And third, it is frequently the case in large-scale settings that $n \ll N_v$, and so the estimated covariance matrix $\hat{\Sigma}$ will not be of full rank, and hence will not have a proper inverse. A number of methods have been proposed for inference in this setting, which we review here, organized according to those based on testing and those based on regression.

7.3.3.1 Methods Based on Testing

Drton and Perlman [127] suggest a method of graph inference that addresses the problem of multiple testing in the case where $n > N_v$. Working with the Fisher transformation $z_{ij|V\setminus\{i,j\}}$, defined in analogy to (7.28), they propose a collection of tests jointly at the α significance level, that assign an edge to each pair $\{i,j\}$ if and only if

$$|z_{ij|V\setminus\{i,j\}}| > n_{N_v}^{-1/2} c_{N_v}(\alpha) \ , \tag{7.33}$$

where $n_{N_v} = n - N_v$ and

$$c_{N_v}(\alpha) = \Phi^{-1}\left[0.5(1-\alpha)^{[2/N_v(N_v-1)]} + 0.5\right] \ , \tag{7.34}$$

for $\Phi(\cdot)$ the cumulative distribution function of the Gaussian distribution with mean 0 and variance 1. Under the assumption that G is the true conditional independence graph underlying the data, it can be shown that G will be correctly inferred by this procedure with probability at least $1-\alpha$, for large n.

Note that this procedure is not applicable to the case $n \ll N_v$. When the sample size is less than the number of vertices, the estimate $\hat{\Sigma}$ in (7.18) will still be unbiased for Σ but will suffer from undue variance, which in turn will be detrimental to any inference on the partial correlation coefficients (7.31) that use (pseudo)inversion of $\hat{\Sigma}$ to estimate Ω. Schäfer and Strimmer [340, 341] have suggested a number of methods for network inference in this context, involving various techniques for producing reduced-variance estimates of Σ.

For example, in Schäfer and Strimmer [340] the use of bootstrap aggregation – or 'bagging' – is proposed for variance reduction. Specifically, simple random sampling of the N_v-dimensional sample vectors $\mathbf{x}$ is used to create a collection of some large number B 'new' datasets, and for each new dataset $b = 1,\ldots,B$, an estimate $\hat{\Sigma}^{(b)}$ is computed. A smoothed covariance estimate $\hat{\Sigma}^{bag}$ can then be obtained through the average of the $\hat{\Sigma}^{(b)}$, and an improved estimate of Ω results from computing the Moore-Penrose inverse of $\hat{\Sigma}^{bag}$. Substitution in (7.31) then yields partial correlation estimates, say $\hat{\rho}^{bag}_{ij|V\setminus\{i,j\}}$. See Breiman [62] or Hastie, Tibshirani, and Friedman [194, Ch. 8.7] for background and motivation on the principle of bagging.

For testing the hypotheses in (7.32), in order to construct an inferred conditional independence graph, an appropriate null distribution for the estimates $\hat{\rho}^{bag}_{ij|V\setminus\{i,j\}}$ must be specified. Since an analytic approach to this task is daunting, Schäfer and Strimmer instead choose to model this distribution, proposing a mixture model of the form

$$f(\rho) = \pi f_0(\rho;\nu) + (1-\pi) f_1(\rho) \ , \tag{7.35}$$

where f_0 has the same functional form as the distribution of the estimates $\hat{\rho}_{ij}$ and $\hat{\rho}_{ij|k_1,\ldots,k_m}$, given in Hotelling [206], and f_1 is the uniform distribution on $[-1,1]$. The parameter ν in f_0 is a degree-of-freedom parameter that is estimated empirically. The mixture weight π also can be estimated empirically. This mixture model itself is similar to models underlying the pFDR methodology of Storey [370], and also Efron's [130] extension of FDR, and therefore multiple testing procedures can be integrated easily into the analysis. See Schäfer and Strimmer [340, 341] for details.

7.3.3.2 Methods Based on Penalized Linear Regression

It has also been proposed to use penalized regression methods to infer G, exploiting a variation on the well-known connection between linear correlation and linear regression. Suppose that the random vector $(X_1,\ldots,X_{N_v})^T$ of vertex attributes has a multivariate Gaussian distribution, with covariance Σ, as assumed above, and also zero mean. Then a standard result yields that, for the attribute X_i of a fixed vertex i, given the values of the attributes at the remaining vertices, say $\mathbf{X}^{(-i)} = (X_1,\ldots,X_{i-1},X_{i+1},\ldots,X_{N_v})^T$, its conditional expectation has the form

$$\mathbb{E}[X_i \,|\, \mathbf{X}^{(-i)} = \mathbf{x}^{(-i)}] = \left(\beta^{(-i)}\right)^T \mathbf{x}^{(-i)} \quad , \tag{7.36}$$

where $\beta^{(-i)} \in \mathbb{R}^{N_v - 1}$. See, for example, Johnson and Wichern [214, Ch. 4]. Furthermore, importantly, the entries of $\beta^{(-i)}$ can be expressed in terms of the entries of the precision matrix Ω, as $\beta_j^{(-i)} = -\omega_{ij}/\omega_{ii}$. See Lauritzen [248, App. C]. Hence, a potential edge $\{i, j\}$ is in the edge set E defined by (7.30) if and only if $\beta_j^{(-i)}$ (and, therefore, also $\beta_i^{(-j)}$) is not equal to zero.

These observations suggest that inference of G be pursued via inference of the non-zero elements of $\beta^{(-i)}$ in (7.36), using regression-based methods of estimation and variable selection. In fact, it can be shown that the vector $\beta^{(-i)}$ is the solution to the optimization problem

$$\arg \min_{\tilde{\beta}:\tilde{\beta}_i = 0} \mathbb{E}\left[\left(X_i - \left(\tilde{\beta}^{(-i)}\right)^T \mathbf{X}^{(-i)}\right)^2\right] \quad . \tag{7.37}$$

It is natural, therefore, to replace this problem by a corresponding least-squares optimization. However, because there will be $N_v - 1$ variables in the regression for each X_i, and in addition it may be the case that $n \ll N_v$, a penalized regression strategy is prudent.

Meinshausen and Bühlmann [275], for example, have suggested using estimates of the form

$$\hat{\beta}^{(-i)} = \arg \min_{\beta:\beta_i = 0} \sum_{k=1}^{n} \left(x_{ik} - \left(\beta^{(-i)}\right)^T \mathbf{x}_k^{(-i)}\right)^2 + \lambda \sum_{j \neq i} \left|\beta_j^{(-i)}\right| \quad . \tag{7.38}$$

This strategy is based on the use of the Lasso method of Tibshirani [380], which not only estimates the coefficients in $\beta^{(-i)}$ but also forces values $\hat{\beta}_j^{(-i)} = 0$ where the association between X_i and X_j is felt to be too weak, with respect to the choice of penalty parameter λ. In other words, the Lasso methodology performs simultaneous estimation and variable selection, a characteristic due to the particular form of the penalty. The estimates in (7.38) can be computed, using the so-called LARS procedure of Efron, Hastie, Johnstone, and Tibshirani [132], in $O(nN_v \min\{n, N_v\})$ steps. Therefore, to compute estimates for all vertices $i \in V$, and presuming that $n \ll N_v$, the computational cost will be slightly worse than $O(N_v^3)$.

Given the estimate $\hat{\beta}^{(-i)}$ for all $i \in V$, an inferred network graph can be constructed by assigning edges according to whether the corresponding estimated coefficients are zero or non-zero. Note that it is not true that $\hat{\beta}_j^{(-i)} \neq 0$ implies $\hat{\beta}_i^{(-j)} \neq 0$, and vice versa. One can therefore specify that a potential edge $\{i, j\}$ be assigned if either $\hat{\beta}_j^{(-i)}$ or $\hat{\beta}_i^{(-j)}$ are non-zero, or require that both be non-zero. Meinshausen and Bühlmann show that, under conditions on Σ, λ, N_v, and n, the true graph G will be inferred with high probability using either of these conventions, even in cases where $n \ll N_v$. Note that the choice of λ is important. Meinshausen and Bühlmann show that selecting λ by cross-validation will yield provably bad results, because

the goal is one of variable selection and not prediction. These authors suggest an empirically driven choice of λ, for which a small simulation study suggests some promise.

The use of penalized regression methods for inference in Gaussian graphical models has generated a substantial amount of interest in the research community. Other contributions in this area include Dobra et al. [119], who describe a Bayesian approach to the problem, Wainwright, Ravikumar, and Lafferty [390], who propose a Lasso-like penalized logistic regression approach, and Yuan and Lin [410], who present a method of penalized maximum likelihood. Friedman, Hastie, and Tibshirani [161] have developed a computationally fast procedure – the so-called graphical lasso – for solving this penalized maximum likelihood problem.

7.3.4 Case Study: Inferring Genetic Regulatory Interactions

Recall our discussion of gene regulatory interaction and microarray measurements in Examples 7.1, 7.2, and 7.3. Here we use a complete set of microarray data and correlation-based methods like those described above to do large-scale inference of regulatory interactions between TF/target gene pairs in the bacteria *Escherichia coli (E. coli)*. The dataset, as well as the basic structure of our analysis, are taken from Faith et al. [140], although the choice of methodologies used here differs.

The data consist of log-expression levels of RNA, relative to a background, measured for genes in *E. coli* when the cells were subjected to various conditions, such as changes in pH, growing media, heat, oxygen concentrations, and genetic composition. In full, a total of 445 vectors of measurements – called 'expression profiles' – were collected across $4,345$ genes. Among these genes, 153 are known TFs. Also available is information on currently known TF/target pairs in *E. coli*, for these 153 TFs and $1,058$ other of the $4,345$ genes, extracted from the database RegulonDB [338]. While the regulatory information in this database is not complete, it is largely so, and the information therein is held to be quite accurate, although not perfect. Thus we have available – somewhat uniquely for this problem – a high-quality truth-set by which to assess the usefulness of the regulatory relationships predicted by the methods we consider.

