Chapter 2
A Panel of Learning Methods for the Reconstruction of Gene Regulatory Networks in a Systems Genetics Context

David Allouche, Christine Cierco-Ayrolles, Simon de Givry, Gérald Guillermin, Brigitte Mangin, Thomas Schiex, Jimmy Vandel and Matthieu Vignes

Abstract In this chapter, we study different gene regulatory network learning methods based on penalized linear regressions (the Lasso regression and the Dantzig Selector), Bayesian networks, and random forests. We also replicated the learning scheme using bootstrapped sub-samples of the observations. The biological motivation relies on a tough nut to crack in Systems Biology: understanding the intertwined action of genome elements and gene activity to model gene regulatory features of an organism. We introduce the used methodologies, and then assess the methods on simulated "Systems Genetics" (or genetical genomics) datasets. Our results show that methods have very different performances depending on tested simulation settings: total number of genes in the considered network, sample size, gene expression heritability, and chromosome length. We observe that the proposed approaches are

D. Allouche · C. Cierco-Ayrolles · S. de Givry · G. Guillermin · B. Mangin · T. Schiex · J. Vandel · M. Vignes (✉)
UR875 MIA-T, INRA Toulouse, 31326 Castanet-Tolosan, France
e-mail: Matthieu.Vignes@toulouse.inra.fr

D. Allouche
e-mail: David.Allouche@toulouse.inra.fr

C. Cierco-Ayrolles
e-mail: Christine.Cierco-Ayrolles@toulouse.inra.fr

S. de Givry
e-mail: simon.degivry@toulouse.inra.fr

G. Guillermin
e-mail: Gerald.Guillermin@toulouse.inra.fr

B. Mangin
e-mail: Brigitte.Mangin@toulouse.inra.fr

T. Schiex
e-mail: Thomas.Schiex@toulouse.inra.fr

J. Vandel
e-mail: Jimmy.Vandel@toulouse.inra.fr

able to capture important interaction patterns, but parameter tuning or ad hoc pre- and post-processing may also have an important effect on the overall learning quality.

2.1 Introduction, Motivations

One of the central targets of *Systems Biology* is to decipher the complex behavior of a living cell in its environment. The effective behavior of the cell is probably defined through multiple layers of interacting entities including DNA, mRNA, non-coding RNA, proteins, and metabolites. In this chapter, we are interested in the so-called *genetical genomics* approach that combines the power of genetics, through the polymorphism, together with the measurement capabilities of gene expression to decipher a gene regulatory network (GRN), a simplified representation of the gene-level interactions occurring under given conditions. In such a network, vertices represent genes and directed edges represent the direct causal effect of genes over the expression of other genes through gene regulation (which can be activation or repression). Although proteins are often considered as the main vector of such regulations (through transcription factors for example), this simplified view of regulation could also accommodate other regulation effective transcribed molecules acting on transcription levels such as ncRNA genes. Moreover, protein levels are still quite difficult to measure.

By deciphering the set of gene regulations that are acting in a given context, one may be able to identify the most important, possibly indirect, players in the network that are capable of influencing a specific gene expression or phenotype of interest (Yvert et al. 2003), one may also link network structure to associated functional properties (Leclerc 2008; Marbach et al. 2009) and more generally understand the way gene interactions can control the overall cell behavior. A variety of mathematical formalisms, continuous or discrete (Boolean network Thomas 1973), defined over time (ordinary differential equations Bansal and di Bernardo 2007 or dynamic Bayesian networks Rau et al. 2010; Lèbre et al. 2010) or in stationary states (Friedman et al. 1999) have been proposed to represent the complex behavior of known gene regulation networks. In this chapter, we consider different statistical models of gene regulation that have been chosen for their ability to automatically infer gene regulations from expression data. As initially proposed by Jansen and Nap (2001), in order to integrate some causality in the inference process, we do not rely on time series of expression data but follow the so-coined *genetical genomics* (or *Systems Genetics*) path that exploits the possible influence of genetic polymorphisms on genic expression in a population (Aten et al. 2008). In a genetically controlled setting, defined for example by a population of recombinant inbred lines (RILs) producing randomly perturbed polymorphism combinations, the influence and genomic positions of polymorphisms that are observed in the population can be exploited to predict causal influences between gene expressions. The added value of having both genetic polymorphisms and perturbed phenotypic data, the expression level of genes, has already been demonstrated, in particular to infer causality (Zhu et al. 2007). From a statistical point of view, GRN learning is cursed by its high-dimensionality: the number of genes in a typical genome is much larger than the number of samples that can reasonably be produced ever. Existing works that decipher GRN structure based on genetical genomics data have been using Bayesian networks using genetic data as prior information (Zhu et al. 2007) or complex multivariate regression in a structural equation modeling framework with multiple testing, greedy search, and filtering steps (Liu et al. 2008). More recently, a meta-analysis of the output of different statistical methods targeted at learning in high dimension (based on penalized linear regression or penalized Bayesian network structure learning) has been shown to define the best performer (Vignes et al. 2011) on different datasets of simulated genetical genomics data, including up to 1,000 genes.