Figure 7.5 shows an image representation of the (log-relative) RNA expression levels across experiments for the 153 genes listed as known TFs in RegulonDB. Due to the nature of the regulation process, the expression levels of TFs and their targets often can be expected to be highly correlated with each other, as we saw earlier. And given that many TFs have multiple targets, including other TFs, the expression levels among many of the TFs can be expected to be highly correlated as well, as we also saw earlier. Such correlations are evident in the image plot, where the order of the genes (rows) and experiments (columns) has been chosen using a cluster analysis, to enhance the visual assessment of correlation.

In predicting regulatory interactions, we focused solely on interactions between TF/target gene pairs. We implemented three different correlation-based methods for

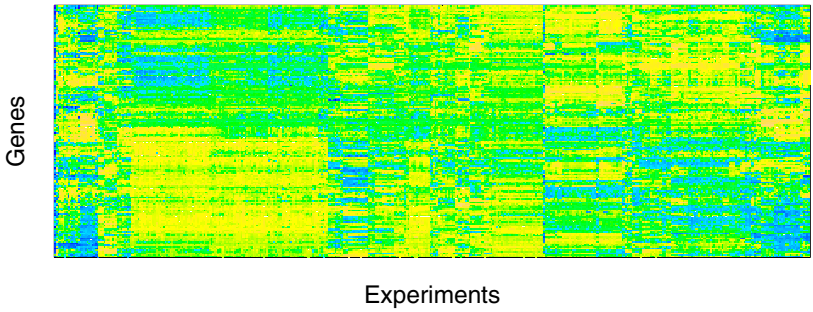

Fig. 7.5 Image representation of 445 microarray expression profiles collected for *E. coli*, under various conditions, for the 153 genes that are listed as known transcription factors in the RegulonDB database. Larger negative values are indicated with darker shades of blue, and larger positive values, in yellow to orange. Shades of green indicate values comparatively close to zero.

quantifying and detecting such interactions, wherein a regulatory interaction was declared for a TF/target pair if a p-value for the corresponding correlation-based statistic fell beneath a given threshold. The first method is based on the standard Pearson correlation, in (7.14), between the expression levels of a TF and a potential target gene. The second method is based on the partial correlation between the same pairs, controlling for possible shared effects of a single additional TF, across all 152 other TFs. More precisely, this method computes partial correlations like that in (7.22), with $m = 1$, with the expression levels of each TF individually playing the role of the conditioning random variable. It then declares a TF/target interaction based on the p-value in (7.29) for the testing problem in (7.26). Finally, the third method also uses partial correlation between TFs and potential targets, but simultaneously adjusts for the expression levels of all remaining $m = 152$ TFs (or $m = 151$, if the potential target also is a TF). We refer to this latter partial correlation as the 'full partial correlation' in what follows.

For each choice of correlation, the appropriate Fisher transformation was applied and p-values were computed using the corresponding asymptotic Gaussian distribution, based on a sample size $n = 445$. A threshold was then varied from 0 to 1. For p-values falling beneath the threshold, the corresponding TF/target pair were said to interact. An association network graph was created by assigning a vertex to each gene and an edge to each TF/target gene pair declared to interact.

We evaluated the performance of each of these three methods of network inference in two ways. First, each inferred network graph was compared to the graph induced by the information in the RegulonDB database, for just the 153 TFs and $153 + 1,058 = 1,211$ genes in that database. Figure 7.6 shows the resulting set of ROC curves. Overall, the partial and full partial correlation methods perform about

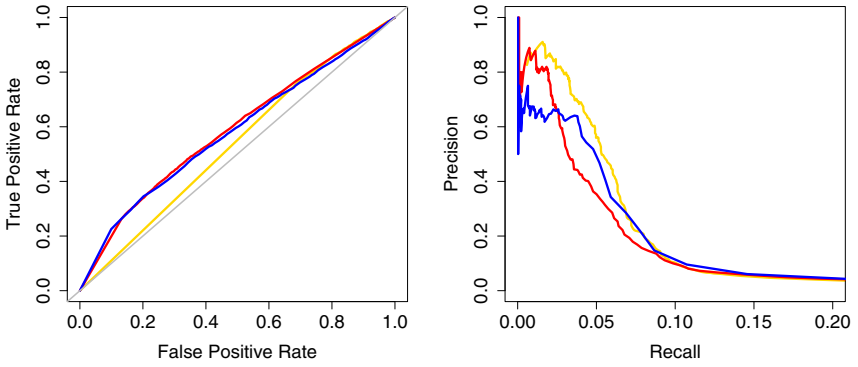

Fig. 7.6 ROC curves (left) and Precision/Recall curves (right) evaluating performance of the correlation (yellow), partial correlation (red), and full partial correlation (blue) methods of network inference described in the text.

equally well, and better than the correlation method. However, none of the methods perform in a particularly stellar fashion, which would seem to suggest that correlation between TF and target expression profiles – whether unadjusted or adjusted for the effects of other TFs – is not an especially strong global indicator for gene regulation in this data.

However, from the standpoint of biology, interest typically centers much more on the prediction of *new* interactions, rather than the recovery of old interactions. Accordingly, we also show in Figure 7.6 a plot of precision (i.e., the fraction of predicted links that are true) versus recall (i.e., the fraction of true links that are correctly predicted) for each of the three methods.[7] All three methods are seen to share a small common region of quite high precision, with the correlation method arguably holding a slight advantage over the two partial correlation methods. On the other hand, in the same region, all three methods show abysmally low recall. Faith et al. [140] have argued that the low recall is due primarily to limitations in both the number and the diversity of the expression profiles produced by the available experiments.

Nevertheless, if we concentrate on the region of high precision, we find that we are able to accurately predict a number of TF/target interactions that were unknown (at least prior to the analysis of Faith et al. [140] with this same data). For example, Figure 7.7 shows the targets predicted by the correlation method, using a threshold

[7] Note that the quantities on the axes of the two plots in Figure 7.6 are related, but not identical. Let TP and FP denote the number of true and false positive predictions, and TN and FN, the number of true and false negative predictions. Then the true positive rate is given by $TPR = TP/(TP+FN)$, and the false positive rate, by $FPR = FP/(FP+TN)$. The true positive rate (FPR) is the same as what is meant by 'recall.' On the other hand, the 'precision' – what is also sometimes called 'positive predictive value' – is given by $PPV = TP/(TP+FP)$.

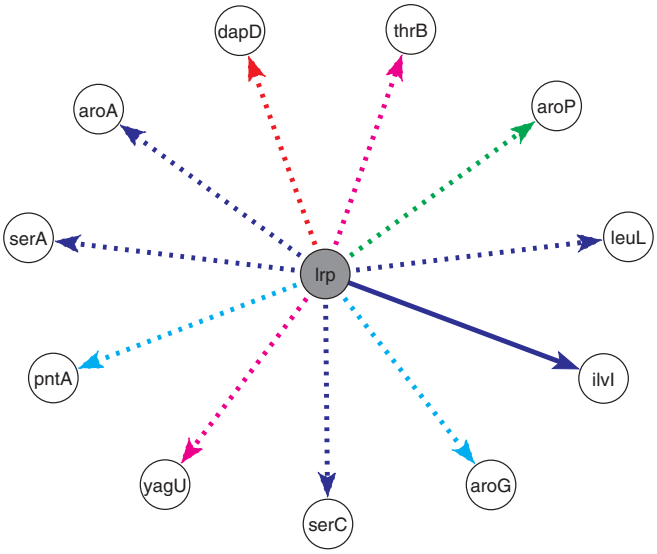

Fig. 7.7 Gene targets predicted by the correlation method for the transcription factor *lrp*, at 60% precision. Patterns and colors of arcs are described in the text.

corresponding to 60% precision[8] in the plot of Figure 7.6, for the transcription factor *lrp*, which is a known regulator of various metabolic processes in *E. coli*. A total of 11 gene targets of *lrp* were predicted. Of these, five (in blue) are indicated to be true positives according to RegulonDB, and one (in green) is indicated to be a false positive. In Faith et al. [140], experimentation was done to directly verify many of the predictions of those authors, including all but one (shown with a solid edge) of the 11 here. All ten of the predictions tested (shown with dotted edges) were found to be true predictions, including the apparent false positive indicated by RegulonDB. Five (in magenta, cyan, and red) of the ten predicted gene targets tested were new, with four of them (in magenta and cyan) present in RegulonDB – but not as targets of *lrp*, and one of them (in red) not present. Of the four present in RegulonDB, two of them (in magenta) were genes previously known to be regulated by at least one other TF, while the other two (in cyan) had no previously known regulator.

7.4 Tomographic Network Topology Inference

In the field of imaging, *tomography* refers to the process of imaging by sections. A number of familiar examples of tomographic imaging are found in the field of medical imaging, where a combination of physical devices (e.g., X-ray or MRI scanners)

[8] A p-value threshold of 10^{-125} was needed to obtain this precision, rendering an FDR-based correction for multiple testing relatively pointless.

and computers are used to create images of internal aspects of the human body (e.g., cross-sections of the head or torso) through reconstruction algorithms that exploit mathematical expressions relating the externally-acquired data to the unknown internal structure. *Tomographic network topology inference* is named in analogy to tomographic imaging and refers to the inference of 'interior' components of a network – both vertices and edges – from data obtained at some subset of 'exterior' vertices. Here 'exterior' and 'interior' are somewhat relative terms and generally are used simply to distinguish between vertices where it is and is not possible (or, at least, not convenient) to obtain measurements. For example, in computer networks, desktop and laptop computers are typical instances of 'exterior' vertices, while Internet routers to which we do not have access are effectively 'interior' vertices.

The problem of tomographic network topology inference is just one instance of what are more broadly referred to as problems of *network tomography*. The term was coined by Vardi [386] and originally was used to describe problems in the context of computer network monitoring, in which aspects of the 'internal' workings of the network, such as the intensity of traffic flowing between vertices, are inferred from 'external' measurements. The problem of traffic matrix estimation, described briefly in Example 2.3, is another such problem, which we will examine in detail later in Section 9.3. Surveys on network tomography may be found in Coates, Hero, Nowak, and Yu [96] and Castro et al. [78].

With information only obtainable through measurements at 'exterior' vertices, the tomographic network topology inference problem can be expected to be quite difficult in general. This difficulty is primarily due to the fact that, for a given set of measurements, there are likely many network topologies that conceivably could have generated them, and without any further constraints on aspects like the number of internal vertices and edges, and the manner in which they connect to one another, we have no sensible way of choosing among these possible solutions.

Our problem is thus an example of what is known as an 'ill-posed inverse problem' in mathematics – an inverse problem, in the sense of inverting the mapping from internal to external, and ill-posed, in the sense of that mapping being many-to-one. In order to obtain useful solutions to such problems, typically additional information is incorporated into the problem statement by way of model assumptions on the form of the internal structure to be inferred. For the tomographic inference of network topologies, a key structural simplification has been the restriction to inference of networks in the form of trees. In fact, nearly all work in this area to date has focused on this particular version of the problem, with exceptions seeming to mainly involve techniques for obtaining more general graphs by merging multiple trees (e.g., Huson and Bryant [210], Coates, Rabbat, and Nowak [97]).

In this section, we will therefore focus on the tomographic inference of tree topologies. Taken as a whole, the amount of work on this topic would appear to easily dwarf that on the topics in Sections 7.2 and 7.3 combined. There are two main areas in which such work may be found: in biology and the inference of phylogenies, and in computer network analysis and the identification of logical topologies (i.e., as opposed to physical topologies). The first area is decidedly older as a field, with roots going back as far as the late 1950's, while the second area goes back no more

than a decade ago, although it has been quite active during that time. Phylogenetic inference is concerned with the construction of trees (i.e., phylogenies) from data, for the purpose of describing evolutionary relationships among biological species. For a comprehensive overview of this area, see the book by Felsenstein [141]; also useful, and shorter, are the two surveys by Holmes [204, 205]. In computer network topology identification, the goal is to infer the tree formed by a set of paths along which traffic flows from a given origin Internet address to a set of destination addresses. Castro et al. [78, Sec. 4] provide a survey of this area.

Despite the fact that there are very different systems, tools, and sciences underlying these two areas, there are actually many common aspects to the structure of the underlying tomographic inference problems, and similarly, to the statistical solutions proposed – although, interestingly, such commonalities do not seem to have been remarked upon widely to date. We will try to emphasize this commonality, wherever possible, in our statement of the problem and exposition of proposed solutions. However, our illustrations and the case study in Section 7.4.5 will emphasize computer network applications, which are somewhat simpler and more self-contained than phylogenetic inference problems, and which also have been comparatively free of the sometimes substantial contention that has traditionally accompanied phylogenetic inference in biology.

7.4.1 Tomographic Inference of Tree Topologies

An (undirected) *tree* $T = (V_T, E_T)$ is a connected graph with no cycles. A *rooted tree* is a tree in which a vertex $r \in V_T$ has been singled out. The subset of vertices $R \subset V_T$ of degree one are called the *leaves*; we will refer to the vertices in $V \setminus \{\{r\} \cup R\}$ as the *internal vertices*. The edges in E_T are often referred to as *branches*. A *binary tree* – also called a *bifurcating tree* – is a rooted tree in which, in moving from the root towards the leaves, each internal vertex has at most two children. We will restrict our attention here almost entirely to binary trees. Trees with more general branching structure can always be represented as binary trees and, indeed, methods for their inference can be built upon inferential methods for binary trees (e.g., Duffield, Horowitz, Lo Presti, and Towsley [128]).

Figure 7.8 shows a schematic representation of a binary tree $T = (V_T, E_T)$, annotated to illustrate our tomographic network inference problem. The five leaf vertices $R = \{1,2,3,4,5\}$, in yellow, are known to us, but the internal vertices $\{i_1, i_2, i_3\}$, in green, are not. Nor are the branches known to us. The root vertex r, in blue, may or may not be known to us, but its existence is assumed in imposing a rooted tree structure upon the unknown network graph we seek to infer. In addition, we associate with each branch a weight w, which we may or may not wish to infer.

We define our tomographic network inference problem as follows. Suppose that for a set of N_l vertices – possibly a subset of the vertices V of an arbitrary graph $G = (V, E)$ – we have n independent and identically distributed observations of some random variables $\{X_1, \ldots, X_{N_l}\}$. Under the assumption that these vertices can be

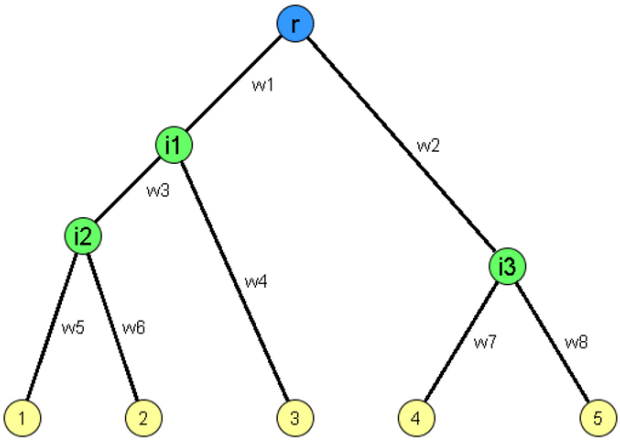

Fig. 7.8 Schematic representation of a binary tree in association with the tomographic network inference problem. Measurements are available at the leaves $1, 2, 3, 4$, and 5, in yellow. The internal vertices, i_1, i_2, and i_3, in green, and possibly the root r, in blue, are unknown, as are the branches joining the various vertices.

meaningfully identified with the leaves R of a tree T, we aim to find that tree $\widehat{T}$ in the set $\mathscr{T}_{N_l}$ of all binary trees with N_l labeled leaves that best explains the data, in some well-defined sense. If we have knowledge of a root r, then the roots of the trees in $\mathscr{T}_{N_l}$ will all be identified with r. In some contexts we may also be interested in inferring a set of weights for the branches in $\widehat{T}$.

Example 7.4 (Binary Measurements: Multicast Probes). As an illustration, consider the case where the random variables X_i are binary and each of the n measurements therefore consists of an N_l-tuple of zeros and ones. An example of such measurements is found in the context of computer networks, in the use of so-called multicast probes. Such a probe consists of a small packet of information, sent simultaneously from a single origin Internet address to multiple destination addresses. On the way towards its destinations, at the first intermediate routing device encountered for which destinations lie along multiple paths, the packet is effectively copied, and a copy is sent along each path. This action is continued as the routes toward the leaves of the corresponding multicast tree continue to branch outwards. In Figure 7.8, for example, copies would be made at each of the interior vertices i_1, i_2, and i_3.

It is possible, however, that along the way some of these packets will be 'lost' – for instance, they can be dropped as part of the manner in which Internet systems keep traffic levels and queuing delays in check. If a packet is lost at a given intermediate router, none of the destinations along paths continuing from that router will receive the packet. For example, in the tree shown in Figure 7.8, a packet sent from the vertex r that was forwarded by vertex i_1 but was lost by vertex i_2 would

not be received by vertices 1 and 2, but could[9] be received by vertex 3. An N_l-tuple $\{X_1,\ldots,X_{N_l}\}$ of binary measurements is used to indicate whether each of N_l destination addresses did (i.e., 1) or did not (i.e., 0) receive a copy of the packet. A set of n measurements corresponds to the results of sending n such packets.

Note that it is natural here to think of the leaf observations $\{X_1,\ldots,X_{N_l}\}$ as arising through the result of a process originating at the root of the underlying tree. Specifically, for a multicast tree T, each vertex $j \in V_T$ can be assigned a binary variable X_j indicating whether or not it received a copy of the packet. The resulting process $\{X_j\}_{j\in V_T}$ will then have certain hereditary constraints to it. More precisely, for a subset $U \subset R$ of the leaves, denote by $a(U)$ the index of that vertex which is the closest common ancestor of the vertices in U. Similarly, for an internal vertex j, denote by $d(j)$ the indices of all immediate descendants (i.e., children) of j. Then $X_{a(U)} = 0$ implies that $X_j = 0$ for all $j \in U$, while if $X_j = 1$ for at least one $j \in d(k)$, then $X_k = 1$.

We will see that this type of structure can be exploited in designing methods of inference. □

Binary measurements also can arise in phylogenetic tree inference. For example, DNA sequence data are often used to provide measurements on species, but sometimes the four DNA bases $\{A,G,C,T\}$ are grouped in pairs $\{A,G\}$ and $\{C,T\}$ (i.e., by purines and pyrimidines, respectively) and simply coded 0 and 1. An N_l-tuple of measurements then indicates whether each of N_l species being studied had a purine or pyrimidine at a given location in the genome. A set of n measurements corresponds to such N_l-tuples at n different locations. The notion of a tree-based process, generating sequence data at the leaves, is also common in this context, where it is motivated by the theory of evolution, as we will discuss briefly later in Section 7.4.3.