This chapter follows on this result in different directions: we first exploit the new genetical genomics simulated datasets that were produced by A. de la Fuente and colleagues, described in more detail in Chap. 1 of this book. These new datasets include a variety of different network topologies and include larger sets of genes (up to 5,000 genes for the largest ones), defining very challenging problems both in terms of dimensionality and in terms of computational learning cost. We also sophisticate and extend the set of statistical methods that are tested on these problems. The original methods included gene-by-gene Lasso regressions (Tibshirani 1996) and the Dantzig selector (Candès and Tao 2007) as well as a penalized Bayesian network learning algorithm. We have improved each of them by bootstrapping, in order to offer more reliable ranked list of edges representing regulations. Finally, we also integrated the random forest approach (Breiman 2001). This method has been used in Huynh-Thu et al. (2010) for GRN learning with expression data only. It is of specific interest for its ability to predict directed edges in all cases, compensating for the weaknesses of linear regression models, that only infer causal orientation from linear marker to genes relations, and Bayesian networks which may not allow to orient edges in all situations because of Markov equivalence (Koller and Friedman 2009).

In Sect. 2.2, we present the mathematical models and associated learning methods which have been used to analyze the data. In Sect. 2.3, we then present and discuss the results obtained by each of these methods and conclude.

2.2 Methods

In this section, we detail the statistical models and learning methods we used to tackle the datasets at hand, and how we adapted them to actually learn GNR with an associated edge-specific confidence score.

2.2.1 Data and Notations

The genetical genomics datasets were provided by Alberto de la Fuente and colleagues from CRS4 Bioinformatica (Pula, Italia). These datasets were simulated with the SysGenSim software (Pinna et al. 2011) and are available at the following http://sysgensim.sourceforge.net/datasets.html. Starting from a given network (a directed graph), the software simulates the network behavior using differential equations capturing expression, regulation, and molecule decay. The datasets and simulator are detailed in Chap. 1 of this book.

For each gene regulation network, a dataset is defined by a sample of n RILs which are measured for p bi-allelic markers and p gene expression levels. Every polymorphism is associated with a single gene and may influence either its direct expression (*cis* polymorphism occurring in the regulatory region of the gene) or its ability to regulate other target genes (*trans* polymorphism in the transcribed gene region itself, influencing its affinity with other gene regulatory complexes). A dataset is therefore defined by:

1. a $n \times p$ matrix $\mathbf{e}$ where e_{ij} gives the steady-state expression level of gene j for the RIL individual i (a real number). Each e_{ij} is an observation of the random variable E_j representing the expression level of gene j. $\mathbf{E}$ is the random matrix of all such variables. For a given gene $g \in \{1, \dots, p\}$ we denote by $\mathbf{E}^{-g}$, matrix $\mathbf{E}$ omitting its gth component.
2. a $n \times p$ matrix $\mathbf{m}$ where m_{ij} gives the allelic state of the polymorphism associated with gene j for RIL individual i (a $0/1$-data). Each m_{ij} is an observation of the random variable M_j representing the allelic state of the polymorphism associated to gene j. $\mathbf{M}$ is the random matrice of all such variables.

The dataset generation is controlled by the network size p, the RIL population size n, chromosome size, and gene expression heritability. The chromosome size is controlled by a mean genetic distance (either 1 or 5 cM) between adjacent marker positions, on all five chromosomes. The shorter the genetic distances between two markers, the stronger their genetic linkage. This leads to highly correlated allelic states between neighboring and even close markers. The latter parameter, heritability, is defined for each gene as the ratio of expression variance due to genetic factors over the expression variance when both genetic factors and biological/technical noise is accounted for. A broader distribution for biological noise implies lower gene heritability.

Our goal is to reconstruct the GRN that gave rise to the observed steady-state expression measures. Following the DREAM5 challenge, a prediction is defined by a directed consensus graph, each directed edge being associated with a "confidence score." In practice, all the statistical inference methods we used produce two confidence scores. One score derives from the estimated influence of allelic states on each expression levels and is denoted w_{kl}^{m} for the $k \to l$ edge. A similar score w_{kl}^{e} is obtained by estimating the influence of expression level E_k on expression level E_l.

In the following sections we introduce the most important components of the statistical learning methods we used: bootstrapping, penalized linear regressions, random forests, and Bayesian network structure learning. We detail how they can produce w_{kl}^m and w_{kl}^e scores. Then, in Sect. 2.2.6, we show how one can produce a final network prediction in the form of a ranked list of edges.

2.2.2 Bootstrapping

Bootstrapping (Efron 1981) is a resampling technique for assigning measures of accuracy to sample estimates. It has the advantage of allowing the estimation of the sampling distribution of virtually any statistic using only a simple resampling approach, at the cost of repeated computations.

The general idea of bootstrapping is to randomly draw N_{boot} replicate datasets with the same sample size as the original data. Each of these replicate dataset is obtained by randomly sampling with replacement from the original sample. For each replicate dataset, the model is fitted, and it is then possible to study the statistical properties of the distribution of the considered statistic on all resampled datasets.

In this chapter, the major use of bootstrapping is to contribute to the construction of the so-called "confidence score" of edges in the predicted GRN. However, bootstrapping offers further opportunities. Since bootstrap datasets are obtained by sampling with replacement, each of them is deprived from around $1 - 0.632 = 36.8\,\%$ of the original samples (Efron and Tibshirani 1997). It is possible to use these out-of-bootstrap samples to study the behavior of any given loss function on those samples. This feature is used internally in the random forest approach and allows to avoid overfitting.