There are a number of classes of methods that have been proposed for tomographic inference of tree topologies. They differ in, for example, how the data are utilized (e.g., all or just part), the criteria for assessing the merit of a candidate tree $T \in \mathscr{T}_{N_l}$ in describing the data, and the manner in which the space $\mathscr{T}_{N_l}$ is searched in attempting to find a tree(s) best meeting this criteria. We will focus on two popular classes of methods: (i) those based on hierarchical clustering and related ideas, and (ii) likelihood-based methods.

In the methods of both of these approaches, although the technical infrastructure can be quite different, the underlying idea is essentially the same. Specifically, they seek to exploit the supposition that, the closer two leaves are in the underlying tree, the more similar their observed characteristics (i.e., the measurements) will be. For example, for computer networks, in the case of the multicast measurements just described, in Example 7.4, it is expected that two destination addresses will be more

[9] It *would* be received if there are no additional routers along the branch from i_1 to 3 that might lose it. If there are additional routers, then it is not certain that the packet would be received. But with the topology dictated by a given choice of leaves and root, we are effectively blind to the number of additional such routers present along any given branch. This issue lies at the heart of the distinction between physical and logical Internet topologies, and will also be seen to be relevant to constructing inferential methods.

likely to both receive or not receive a copy of the packet if they share more of the same path from the root. Similarly, two biological species are presumed to share more of their genome if they split from a common ancestor later in evolutionary time.

Note that regardless of the methodology used, if we truly aim to search over all of $\mathscr{T}_{N_l}$, we are faced with a daunting computational task. In particular, the space $\mathscr{T}_{N_l}$ of (semi)labeled, rooted, binary trees is found to have

$$3 \times 5 \times 7 \times 9 \times 11 \times \ldots \times (2N_l - 3) = (2N_l - 3)!! \tag{7.39}$$

elements (e.g., Felsenstein [141, Ch. 3]). Using Stirling's approximation, it can be shown that this quantity grows like $(N_l - 1)^{N_l - 1}$. Accordingly, many of the specific tree-selection problems posed in this area have been shown to be *NP*-hard, although sometimes clever use of efficient search strategies can make exhaustive search computationally feasible in problems of relatively small, but non-trivial, magnitude. An example is the 'maximum parsimony' problem in phylogenetic inference, which seeks to construct a tree for data involving the minimum number of necessary evolutionary changes, and for which the 'branch-and-bound' algorithm can be used to greatly speed up the search of $\mathscr{T}_{N_l}$. See Felsenstein [141, Ch. 5]). In general, however, exhaustive search is unrealistic and alternative approaches, based on greedy or randomized search algorithms, are utilized.

7.4.2 Methods Based on Hierarchical Clustering

Given a set of N_l objects, recall from our discussion in Section 4.3.3.1 that hierarchical clustering is concerned with the grouping of those objects into hierarchically nested sets of partitions, based on some notion of their (dis)similarity. And, in particular, recall that generally these nested partitions are represented using a tree. The usage of hierarchical clustering we saw in Section 4.3.3.1 was for the purpose of graph clustering. But it also is a natural tool to use for the tomographic inference of tree topologies, where we treat the N_l leaves as the 'objects' to be clustered and the tree corresponding to the resulting clustering as our inferred tree $\widehat{T}$. It is perhaps worth emphasizing that, whereas in standard applications of hierarchical clustering the goal often ultimately is to select just one of the nested partitions to represent the data, here the tree corresponding to the entire set of partitions is our focus.

Recall that hierarchical clustering methods require some notion of (dis)similarity, which is usually constructed from the observed data, but sometimes measured directly. In the context of our tomographic inference problem, it is logical to use the n observations of the random variables $\{X_1, \ldots, X_{N_l}\}$ at the leaves to derive an $N_l \times N_l$ matrix of (dis)similarities. In principle, standard notions of distance between vectors may be used, followed by an off-the-shelf hierarchical clustering algorithm of choice. See Gordon [185] or Jain, Murty, and Flynn [213], for example, for extensive reviews. In practice, however, it can be important to tailor the notion of

(dis)similarity to the tomographic inference problem at hand. We illustrate through the following variant of agglomerative hierarchical clustering for the multicast measurements of Example 7.4.

Example 7.5 (Agglomerative Hierarchical Clustering for Multicast Data). Ratnasamy and McCanne [323] observe that the observed rate of shared losses of packets should be fairly indicative of how close two leaf vertices (i.e., destination addresses) are on a multicast tree T. They propose a greedy, clustering-based algorithm for computer network topology identification that exploits this observation. Duffield, Horowitz, Lo Presti, and Towsley [128] formalize and extend their algorithm and establish certain of its properties, and it is their notation that we adopt here.

An important observation in this problem is that there are two different types of shared loss between a pair of leaf vertices $\{j,k\}$ – termed 'true' and 'false' shared loss by Ratnasamy and McCanne [323]. The true shared losses are due to loss of packets on the path common to the vertices i and j, while the false shared losses are due to loss on the parts of paths following after the closest common ancestor, $a(\{j,k\})$. For example, in the tree of Figure 7.8, true shared losses for the leaves 1 and 3 would be losses incurred on the path from r to the internal vertex i_1; false shared losses would refer to cases where packets were lost separately on the two paths from i_1 to the vertices 1 and 3, respectively. Since the net shared loss rate (i.e., the fraction of packets commonly lost to j and k) includes the contribution of both types of losses, it can be misleading to use this number as a similarity. Fortunately, it is possible to obtain information on the true loss rates from these net loss rates, through the use of a simple packet-loss model.

Specifically, consider the cascade process $\{X_j\}_{j\in V_T}$ described previously. We assume $X_r = 1$. For each internal vertex k, if $X_k = 0$, then $X_j = 0$ for all $j \in d(k)$. If not, then we assume

$$\mathbb{P}(X_j = 1|X_k = 1) = 1 - \mathbb{P}(X_j = 0|X_k = 1) = \alpha_j \ , \tag{7.40}$$

where $0 < \alpha_j < 1$ for all j. That is, we specify a Markov-like model down the tree, from the root to the leaves. The quantity $A(k) = \prod_{j\succ k} \alpha_j$ is the probability that a packet is successfully transmitted from r to k, where $j \succ k$ indicates ancestral vertices of k on the path from r. The true shared loss rate for two leaf vertices $j,k \in R$ is then just $1 - A(a(\{j,k\}))$.

Now define $R(k)$ to be the set of leaf vertices in R that are descendants of k, for interior vertices k, and let

$$\gamma(k) = \mathbb{P}\left(\cup_{j\in R(k)}\{X_j = 1\}\right) \tag{7.41}$$

be the probability that at least one of the leaves descended from k receive a packet. Similarly, for an arbitrary subset of vertices U, let

$$\gamma(U) = \mathbb{P}\left(\cup_{k\in U}\cup_{j\in R(k)}\{X_j = 1\}\right) \tag{7.42}$$

be the probability that at least one of the leaves descended from the vertices in U receive a packet. In the case $U = d(k)$, we have $\gamma(U) = \gamma(k)$. It is not difficult to

show that for any $U \subseteq d(k)$, we have the relation

$$[1 - \gamma(U)/A(k)] = \prod_{j \in U} [1 - \gamma(j)/A(k)] \quad . \tag{7.43}$$

Therefore, given the values $\{\gamma(k)\}_{k \in V_T}$, we can find $\{A(k)\}_{k \in V_T}$ as the unique solutions to the appropriate equations (7.43). See Caceres, Duffield, Horowitz, and Towsley [71] and Duffield, Horowitz, Lo Presti, and Towsley [128].

Although the $\gamma(k)$ are not available to us, they can be estimated by the relative frequencies

$$\hat{\gamma}(k) = (1/n) \sum_{i=1}^{n} \left[\prod_{j \in R(k)} x_j^{(i)} \right] \quad , \tag{7.44}$$

where $x_j^{(i)}$ is the observed value of X_j corresponding to the i-th sample. Estimates $\hat{A}(k)$ then follow accordingly. A greedy, agglomerative algorithm may then be defined as follows. For all pairs of vertices $j,k \in R$, the estimated shared loss rate $1 - \hat{A}(a(\{j,k\}))$ is calculated, where the ancestor $a(\{j,k\})$ is implicitly defined through the proposed grouping of j and k. That pair j,k for which this loss rate is maximized are merged and replaced by their corresponding ancestor vertex. The process is then repeated on the $N_l - 1$ remaining nodes (i.e., $a(\{j,k\})$ and $R \setminus \{j,k\}$) and iterated similarly thereafter, until a full tree is formed.

Duffield, Horowitz, Lo Presti, and Towsley [128] provide a formal statement of this algorithm and prove the consistency of the resulting estimator $\widehat{T}$ for recovering a binary multicast tree T, under the model assumptions. These authors also present and examine three extensions of this algorithm to the problem of inferring non-binary trees. □

Hierarchical clustering methods have also been used quite frequently in phylogenetic inference, and in fact were some of the earliest proposed methods for this task. For DNA sequence data, it is standard to use some notion of dissimilarity of the genetic material of pairs of species as input for the clustering algorithm. A dissimilarity of this sort is commonly termed a 'genetic distance,' and methods involving such distances are called 'distance-matrix' methods. There are numerous measures of genetic distance. The simplest is just the fraction of mismatched elements in two (aligned) DNA sequences. More sophisticated measures attempt to correct for various expected characteristics of DNA evolution. These corrections often are done in a model-based fashion, using Markov-like cascade models similar in principle to that described above for the multicast data. Sometimes, the distances obtained from the data are treated as empirical versions of some true, unknown distances (e.g., along the branches of the underlying phylogenetic tree), and the empirical distances are used to estimate the true distances through, for example, least-squares methods. See Felsenstein [141, Ch. 11] for an overview of distance-matrix methods and a detailed discussion of some of the potentially critical issues associated with how genetic distance is defined. The property of consistency has been shown to hold for

certain distance-matrix methods of phylogenetic tree estimation, under appropriate conditions.