A major drawback of bootstrapping is that it roughly multiplies the computational burden by N_{boot}. The loss of 36.8 % of the data in every bootstrap sample may also affect the sharpness of estimates on every resampled dataset.

2.2.3 Penalized Linear Regression Approaches

A natural approach to solve the network inference problem is to consider each gene g individually from the others and consider that its expression value can be represented as a linear function of all other gene expression levels and of all polymorphisms:

$$E_g = \sum_{j=1}^{p} \alpha_{gj} M_j + \sum_{\substack{j=1 \\ j \neq g}}^{p} \beta_{gj} E_j + \varepsilon_g,$$

where α_g is the p-vector of linear effects of polymorphisms on E_g, β_g is the p-vector of linear effects of other expression levels on E_g (we assume $\beta_{gg} = 0$), and ε_g is a Gaussian residual error term.

Parameters α_g and β_g can be estimated for each gene g from matrices $\mathbf{m}$ and $\mathbf{e}$ using a variety of linear regression methods. The main strength of this model lies in its simplicity, leading to only $2p$ parameters to estimate for E_g, a very desirable property for estimation in high-dimensionality settings. More complex linear models including interactions terms between polymorphisms and expression levels would immediately lead to a quadratic number of parameters.

Given the fact that $n \ll p$ and that regulation networks are expected to be sparse, penalized regression methods leading to variable selection such as the Lasso regression (Tibshirani 1996) or the Dantzig selector (Candès and Tao 2007) were chosen to perform the regression.

2.2.3.1 Lasso Penalized Regression

In the linear regression problem, a response variable Y is a linear combination of r regressors $X = (X_1, \dots X_r)$ and Gaussian noise ε:

$$Y = X\theta + \varepsilon$$

Having observed Y and X on a sample of size n and assuming Gaussian distributions, the estimation $\hat{\theta}$ of the parameters θ is obtained by minimizing the residual sum of squares (RSS):

$$\hat{\theta}^{\text{rss}} = \arg\min_{\theta} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{r} x_{ij}\theta_j)^2 = \arg\min_{\theta} \parallel Y - X\theta \parallel_{\ell_2}^2,$$

where y_i and x_{ij} are the observed values of Y and X_j for the ith individual.

The Lasso regression (Tibshirani 1996) penalizes this RSS criteria by the sum of the absolute values of the parameters (their ℓ_1 norm):

$$\hat{\theta}^{\text{lasso}} = \arg\min_{\theta} \parallel Y - X\theta \parallel_{\ell_2}^2 + \lambda \parallel \theta \parallel_{\ell_1} \tag{2.1}$$

This penalization leads to a shrinkage of parameter estimation. More importantly, the shape of the ℓ_1-norm specifically favors the estimation of zero values, leading to a natural variable selection behavior. Shrinkage and selection levels are controlled by the magnitude of the penalty term λ. The Lasso criterion can also be written in its dual form (by Lagrangian transform), which makes the constraint on parameters more explicit:

$$\hat{\theta}^{\text{lasso}} = \arg\min_{\theta} \parallel Y - X\theta \parallel_{\ell_2}, \text{ subject to } \parallel \theta \parallel_{\ell_1} \leq t \tag{2.2}$$

In Eq. (2.1), the larger λ is, the greater the amount of shrinkage, and the more parsimonious the selected model will be. More precisely, λ is an upper bound on the correlation between regressors excluded from the model and the regression residual. Interpreting t of Eq. (2.2) is also possible by considering the specific value $t_0 = || \hat{\theta}^{rss} ||_{\ell_1}$. Then, setting t to $t_0/2$ roughly shrinks active coefficients in the regression by 50 % (Hastie et al. 2009).

Equation (2.1) was solved using the Least Angle Regression (LAR) algorithm implemented in the R `glmnet` package (Friedman et al. 2010). The choice of the penalty λ is presented later.

2.2.3.2 The Dantzig Selector

The Dantzig selector (Candès and Tao 2007) is a related penalized linear regression method based on ℓ_1 norm penalization of the parameters subject to a constraint bound on the maximum absolute correlation between the residuals and regressors (we use ${}^\top X$ to denote the transpose of X):

$$\hat{\theta}^{\text{dantzig}} = \arg\min_{\theta} || \theta ||_{\ell_1}, || {}^\top X(Y - X\theta) ||_{\ell_\infty} \leq \delta, \tag{2.3}$$

where δ is the actual bound on the correlation between the residual and each regressor. With no bound, the Dantzig selector sets all coefficients to zero which minimizes the ℓ_1 norm of the parameters. When the bound tends to 0, the Dantzig selector imposes a null correlation between the residual and the regressors. This condition is satisfied by the RSS estimate, as it is equivalent to enforcing a null derivative of the RSS (Hastie et al. 2009). Equation (2.3) can be written in its dual form, as an analog of Eq. (2.2) for the Lasso:

$$\hat{\theta}^{\text{dantzig}} = \arg\min_{\theta} || {}^\top X(Y - X\theta) ||_{\ell_\infty}, || \theta ||_{\ell_1} \leq t \tag{2.4}$$

In Vignes et al. (2011), we used the reduction of the Dantzig selector to linear programming and an open source linear programming solver (glpk) for resolution. Because of the increased computational burden generated by bootstrapping, we decided to instead use a dedicated homotopy Dantzig algorithm (Asif and Romberg 2010) and its companion Matlab package, which was run using Octave. The choice of the value for the parameter δ is difficult and is described in the following.

2.2.3.3 Confidence Scores with Penalized Linear Regressions and Bootstrap

We want to provide confidence scores on the prediction of every oriented edge $j \to g$ capturing the causal influence of gene j on gene g. For a fixed value of the penalization and for a given bootstrap sample, two distinct cases can be identified. If α_{gj} estimation is nonzero, marker j is assumed to have a direct effect on the expression of gene g. The converse is impossible since expression levels cannot affect polymorphism. The interpretation of a nonzero β_{gj} (or β_{jg}) is slightly different: this indicates that a relationship exists between the expressions of genes j and g, but the causal orientation is unknown: either j influences g or g influences j.

Choosing the 'right' level of penalization in the Lasso regression or in the Dantzig selector is a difficult model selection problem. To avoid this problem, we follow the idea of Vignes et al. (2011), described here for the Lasso regression. Instead of choosing a fixed value for the penalty term λ, we explore an evenly spaced grid of possible penalty values from a starting value 0 (no penalization) to a maximum value $\lambda_{\max}$: the infimum of the set of all λ that prevents a single regressor to be included in any of the regressions. A total of $q = 10$ different penalty values we used from $\frac{\lambda_{\max}}{q}$ (low penalty level) to $\lambda_{\max}$ (maximal penalty level). A similar mechanism, with the same number q of penalty values, is used for the Dantzig selector. The fraction of times, over all penalizations, that a regressor is introduced with a nonzero parameter estimate was then used as a confidence score.

In our case, bootstrapping offers a second dimension that can be exploited to evaluate a confidence score. Besides the first dimension defined by the grid of q evenly spaced values of penalizations, a second dimension is available through the set of N_{boot} different bootstrap samples. We denote by $\#(\alpha_{gj})$ (resp. $\#(\beta_{gj})$ the total number of regressions along these two dimensions where $\alpha_{gj} \neq 0$ (resp. $\beta_{gj} \neq 0$). The marker-based confidence score w^m_{kl} of the oriented edge $k \to l$ is then defined as the frequency:

$$w^m_{kl} = \frac{\#(\alpha_{lk})}{q\,N_{\text{boot}}}$$

The computation of the expression-based confidence score w^e_{kl} of the oriented edge $k \to l$ has to take into account the fact that it can be derived from any (or both) of β_{kl} and β_{lk} to be nonzero and also that this information is uncertain on the two possible edge orientations. This leads to:

$$w^e_{kl} = \frac{\#(\beta_{kl}) + \#(\beta_{lk})}{4\,q\,N_{\text{boot}}}$$

Bach (2008) studied the asymptotic ($n \to \infty$) properties of the Lasso variable selection for some penalization decays. In specific settings, the Lasso tends to select correct variables with probability 1 and irrelevant variables with a probability which is strictly between 0 and 1. Hence Bach (2008) proposed to use bootstrapping to assess the probability of selecting a variable at a given penalty level, keeping only those variables that are always selected.

Our strategy is quite different from the strategy proposed in Bach (2008), and it may ultimately include false positive regressors in the model (especially for low confidence scores). But the properties of the analyzed data—including their high dimensionality—and the predicted object, with edge confidence scores, are different from those considered in Bach (2008).

We tested the Lasso method with values of N_{boot} equal to 100 and 200. Given the very limited impact of this choice on the results, we ultimately decided to use the computationally favorable solution of $N_{\text{boot}} = 100$.

2.2.4 Random Forests

In the previous section, we reduced the GRN learning procedure to a gene-by-gene linear regression problem. However, nonlinear regression methods can also be considered, assuming that the expression level of E_g is a function of the remaining expression levels E^{-g} and of allelic states M:

$$E_g = f_g(\mathbf{M}, \mathbf{E}^{-g})$$

The use of random forest for GRN reconstruction from expression data alone has been originally proposed in GENIE3 (Huynh-Thu et al. 2010). Indeed, one way to solve such a nonlinear regression problem between a response variable Y and regressors X is to try to recursively split the observed data with binary tests based each on a single regressor variable, trying to reduce as much as possible the variance of the response variable in the resulting subsets of samples. Each test is a node in a binary tree and typically compares the input variable value with a threshold which is determined during the tree growing. Ultimately, the leaves of the tree give the predicted numerical value for the response variable.

A random forest (Breiman 2001) is a collection of such trees grown partially at random, using two sources of randomness:

- Each tree is grown using a random bootstrapped sample of the data.
- The variable used at each split node is selected only from a random subset of all variables (typically of a fixed size K).

The random forest predicted response for a sample is the mean of all the regressions predicted by each tree. Besides the possible nonlinearity of the response, a specific strength of random forests lies in the fact that they can use the internal bootstrapping to estimate the importance of any regressor. After shuffling the values of the regressor considered in the samples that have not been used in each bootstrapped sub-sample, it is possible to compute the resulting increase in the variance of the regression error compared to non-permuted samples. This provides an evaluation of the regressor importance.