7.4.3 Likelihood-based Methods

We have seen that probability models for the leaf random variables $\{X_1,\ldots,X_{N_l}\}$ are sometimes used in defining (dis)similarities in hierarchical clustering methods for tomographic inference of trees. But if we are willing to specify probability models, then we have the potential for likelihood-based methods of inference as well.

Suppose we have specified a conditional density or probability mass function $f(\mathbf{x}|T)$ for the N_l-length vector of random variables $\mathbf{X} = (X_1,\ldots,X_{N_l})^T$, given a tree-topology T. If we assume independence among the n observations $\{\mathbf{x}^{(i)}\}_{i=1}^n$, then the likelihood has the form

$$\mathscr{L}(T) = \prod_{i=1}^{n} f(\mathbf{x}^{(i)}\,|\,T) \quad , \tag{7.45}$$

and the maximum likelihood estimator is simply

$$\widehat{T}_{ML} = \arg\max_{T\in\mathscr{T}_{N_l}} \mathscr{L}(T) \quad . \tag{7.46}$$

However, it is typically the case that we specify a probability model involving additional parameters besides T, that is, $f(\mathbf{x}|T,\theta)$, for some m-length parameter vector θ. For example, θ might contain parameters relating to the evolution of a cascade process $\{X_j\}_{j\in V_T}$ down a tree T. Then the likelihood in (7.45) is an integrated likelihood, in that it takes the form

$$\mathscr{L}(T) = \prod_{i=1}^{n} \int f(\mathbf{x}^{(i)}|T,\theta) f(\theta|T) d\theta \quad , \tag{7.47}$$

where $f(\theta|T)$ is an appropriately defined distribution on θ, given T. Depending on the nature of their integrands, the integrals in (7.47) may or may not lend themselves well to computational evaluation.

An alternative that can be more computationally tractable, or that can be used when we cannot or do not wish to specify a distribution $f(\theta|T)$, is to define $\widehat{T}$ through maximization of a profile likelihood. Specifically, let

$$\mathscr{L}(T,\theta) = \prod_{i=1}^{n} f(\mathbf{x}^{(i)}|T,\theta) \quad , \tag{7.48}$$

and define

$$\hat{\theta}_T = \arg\max_{\theta} \mathscr{L}(T,\theta) \quad . \tag{7.49}$$

Then let

$$\widehat{T}_{PL} = \arg\max_{T \in \mathscr{T}_{N_l}} \mathscr{L}(T, \hat{\theta}_T) \ . \tag{7.50}$$

The quantity $\mathscr{L}(T, \hat{\theta}_T)$ is called the profile likelihood, and $\widehat{T}_{PL}$, the maximum profile likelihood estimator. See Berger, Liseo, and Wolpert [35] for an overview of issues and trade-offs related to the use of integrated versus profile likelihoods.

We illustrate some of these concepts in the context of the multicast data described earlier.

Example 7.6 (Profile Maximum Likelihood Estimation of Multicast Trees). Recall the cascade model for multicast measurements, surrounding equation (7.40), in Example 7.5. The likelihood (7.48), with θ replaced by a vector α containing the parameters $\{\alpha_j\}$, takes the form

$$\mathscr{L}(T, \alpha) = \prod_{i=1}^{n} \prod_{j \in V_T} \eta_j^{(i)} \ , \tag{7.51}$$

where

$$\eta_j^{(i)} = \begin{cases} 1, & \text{if } j = r \ , \\ \alpha_j^{x_j^{(i)}} (1-\alpha_j)^{1-x_j^{(i)}}, & \text{if } x_{pa(j)}^{(i)} = 1 \ , \\ 1, & \text{if } x_{pa(j)}^{(i)} = 0 \ , \end{cases} \tag{7.52}$$

and $pa(j)$ denotes the parent of j in T. Duffield, Horowitz, Lo Presti, and Towsley [128] propose the profile maximum likelihood estimator

$$\widehat{T}_{PL} = \arg\max_{T \in \mathscr{T}_{N_l}} \mathscr{L}(T, \hat{\alpha}_T) \ , \tag{7.53}$$

where

$$\hat{\alpha}_T = \arg\max_{\alpha} \mathscr{L}(T, \alpha) \ , \tag{7.54}$$

and prove its consistency under appropriate conditions.

Implementation of this estimator, however, is non-trivial. For example, consider the estimation of α, a parameter vector of potentially quite large dimension. An efficient, recursive estimation algorithm is proposed by Caceres, Duffield, Horowitz, and Towsley [71]. These authors show that the elements α_k may be recovered uniquely from the values $A(k)$, through the relation

$$\alpha_k = A(k)/A(pa(k)) \ . \tag{7.55}$$

And recall that the $A(k)$ may be obtained from the values $\gamma(k)$ through equation (7.43). Accordingly, estimates, say $\tilde{\alpha}_k$, of the α_k may be obtained from the estimates $\hat{\gamma}(k)$ of the $\gamma(k)$, defined in (7.44). Although the resulting estimator $\tilde{\alpha}$ is not strictly the maximum likelihood estimate, Caceres et al. show that, when $\alpha_k \in (0,1)$ for all k, we nevertheless have $\tilde{\alpha} = \hat{\alpha}$ with high probability.

As for the optimization of the profile likelihood $\mathscr{L}(T, \hat{\alpha}_T)$ in T, Duffield, Horowitz, Lo Presti, and Towsley [128] seem to have performed this optimization

only in simulation and for trees with very small values of N_l, such as $N_l = 3$ and 5, for which they were able to conduct an exhaustive search of $\mathscr{T}_{N_l}$. For topologies of less trivial sizes, presumably either greedy techniques (like those mentioned below) or MCMC-based techniques (like that we describe later in Section 7.4.5) may be adopted. □

Likelihood-based methods are also used in phylogenetic inference. Similar to the case of our multicast measurements, it is common in this context to assume independence between measurements $\mathbf{x}^{(i)}$ (e.g., between locations in the measured DNA sequence) and, as was remarked earlier, to use Markov-like cascade models to capture the effects of evolution through time. A simple example of such a model, in the case of binary measurements (e.g., when the DNA base characters are coded simply as purines or pyrimidines), is a symmetric model of change between the two states of zero and one, in traversing from one end of a branch to the other, according to the probability

$$\mathbb{P}(X_j = 1 \,|\, X_k = 0) = \frac{1}{2}\left(1 - e^{-2w_j}\right) \quad , \tag{7.56}$$

where $j \in d(k)$ and w_j is the length of the branch leading to j. More sophisticated models, for various types of molecular data, are discussed in Felsenstein [141, Chs. 13–15].

The likelihood corresponding to this model is expressed as

$$\mathscr{L}(T, \mathbf{w}) = \prod_{i=1}^{n} \sum_{\{x_j^{(i)}\}_{j \in V_T \setminus R}} \prod_{j \in V_T} \eta_j^{(i)} \quad , \tag{7.57}$$

where

$$\eta_j^{(i)} = \begin{cases} \frac{1}{2}\left(1 - e^{-2w_j}\right), & \text{if } x_j^{(i)} \neq x_{pa(j)}^{(i)} \ , \\ \frac{1}{2}\left(1 + e^{-2w_j}\right), & \text{if } x_j^{(i)} = x_{pa(j)}^{(i)} \ . \end{cases} \tag{7.58}$$

Note that the expression in (7.57) is similar to that in (7.51), except for the presence of an additional summation term between the two products. This summation is over all possible ways that the states zero or one can be assigned to the internal vertices of the tree T, and is absent in the case of multicast data because of the hereditary constraints on the process $\{X_j\}_{j \in V_T}$. In general, the presence of such product-sum combinations can be problematic from a computational perspective. However, in this particular case, the likelihood can be calculated efficiently using a dynamic programming algorithm – the so-called pruning algorithm – proposed by Felsenstein [142]. Working recursively from the leaves towards the root, components of the likelihood are computed on sub-trees and combined in a clever fashion to yield the likelihoods on larger sub-trees containing them, until the likelihood for the entire tree is obtained.

Because the branch-lengths $\mathbf{w}$ are meaningful and important in a phylogeny, the maximum likelihood approach aims to find the pair $(\widehat{T}, \hat{\mathbf{w}})$ that maximizes (7.57). This problem was only recently shown by Chor and Tuller [82] to be *NP*-hard, although it was widely suspected to be so for some time. In practice, a profile max-

imum likelihood method typically is used, as described, for example, in Felsenstein [143]. For a given tree T, optimal branch lengths $\hat{\mathbf{w}}_T$ can be calculated using an algorithm that generalizes the pruning algorithm mentioned above. The space $\mathscr{T}_{N_l}$ of trees T is then explored in a greedy fashion. For example, Felsenstein [143] proposes an algorithm that grows a tree by sequentially adding species one at a time, where the branch is placed in a position optimal at the current stage of the calculations, and which pauses between each new addition to explore the possibility of 'local re-arrangements.' This algorithm will not search all of tree space, of course, but it is argued that it will explore at least $2N_l^2 - 9N_l + 8$ different topologies.