In the context of GRN learning for any gene $g \in \{1, \dots, p\}$, the random forest method provides us with importance factors f_{ig}^m with $i \in \{1, \dots, p\}$ and f_{jg}^e with $j \in \{1, \dots, p\}, j \neq g$, that respectively give the importance of allelic state M_i and of expression level E_j to predict E_g. These weights can then be normalized for each gene (Huynh-Thu et al. 2010). For each $g \in \{1, \dots, p\}$ independently, we normalized the f_{ig}^m and f_{jg}^e by their estimated standard deviation. All these importance factors can then be sorted producing global ranks r_{ig}^m and r_{ig}^e. The marker-based "confidence scores" for oriented edge $k \to l$ are then defined as:

$$w_{kl}^m = 1 - \frac{r_{kl}^m - 1}{N},$$

where N is the largest overall rank. A similar definition is used for the w_{kl}^e.

The computation was performed using the `randomForest` R package (Liaw and Wiener 2002). The number of trees was set to 1,000 and other parameters (defining individual tree depth or the number of variable to draw at each split for example) were kept at their default value.

2.2.5 Bayesian Networks

A Bayesian network is a directed acyclic graphical (DAG) model that captures the joint distribution probability over a set of variables by a factorization in local conditional probabilities linking one random variable with its "parents." The fact that a Bayesian network naturally defines a directed graph makes this formalism highly suitable for learning directed GRNs and unsurprisingly, this mathematical model has already been used to predict GRN in the context of pure expression data analysis in the seminal paper (Friedman et al. 2000).

More formally, a Bayesian network denoted by $B = (\mathscr{G}, P_{\mathscr{G}})$ is defined by a DAG $\mathscr{G} = (V, A)$ with vertices representing random discrete variables $V = \{V_1, \ldots, V_m\}$, linked by a set of directed edges A, and a set of conditional probability distributions $P_{\mathscr{G}} = \{P_1, \ldots, P_m\}$. The variables involved in each conditional probability table P_i are defined by the DAG: $P_i = \mathbb{P}(V_i|Pa(V_i))$, where $Pa(V_i) = \{V_j \in V \mid (V_j, V_i) \in A\}$ is the set of parental nodes of V_i in $\mathscr{G}$.

The DAG of a Bayesian network B implicitly captures a set of conditional independencies between variables and represents a joint probability distribution on V defined as:

$$\mathbb{P}(V) = \prod_{i=1}^{m} \mathbb{P}(V_i|Pa(V_i)) \tag{2.5}$$

To model the available data, as in previous approaches, we used one variable E_i to represent the expression level of gene i and one variable M_i to represent the associated allelic state (for all $i \in \{1, \ldots, p\}$). All variables are discrete (see below for expression level discretization scheme) , allowing to capture nonlinear relationships between variables. If a given DAG structure G is assumed, maximum likelihood estimates of the parameters defining the conditional probability tables can be computed by simple counting. The GRN learning process then reduces to the problem of learning a DAG structure among these variables that maximizes $\mathbb{P}(\mathscr{G}|D) \propto \mathbb{P}(D|\mathscr{G})\mathbb{P}(\mathscr{G})$, where D represents the observed data.

Under specific assumptions, the marginal loglikelihood $\log(\mathbb{P}(D|\mathscr{G}))$ can be expressed as a decomposable scoring function. We used the BDeu score (Heckerman et al. 1995).

2.2.5.1 GRN Structure Learning with Bootstrapped Greedy Search

Learning Bayesian networks is an NP-hard problem with a super-exponential search space of potential DAG structures (Chickering et al. 2004). Even a greedy search heuristic method can be very time-consuming when the number of variables p is large. In order to get reasonable computation times, we selected for each gene a set of candidate parents if the local BDeu score increases when each candidate parent is added separately in comparison to the empty graph. Furthermore, we kept only one marker having the best BDeu score increase in a sliding window of 10 markers along the chromosomes. We also limited the maximum number of parents per gene to 5. In addition, we took into account biological knowledge of the observed data to reduce the search space.

First, genetic linkage between close markers induces strong correlations between their allelic states. Learning the corresponding edges between marker variables M_i is useless for reconstructing the GRN and just makes the structure search more complex. We therefore forbade such edges. We also forbade edges from genes to markers, which have no biological meaning.

Furthermore, a preliminary analysis of variance was used to predict *cis*-regulatory markers: detected positive markers (Bonferroni corrected p-value < 0.1) were those giving the most significant signal in a seven marker-width window, centered on the gene, to avoid false marker influence due to genetic linkage. We used this *cis*-effect information to constrain the structure: since each *cis*-marker M_i had an effect on its associated gene activity E_i only, we constrained our model to use an $M_i \rightarrow E_i$ edge. In the opposite case, when the marker M_i was detected as not being a *cis*-regulatory marker, we only forbade the $M_i \not\rightarrow E_i$ edge.