In general, maximum likelihood inference of tree topologies may also be pursued through the use of MCMC, such as we will do in the case study of Section 7.4.5. And MCMC is critical to the use of Bayesian methods to tomographic inference of trees. With a likelihood in hand, if we are willing and able to specify prior distributions for trees T and parameters θ, then inference may be based upon the posterior $\mathbb{P}(T|\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)})$. A sizable sub-literature exists on Bayesian methods for phylogenetic inference. See the review by Huelsenbeck, Ronquist, Nielsen, and Bollback [207], for example, and also Felsenstein [141, Ch. 18]. Bayesian methods also have been suggested in the computer network context. See Duffield, Horowitz, Lo Presti, and Towsley [128], for example, in the case of multicast measurements. For a prior on the space $\mathscr{T}_{N_l}$ of trees, the uniform distribution is a common choice. See Felsenstein [141, Ch. 18] for a discussion of the choice of prior in the context of phylogenetic inference, including various non-uniform priors. Felsenstein also provides some discussion on the important issue of appropriate design of MCMC algorithms to ensure effective exploration of the space of trees $\mathscr{T}_{N_l}$.

7.4.4 Summarizing Collections of Trees

All of the material so far in this section has focused on methods for producing a single inferred tree $\widehat{T}$. That is, we have focused on roughly the equivalent of a point estimate in standard statistical inference. But a much richer inferential statement would involve producing a collection of trees, in the spirit of an interval estimate. In practice, such collections arise in various ways, such as listing a number of trees of nearly maximum likelihood, rather than simply a single maximum-likelihood tree, or from bootstrap re-sampling in an effort to assess the variability in an inferred tree, or from MCMC sampling of an appropriate posterior distribution on $\mathscr{T}_{N_l}$.

As an illustration, consider the process of bootstrapping in tomographic tree inference, which has been studied extensively in the context of phylogenetic trees (e.g., see Felsenstein [141, Ch. 20]). The essence of the basic method in this context is identical to its implementation in more standard problems, which is described in Efron and Tibshirani [131], for example. Given a sample $\{\mathbf{x}^{(i)}\}_{i=1}^n$ of size n, assumed to consist of independent and identically distributed draws from an underlying distribution, we draw a set of B bootstrap samples, say $\{\mathbf{x}^{i,b}\}_{i=1}^n$, for $b = 1,\ldots,B$, each of which derives from random sampling with replacement from the original sam-

ple. For each bootstrap sample, we apply the same tree-inference method used to construct our original estimate $\widehat{T}$ from the original data, resulting in a collection of estimates $\{\widehat{T}^{(b)}\}_{b=1}^{B}$.

Given such a collection of trees, how can we usefully summarize the information therein? Of course, what is 'useful' will depend to no small extent on the questions being asked. But two common ways to think about summarizing collections of trees are by concentrating on (i) how they are similar and (ii) how they are different.

For thinking about similarities among trees, consensus methods are popular. A *consensus tree* is a single tree that aims to summarize the information in a collection of trees in a 'representative' manner. There are many methods that have been proposed for defining consensus trees. For example, Margush and McMorris [268] define a class of consensus trees, called M_l-trees, indexed by a parameter $l \in [0.5, 1]$. An M_l-tree contains all groups of leaves $U \subseteq R$ (i.e., as determined by the branching) that occur in more than a fraction l of the trees in a collection. In the case of $l = 1.0$, we have what is termed a *strict consensus tree*, while for $l = 0.5$, it is called a *majority-rule consensus tree*. The computation of an M_l-tree over a collection of t trees with N_l leaves has traditionally been done using $O(tN_l^2)$ algorithms, but Amenta, Clarke, and St. John [9] have more recently proposed a randomized algorithm with $O(tN_l)$ expected run time. A comprehensive summary and classification of these and additional consensus tree methods can be found in the review by Bryant [68].

Note that additional information can be added to a consensus tree in the form of branch weights, where appropriate. For example, in phylogenetic inference, when the collection of trees results from bootstrapping, it is common to annotate each branch in a majority-rule consensus tree with the fraction of bootstrap trees $\widehat{T}^{(b)}$ that contain its corresponding group of leaves.[10]

For thinking about differences among trees, various notions of distance between pairs of trees have been employed. For example, the *symmetric difference,* introduced by Bourque [49] and by Robinson and Foulds [332], works by counting branches. More specifically, given two rooted trees, each branch of each tree is associated with the group of leaf vertices descended from it (similar as to the M_l-tree consensus methods described above), and the symmetric difference is defined to be the number of groups that differ (i.e., that are induced by one tree but not both). It is not hard to show that this quantity forms a proper distance (i.e., a metric) on $\mathscr{T}_{N_l}$. Another popular quantity for quantifying the difference between two trees is the *nearest-neighbor interchange* (NNI) distance, which essentially counts the number of swaps of adjacent branches that must be made to transform one tree into the other.[11] Unfortunately, the task of calculating the NNI distance is computationally daunting. Li and Zhang [257] have shown a similar problem is *NP*-complete. In

[10] These annotations, intended as estimates of the evidence in support of each branch, have been noted to have a particular type of bias to them. Various methods, based on more sophisticated resampling schemes, have been suggested for correction of this bias. See Felsenstein [141, Ch. 20], for details and references.

[11] More formally, it is associated with counting the number of 'rotation' operations, of two 'sibling' branches about their common vertex, needed to move from one tree to the other in $\mathscr{T}_{N_l}$.

practice, greedy approximation algorithms are used (e.g., see Felsenstein [141, Ch. 4]). NNI distance too can be shown to be a metric on $\mathscr{T}_{N_l}$.

Perhaps not surprisingly, relationships can be demonstrated between certain consensus measures and difference measures. For example, Barthélemy and McMorris [21] show that, if we define a *median tree* in a collection to be a tree T whose total distance – based on the symmetric difference – to all other trees is a minimum, then this tree is equivalent to the majority-rule tree, when the number of trees t in the collection is odd.

In general, as Holmes [205] has remarked, there remains much to be done in the area of formal methods of comparisons of trees (not to mention more general graphs), and in the usage of such methods for higher-level tasks of statistical modeling and inference.

7.4.5 Case Study: Computer Network Topology Identification

We have already encountered, through the several examples running throughout this section, an illustration of the basic structure of a computer network topology identification problem and its solution, in the context of the binary multicast-based measurements of packet loss. Here we present an extended illustration of the same problem, but based on somewhat different measurements of a continuous type. The data and analysis we describe are taken from the work of Coates and Nowak and their colleagues (e.g., Coates et al. [95], Castro, Coates, and Nowak [79]).

In properly functioning computer networks, the loss of packets should be a relatively rare phenomenon (e.g., with a rate of less than 2%). While this fact is good news for network users, it also means that multicast-based methods can have trouble discerning the underlying topology in such circumstances due to lack of information. An alternative is to utilize the time delays between when packets are sent and when they are received, which yields a set of measurements that have been found to provide greater discernibility.

For example, Coates et al. [95] propose a measurement scheme they call 'sandwich probing,' which sends a sequence of three packets, two small and one large. The large packet is sent second, after the first small packet and before the second small packet. The two small packets are sent to one of a pair $\{i, j\}$ of leaf vertices, say i, while the large packet is sent to the other, that is, j. The basic idea is that the large packet induces a greater delay in the second small packet, compared to that of the first, and that the difference in delays of the two small packets will vary in a manner reflective of how much of the paths are shared from the origin to i and j. For example, in Figure 7.8, we expect a longer difference in delays for the two small packets when they are sent to vertex 1, and the large, to vertex 2, as compared to when the large is instead sent to vertex 3. This reasoning follows from the fact that in the former case the second small packet lingers behind the large packet from r until i_2, while in the latter case, it does so only from r until i_1.

Figure 7.10 (at the top) shows the logical topology, during a period of 2001, of that portion of the Internet leading from a desktop computer in the Electrical and Computer Engineering Department at Rice University to similar machines at ten other university locations. Specifically, two of the destination machines were located on a different sub-network at Rice, two at separate locations in Portugal, and six at four other universities in the United States. The topology shown here was itself measured, using the `traceroute` tool described in Section 3.5.2, and therefore is not a perfect representation of the true logical topology, but is an acceptable proxy for a 'groundtruth.'[12]

Coates et al. [95] conducted an experiment in which, over a period of eight minutes, a sandwich probe was sent to a randomly chosen pair of these ten destinations, once every 50 *ms*. These pairs included self-pairs (i.e., where the two small packets and the large packet were all sent to the same destination), which allows for a type of calibration in interpreting the measurements. For each probe, the difference in delay measured in the two small packets was recorded. The experiment resulted in a total of $9,567$ measured delay differences, after accounting for probes with packet losses, of which there were very few. Figure 7.9 shows an image representation of the mean delay differences, symmetrized to eliminate variations within each pair due to receipt of the small packets versus the large packet. The hierarchical relationship among the destinations in the logical topology in Figure 7.10 is clearly evident in the relative magnitudes of the mean delay differences. This association can be used to infer the topology from the delay differences.

More specifically, consider the following model. For each pair of destinations $i, j \in R$, let x_{ij} be the average delay difference, over the course of the experiment, where the first index, i, refers to the destination receiving the two small packets, and the second, j, the large packet. The central limit theorem suggests that the x_{ij}, as averages, will be approximately Gaussian in distribution, say with means $\quad_{ij}$ and variances σ_{ij}^2. In addition, based on the construction of the experiment, an assumption of independence between the x_{ij} is not unreasonable. Coates et al. [95] use these basic Gaussian model components to derive two methods of topology inference for this data, the results of which are shown in Figure 7.10.