The structure and parameters of the underlying graph can then be estimated using the BDeu score-based structure learning algorithm described in Vandel et al. (2012), using the same adaptive discretization policy into 2–4 states as in Vignes et al. (2011). We selected the DAG with the best BDeu score among three restarts of the Stochastic Greedy Search algorithm which exploits extended local move operators (SGS^3, Vandel et al. 2012). We used BDeu *equivalent sample size* parameter $\alpha = 1$. Each learnt DAG structure produces a set of oriented edges which therefore translate directly into the predicted GRN. For each edge $M_i \rightarrow E_j$ or $E_i \rightarrow E_j$ in the learnt structure, we predict the existence of a directed edge $i \rightarrow j$ in the GRN. Notice that thanks to this mapping procedure, an initial acyclic directed graph may ultimately lead to the prediction of a cyclic GRN.

This procedure was improved by bootstrapping, allowing to produce a set of directed edges with confidence scores set to the frequency of the corresponding edge over all bootstrap samples ($N_{\text{boot}} = 100$). Again, this may lead to cycles in the predicted GRN. Because of the important computational burden generated by bootstrapping, the Bayesian network approach could, however, not be applied to the largest datasets with $p = 5{,}000$ genes.

2.2.6 Postprocessing

Each of the methods produces two weighted oriented graphs, one that connects marker variables M_j to expression variables E_g (where edge $j \rightarrow g$ is weighted by w^m_{jg}) and another connecting expression variables E_j to other expression variables E_g (where edge $j \rightarrow g$ is weighted by w^e_{jg}). Ultimately,a single ranked list of edges is wanted, representing one gene regulation network.

In order to combine the two informations, we simply ranked all edges $k \rightarrow l$ by the sum of their weights $w^m_{kl} + w^e_{kl}$. This ranking is denoted in the rest of the paper as the "genes-and-markers" ranking.

To further evaluate the influence of the post-processing on the ultimate results and the specific contribution of the causal marker-expression information, we also tested two additional post-processings. The first variant focuses on the genetic information and ranks each edge $k \rightarrow l$ using the w^m_{kl} weight only. Whenever marker j influences the expression of gene g, the strong correlations between the allelic states of j and other close markers on the genome (caused by genetic linkage) may easily lead to spurious predictions involving neighboring markers. We therefore scanned all markers along the genome and removed all edges that were locally dominated by other edges with the same target. More precisely, for all genes g, we removed any marker i such that there exists another marker j within a window of five markers containing i with $w^m_{jg} > w^m_{ig}$. The resulting ranked list of edges is called the "filtered markers" ranking in the following sections.

To evaluate the importance of this filtering process, we also included this marker filtering process in the initial "genes-and-markers" ranking method. In this post-processing method, the w^e and the w^m weights are combined by addition, as in the initial "genes-and-markers" post-processing, but only weights w^m coming from the list of "filtered-markers" edges are used. This third post-processing is naturally termed "genes-and-filtered-markers."

2.3 Results

2.3.1 General Analysis of the Predicted Networks

The above methods have been applied to the 72 datasets, including cases with 100, 1,000, and 5,000 genes. In this section we report results on the 1,000 gene datasets. The 100-gene networks were essentially provided as test datasets. The general trends in the 5,000 gene situation is that it is similar to the 1,000 gene situation (except for the fact that the Bayesian network approach could not be applied, our current implementation being limited to 4 GB corresponding to less than 3,000 variables, providing less opportunity for comparison) with slightly degraded overall performances.

We first present in Table 2.1 performances obtained by individual approaches as the area under the precision versus recall curve (AUPR). We remind the reader that

Table 2.1 Area (in percentage) under the precision versus recall curve (AUPR) for predicted networks with 1,000 genes by, respectively, the Lasso penalized regression, the Dantzig selector, random forests (RF), and Bayesian networks (BN)