The first inferred topology derives from the agglomerative likelihood tree (ALT) method of Castro, Coates, and Nowak [79], which is basically a hierarchical clustering algorithm with a likelihood-based measure of similarity. More specifically, note that since the expected delay difference $\quad_{ij}$ is ideally the sum of expected delay differences on all branches shared along the paths from the root to the destinations i and j, it is appealing as a measure of similarity between i and j. We obviously do not know the $\quad_{ij}$, but can estimate them from the x_{ij}. Let $\ell_{ij}(\quad) = \log f(x_{ij}|\quad)$ denote the Gaussian log-likelihood for the single variable x_{ij}, as a function of its mean $\quad$, where we have suppressed the dependence on σ_{ij}^2, which we assume to be known

[12] Of course, one may ask, if `traceroute` may be used to measure and construct such topologies, why is there even a need for the tomographic methods described here? In reality, `traceroute` has its own problems, not the least of which is the fact that an increasingly greater fraction of routing devices in the Internet over the years have had their ability to interact with `traceroute` probes 'turned off' by network administrators, in part due to security concerns.

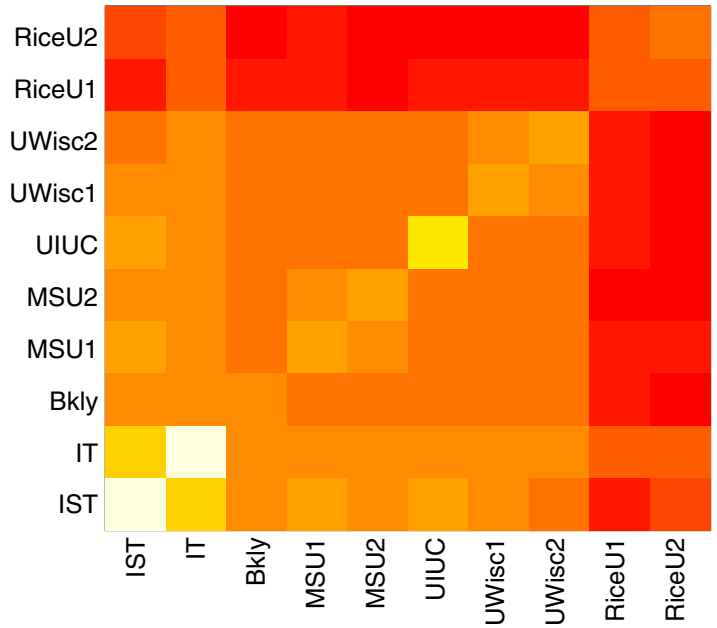

Fig. 7.9 Image representation of pairwise delay differences in the data of Coates et al. [95], with increasing darker red corresponding to lower values, and increasingly brighter yellow, to higher values.

(or, practically speaking, well estimated from the data). Let S be a set of vertices, initialized to be the set of leaves (i.e., $S = R$), and define our similarity measure among leaves to be the estimated mean delay difference

$$\hat{}_{ij} = \hat{}_{ji} = \arg\max \left[\ell_{ij}(\ \ ) + \ell_{ji}(\ \ )\right] \ , \tag{7.59}$$

for all distinct pairs $i, j \in S$. The ALT algorithm merges those two leaves $i, j \in R$ with the largest value $\hat{}_{ij}$, and replaces them in the set S by an ancestor vertex $a(\{i, j\})$. The algorithm then iterates, calculating

$$\hat{}_{kl} = \hat{}_{lk} = \arg\max \sum_{m \in R(k)} \left[\ell_{ml}(\ \ ) + \ell_{lm}(\ \ )\right] \ , \tag{7.60}$$

for all distinct pairs $k, l \in S$, until the set S consists of a single vertex. Recall that $R(k)$ denotes the set of leaf vertices in R descended from an interior vertex k.

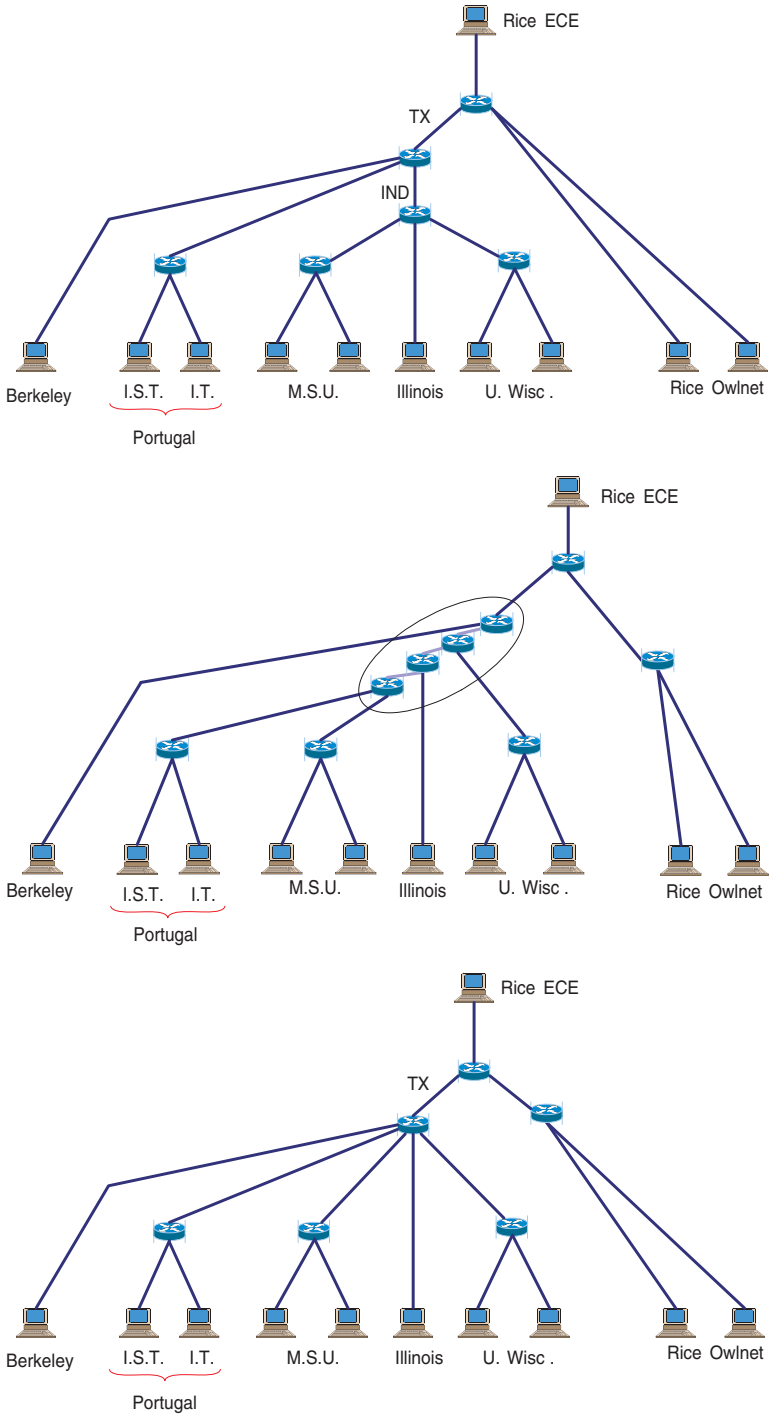

Fig. 7.10 Computer network topologies from the experiment of Castro, Coates, and Nowak [79]. Top: Groundtruth topology, based on `traceroute` measurements. Middle: Inferred topology, based on 'sandwich' probe measurements, using a hierarchical clustering algorithm. Bottom: Inferred topology, based on the same measurements, using a complexity penalized maximum likelihood method. Figures courtesy of Rui Castro.

We note that the ALT algorithm is similar to the unweighted pair group method with arithmetic mean (UPGMA), also known as the average linkage method, which was developed originally in the context of phylogenetic inference by Sokal and Michener [362].

The second inferred topology in Figure 7.10 derives from a complexity penalized maximum (profile) likelihood method, proposed by Coates et al. [95]. More precisely, the topology is defined through the expression

$$\widehat{T} = \arg\max_{T} \left[\log \mathscr{L}(T, \hat{\nu}_T) - \lambda J(T) \right] \ , \tag{7.61}$$

where $\mathscr{L}(T, \nu)$ is the likelihood of the full set of observations, as a function of the tree T and a parameter vector ν, $J(T)$ is a penalty that counts the number of branches in T, and $\lambda \geq 0$ is a user-specified smoothing parameter. The vector $\nu = \nu_T$ contains, for a given tree T, the set of expected delay differences on the individual branches of T, the various sums of which determine the expected delay differences $_{ij}$ on the shared portions of paths to all pairs of leaf vertices $i, j \in R$. The quantity $\hat{\nu}_T$ is the maximum likelihood estimate of ν, conditional on T. The function $J(T)$ acts to encourage the use of parsimonious representations of the data, while λ dictates the trade-off between model goodness-of-fit and model complexity. Together, through the term $-\lambda J(T)$, they essentially act in the role of a (unnormalized) log-prior on T, and $\widehat{T}$, in the role of a *maximum a posteriori* estimator. The optimization in (7.61) is over all trees (i.e., binary and non-binary) with N_l leaf vertices.

Computation of the complexity-penalized maximum likelihood estimate in (7.61) is facilitated by the use of MCMC sampling. Coates et al. [95] propose the use of a standard Metropolis-Hastings algorithm, where the proposed moves from tree to tree are controlled through insertion or deletion of individual interior vertices between two existing vertices. See Coates et al. [95] and Castro, Coates, and Nowak [79] for details. Although this approach is rather simple, compared to many of the MCMC methods in this area, the algorithm appears to work quite well, a fact that the authors attribute in good part to their use of the ALT-inferred tree as a starting value.

Looking at the two inferred topologies, and the groundtruth topology, we can see a number of interesting differences. First, while the ALT topology is, naturally, binary, the maximum likelihood topology is not. The biggest implication of this difference seems to be in the inclusion in the ALT topology of a sequence of four closely spaced consecutive internal vertices where the maximum likelihood topology has only one. In fact, Castro, Coates, and Nowak [79] state that the improvement due to inclusion of these vertices is slight – an order of magnitude smaller than for other branches, and conclude that they are largely an artifact of the binary nature of the underlying hierarchical agglomeration methodology. Second, beyond this artifact, the ALT and maximum likelihood topologies are quite similar. Both, for example, miss the Indiana (IND) router, and both add an additional vertex between the Rice ECE root and the two Rice leaf vertices. Coates et al. [95] speculate that the Indiana router is missed because the connection between the Texas (TX) and Indiana routers is too high-speed to produce a measurable delay difference. Interestingly, upon further examination, the extra vertex between the Rice machines was found to be real

– a so-called switching device that, not being itself a router, would not be detectable by `traceroute` but does indeed produce a *bona fide* branching in the underlying logical topology.

7.5 Additional Related Topics and Reading

We have focused our discussion in this chapter upon three arguably canonical problems in network inference. But various other network inference problems have been considered in the literature as well, arising from specific applications. Two examples are the problem of inferring networks from so-called co-occurrence data, as considered in Rabbat, Figueiredo, and Nowak [318, 319], motivated by measurements in telecommunications and in genetics, and the problem of inferring protein complexes, as considered by Scholtens and Gentleman [343]. Another interesting variation, related to some extent with the former of these two problems, is that of inferring networks with links defined according to the dynamic interactions of elements in a system (e.g., such as of information passing between neurons firing in sequence). See Smith et al. [355] for an example of such 'information flow networks.' Finally, while our presentation of the material on tomographic network topology inference concentrated primarily on the two contexts of computer networks and phylogenetic inference, related to the latter is the problem in biology of inferring coalescent trees, which have to do with the evolution of genes within a population (i.e., below the level of species). See Felsenstein [141, Ch. 26] for an overview and references.

Our treatment of association network inference in Section 7.3 was centered on the use of measures of association defined in terms of linear correlation. But other measures may be preferable, depending upon the nature of the data. For example, Spearman rank correlation and partial correlation might be used as robust alternatives to Pearson correlation (e.g., see Shipley [350, Sec. 3.9] and the discussion in de la Fuente, Bing, Hoeschele, and Mendes [110]). Or a measure capable of summarizing nonlinear association, such as mutual information, might be used, such as Faith et al. [140] have done in the context of gene regulatory network inference.

The Gaussian graphical models we encountered in Section 7.3.3 are, as was mentioned, just one representative of the broader class of graphical models. This class also includes, for example, models based on directed graphs and models for discrete data, both aspects of which have found substantial usage in problems of network inference. For problems of inferring biological networks, the graphical models formalism has been used in various ways with notable success. See, for example, the paper by Friedman, Linial, Nachman, and Pe'er [162], in the context of gene regulatory networks, and that by Sachs et al. [337], in the context of protein-signaling networks. However, with the adoption of such models in their full generality, there comes a number of non-trivial challenges for network inference in terms of model specification, fitting, and selection, especially in large-scale problems. While these issues are now reasonably well studied, they are not necessarily 'solved,' and a care-

ful development of the topic is beyond the scope of this book. Nevertheless, we shall revisit it briefly in Chapter 10.

Lastly, we point out, per our discussion in the introduction in Section 7.1, that while the adoption of a statistical inference framework allows for the *potential* for questions of validation of network inferences to be addressed in a precise and well-grounded manner, we still lack much of the technical infrastructure needed to do so to the extent we currently can with traditional, non-network problems. For example, most work on basic questions of consistency, consensus, robustness, and similar has only been done in the context of trees, in the area of phylogenetic inference, and even there the topic has been nowhere near exhausted. See Holmes [205], for example, for discussion in this vein, with respect to phylogenetic inference, and Ni, Xie, Tatikonda, and Yang [302], for progress in the area of computer network topology inference. More fundamentally, even basic questions regarding notions of accuracy merit substantial additional study. In most large-scale network inference problems an inferred graph $\widehat{G}$ will not fully match an underlying network graph G, and often waiting for increasingly larger sample sizes is unrealistic. Hence, consistency results can be less than compelling. An open area of inquiry, therefore, is how to best characterize the merit of partially accurate estimates. Such characterizations might depend in turn on intended high-level uses of $\widehat{G}$, focusing for instance on the accuracy with which small subgraphs (e.g., network motifs) or paths are recovered.

Exercises

7.1. For a network of your choice, implement and compare some of the various methods for link prediction discussed in Section 7.2, using a cross-validation design like that used in the study presented in Section 7.2.3. Then examine the effect that modifying the mechanism of missingness in your data has on predictive performance. You should change the mechanism of missingness from missing-at-random to missing in some more substantive way. For example, you might allow missingness to be driven by the degrees of the vertices incident to an edge or, if available, vertex attributes.

7.2. For the partial correlation coefficients defined in (7.22), there exist recursive relationships among coefficients at adjacent orders $m-1$ and m that allow for their efficient recursive calculation. Here we derive this relationship for the case of just three random variables, say X, Y, and Z, and $m = 1$. Specifically, you are asked to derive an expression relating $\rho_{XY|Z}$, the partial correlation of X and Y, given Z, to the Pearson product-moment correlations ρ_{XY}, ρ_{XZ}, and ρ_{YZ}.

For convenience, we express the covariance matrix Σ of $(X,Y,Z)^T$, in standardized form (i.e., as a correlation matrix), as

$$\Sigma = \left[\begin{array}{cc|c} 1 & \rho_{XY} & \rho_{XZ} \\ \rho_{YX} & 1 & \rho_{YZ} \\ \hline \rho_{ZX} & \rho_{ZY} & 1 \end{array}\right] \quad ,$$

where the horizontal and vertical lines indicate a block partitioning in the manner of equation (7.24). Use (7.23) to derive a closed-form expression for the partial covariance matrix $\Sigma_{XY|Z}$. Then combine the elements of this matrix according to (7.22) to show that

$$\rho_{XY|Z} = \frac{\rho_{XY} - \rho_{XZ}\rho_{YZ}}{\sqrt{\left(1-\rho_{XZ}^2\right)\left(1-\rho_{YZ}^2\right)}} \quad .$$

7.3. The regression-based approach to inference of a conditional independence graph G in a Gaussian graphical model is motivated by the optimization problem in (7.37). Here we derive the solution to that optimization problem in the case of three variables.

Let $(X,Y,Z)^T$ be a Gaussian random vector with zero mean and covariance matrix Σ, where Σ is defined as in Exercise 7.2. Consider the task of optimally predicting Z as a linear combination of X and Y, using mean-squared error as a cost function. That is, consider the optimization problem

$$\min_{\beta_{ZX},\beta_{ZY}} \mathbb{E}\left[\left(Z - \beta_{ZX}X - \beta_{ZY}Y\right)^2\right] \quad . \tag{7.62}$$

a. Show that under our model assumptions the expectation in (7.62) takes the form

$$1 + \beta_{ZX}^2 + \beta_{ZY}^2 - 2\beta_{ZX}\rho_{ZX} - 2\beta_{ZY}\rho_{ZY} + 2\beta_{ZX}\beta_{ZY}\rho_{XY} \quad .$$

b. Differentiating the expression in part (a), with respect to β_{ZX} and β_{ZY} and setting the resulting expressions equal to zero, show that the vector $(\beta_{ZX},\beta_{ZY})^T$ must satisfy the linear system of equations

$$\begin{pmatrix} \rho_{ZX} \\ \rho_{ZY} \end{pmatrix} = \begin{bmatrix} 1 & \rho_{XY} \\ \rho_{YX} & 1 \end{bmatrix} \begin{pmatrix} \beta_{ZX} \\ \beta_{ZY} \end{pmatrix} \quad .$$

c. Solve the above linear system to show that

$$\begin{pmatrix} \beta_{ZX} \\ \beta_{ZY} \end{pmatrix} = \begin{pmatrix} \frac{\rho_{ZX} - \rho_{ZY}\rho_{XY}}{1-\rho_{XY}^2} \\ \frac{\rho_{ZY} - \rho_{ZX}\rho_{XY}}{1-\rho_{XY}^2} \end{pmatrix} \quad .$$

Compare this solution to the partial correlations $\rho_{ZX|Y}$ and $\rho_{ZY|X}$.

7.4. Consider the problem of tomographic inference of tree topologies.

a. The multicast measurement model described in Examples 7.4 and 7.5 is straightforward to simulate. Using the tree $T = (V_T, E_T)$ shown in Figure 7.8, and probabilities $\{\alpha_j\}_{j \in V_T}$ of your choosing, write a computer program to simulate the cascade process $\{X_j\}_{j \in V_T}$ defined by (7.40), and hence measurements $\{X_j\}_{j \in R}$ at the leaf vertices $j \in R \subset V_T$.

b. Implement the hierarchical clustering algorithm described[13] in Example 7.5 and apply it to the simulated data generated by your work in part (a). Characterize the performance of this algorithm in recovering the underlying tree topology, for various choices of probabilities $\{\alpha_j\}_{j \in V_T}$.

[13] For a formal statement of this algorithm, see Duffield, Horowitz, Lo Presti, and Towsley [128, Fig. 2].