Network/configuration/ data-set	AUPR with edge orientations				AUPR without edge orientations			
	Methods				Methods			
	Lasso	Dantzig	RF	BN	Lasso	Dantzig	RF	BN
Net4-Conf1-DS25-300SH	11.65	12.02	9.63	**14.20**	15.76	**16.41**	11.06	15.90
Net4-Conf2-DS26-900SH	15.88	15.66	17.95	**18.30**	**21.97**	21.68	20.05	20.08
Net4-Conf3-DS27-300SL	11.20	11.35	3.88	**11.83**	16.64	**17.18**	5.29	15.01
Net4-Conf4-DS28-900SL	21.49	21.78	9.64	**27.28**	32.46	**33.30**	11.31	32.95
Net4-Conf5-DS29-300DH	4.89	5.02	**7.31**	7.13	6.97	7.29	8.41	**8.60**
Net4-Conf6-DS30-900DH	9.68	10.05	13.82	**20.15**	13.81	14.53	15.60	**22.23**
Net4-Conf7-DS31-300DL	8.60	9.57	3.09	**13.18**	13.07	14.95	4.38	**16.59**
Net4-Conf8-DS32-900DL	16.20	17.43	7.39	**23.24**	24.20	26.71	9.12	**28.76**
Net5-Conf1-DS33-300SH	16.05	15.71	16.16	**16.96**	**21.52**	21.27	17.81	18.89
Net5-Conf2-DS34-900SH	22.17	21.71	23.96	**30.46**	31.08	30.64	26.28	**32.25**
Net5-Conf3-DS35-300SL	14.55	**14.61**	5.56	13.28	21.69	**22.10**	7.42	16.89
Net5-Conf4-DS36-900SL	24.57	24.70	13.53	**25.56**	37.38	**37.85**	15.86	31.37
Net5-Conf5-DS37-300DH	6.66	6.74	**9.04**	8.71	9.34	9.63	**10.58**	10.27
Net5-Conf6-DS38-900DH	12.80	12.67	21.76	**23.74**	17.55	17.76	23.73	**25.66**
Net5-Conf7-DS39-300DL	10.71	11.16	3.60	**15.36**	17.10	18.19	5.20	**18.71**
Net5-Conf8-DS40-900DL	17.42	17.92	11.04	**25.57**	26.33	27.75	12.86	**30.71**
Net6-Conf1-DS41-300SH	13.07	12.83	13.34	**15.75**	**17.90**	17.64	15.05	17.72
Net6-Conf2-DS42-900SH	17.54	17.59	23.63	**24.13**	24.81	24.80	25.56	**26.14**
Net6-Conf3-DS43-300SL	12.62	12.72	4.32	**13.40**	19.00	**19.38**	5.64	17.02
Net6-Conf4-DS44-900SL	20.72	**21.07**	10.67	20.14	32.06	**32.72**	12.69	26.12
Net6-Conf5-DS45-300DH	5.43	5.51	**7.41**	5.70	7.79	7.98	**8.83**	6.98
Net6-Conf6-DS46-900DH	8.55	8.43	**15.90**	12.34	11.91	11.95	**17.67**	14.13
Net6-Conf7-DS47-300DL	8.70	9.23	2.57	**10.07**	13.69	**14.84**	3.98	13.42
Net6-Conf8-DS48-900DL	14.68	15.33	7.82	**16.11**	22.86	**24.41**	10.06	21.36

Each network name ends by a short string which defines the sample size ($n = 300$ or 900), the gene density (D/S for dense or sparse), and the simulated gene expression heritability (H/L for high or low)

the recall is the ratio of correctly predicted edges among all edges to predict. The precision is the ratio of correctly predicted edges among all predicted edges. Since the predicted list of edges is ranked, edges are successively introduced with decreasing confidence scores, and precision and recall levels are computed at each step, defining a curve in the precision–recall space. The AUPR score is a compromise of the global performance of the method. It is a usual criterion in the Machine Learning community. Its values range from 1 (perfect recovery of all true edges with no error) to 0 (all predicted edges are incorrect). Note that a simple random guessing of edges should produce an AUPR of $\frac{\sharp\{\text{true edges}\}}{\sharp\{\text{possible edges}\}}$ often close to 0 if the network is sparse (with a number of edges much lower than the potential number of directed edges of $n(n-1)$). We report AUPR scores with and without taking edge orientations into account.

Sensitivity to Simulated Parameters

As described in Chap. 1, the different datasets were generated by varying different parameters: the underlying directed network, the number of simulated RILs (samples), the spacing between markers (and therefore gene density given the one gene/one marker assumption), and the gene expression heritability. The first observation that can be done on these datasets is the sensitivity of all methods to sample size that was expected. For 300 individuals (odd configuration numbers), the AUPR is usually between 5 and 15 % roughly. These results are weaker than the results presented in Vignes et al. (2011), using a set of different simulation parameters. With 900 individuals, the performance of all methods almost doubles, but still remains relatively poor.

Considering the inter-marker distance (low distance for configurations 5–8), all methods were sensitive to gene density. The higher the gene density, the lower the AUPR for the linear regressions and the random forest-based method. Indeed, with a high density marker map, the allelic states of two neighboring markers are correlated. It therefore becomes more difficult to precisely predict which marker i regulates a gene g: all neighboring markers offer essentially the same predictive information. Compared to these methods, the Bayesian network approach benefits from the dedicated marker pre-processing aimed at identifying *cis*-regulatory markers. When such a *cis* marker is detected, the corresponding edge is forced into the Bayesian network structure, avoiding increasingly likely mistakes in high-density configurations.

This marker distance effect became less important in configurations with a great number of individuals. This is due to the increased power and reliability of the regulatory marker localization and the expression data itself that becomes more informative.

From a more usual linkage analysis point of view, we must point out that the evaluation criteria used here is a very hard one. Imagine M_i is the regulatory marker that should have been identified as influencing gene g. If M_{i+1} (or any other neighbor marker/gene) is predicted instead, then the difference in terms of the chromosomal region that influences E_g is negligible and becomes increasingly small with an increasing gene density. Despite this increasingly small error, our edge detection criteria is purely Boolean and counts any neighbor prediction as a totally bad prediction. This probably advocates for new evaluation criteria beyond a pure $0/1$ edge detection event.

Another possibility that could explain this sensitivity to inter-marker distances lies in the quality of our confidence score. Compared to Vignes et al. (2011), the explored grid of penalizations was reduced to only $q = 10$ different values (instead of 20), offering a coarser image of the frequency of inclusion of a variable. This was required, from a computational point of view, because of the additional computational burden generated by bootstrapping. Another possibility would be that the bootstraping itself facilitates the selection of neighboring markers, because of the generated sampling noise. Our later evaluation of the bootstrapping influence (in one network), however, tends to show this is probably not the case.

Finally, all methods seem to be sensitive to the simulated gene expression heritability. In many cases, and quite unexpectedly at first, a weak heritability (high biological variance) seems to be favorable, especially for linear regression approaches. A likely explanation lies in the relationships that one can observe between the expression levels of two genes g_1 and g_2 that have a common parent regulator. In high heritability situations, a strong nonlinear dependency between the two genes was visible. In low heritability situations, the nonlinearity of the relationship seemed to fade away, which was favorable to linear regression methods. A possible way to enhance the behavior of linear regression in these conditions would be to apply a power transform such as the Box–Cox transform (Box and Cox 1964), a useful data pre-processing technique used to stabilize variance and make the data more normal distribution-like.

Comparison of Different Methods

The Lasso regression and the Dantzig selector had, as expected, very similar performances (both are linear regression approaches, using the same underlying model). Although, Dantzig seemed to offer better performances than Lasso in a majority of cases, the difference was usually negligible. This is different from our previous comparison in Vignes et al. (2011), where the Dantzig selector seemed to offer better performance than the Lasso regression (the Dantzig selector approach finished second of the DREAM5 challenge, and was outperformed only by a consensus meta-analysis that exploited its predictions). More tests would be needed to check whether this could be caused by the shift in the underlying optimisation method. Vignes et al. (2011) used an exact simplex-based linear programming solver, while we used a more efficient homotopy-based method (Asif and Romberg 2010) to cope with the additional computational burden generated by the bootstrapping process.

Surprisingly, our random forest approach often gave the worst performance (except on configurations 2 and 6, with high heritability). A likely explanation for the limited performance of our random forest approach lies in the fact that we handled expression data E_i and allelic states M_j together, as possible splitting variables in the same trees. However, these two variables are very different. Expression data is a continuous variable, which offers a lot of freedom on possible splitting decisions. Allelic states are Boolean variables, and therefore offer no freedom in the possible splitting decision. The criteria used to grow forests and decide which variable is taken next being its ability to reduce the variance in the separated datasets, it is likely that allelic state variables tend to be inserted lately compared to expression data variables. This tendency is confirmed by Strobl et al. (2007) when random forests are used for classification purpose: the algorithm tends to be biased and to be more likely to select variables with a higher number of levels than variables with few levels like binary variables. This may lower the performance of the approach a lot compared to others. A better approach would probably be to handle expression data variables and allelic state variables in two independent random forest constructions to later mix the associated confidence scores in a single confidence score (Geurts and Huynh-Thu 2012) instead of mixing the simultaneous edge ranks as we did (see also

Chap. 7 of this book). The better performances (compared to other methods) on the configurations 2 and 6 (high heritability) can also be explained with the previously mentioned highly nonlinear relationships that appear between co-regulated genes in this case. The ability of random forests to capture nonlinear relationships may explain its relative success on these configurations.

Overall, Bayesian networks offer the best performance in most cases, comparable to those obtained in Vignes et al. (2011). We observe that each method has an advantage in a specific area of the parameter space, showing their potential complementarity.

Directed or Undirected Edges

In our linear regression methods, the current post-processing only partially allows us to orientate edges, thanks to marker data: it is known that genetic content influences observed phenotypes, including gene expression levels, and not the converse. Learnt relationships between variables that account for gene expression levels are symetrical, hence the predicted direct links between these variables are given a disadvantage. The other way to see this is that edges from a marker variable to a gene expression are favored even if they are slightly less supported by the data according to a modeling criterion. This may result in either spurious marker to gene expression arcs being assigned relative scores higher than they should or significant relationships between gene expression levels being moved back in the list of predicted edges. For this reason, we also evaluate the performances considering undirected edges. This analyzes the ability to predict noncausal relationships between genes. The corresponding AUPR scores are given in the four rightmost columns of Table 2.1.

The limited ability of penalized linear regression methods to infer causality (which is only done when a polymorphism is detected as a possible regressor of the expression level) leads to visible improvements in their predictive capabilities. The Lasso regression and the Dantzig selector are performing best in only two cases if directed edges are considered and in 12 cases over 24 in the undirected edge case. In the best cases, the Dantzig selector can offer an AUPR of almost 40 %, with performances that more frequently exceed those of Bayesian networks.

Overall, both the Bayesian Network approach and the random forest approach seems to benefit less from this relaxation in the evaluation criteria. This means that when they predict a directed edge, in most cases, this edge may be a correct directed edge or else, it is often a false edge (even ignoring orientation).

Table 2.2 Effect of different post-processings on the AUPR scores (in percentage) for learning Net5-Conf2-DS34 ($n = 900$, sparse marker map and high gene expression heritability) with the Lasso regression, the Dantzig selector and our random forest method (RF)

Post-processing	Methods		
	Lasso	Dantzig	RF
Genes-and-markers	22.17	21.71	23.96
Genes-and-filtered-markers	23.38	23.21	**25.66**
Filtered-markers	**28.10**	**26.30**	20.05

2.3.2 *Focus on a 1,000 Gene Network with $n = 900$, High Gene Expression Heritability, and 5 cM Between Consecutive Markers (Net5-Conf2-DS34-900SH)*

Implementing a practical learning scheme for GRN in System Biology cannot be just characterized by the underlying mathematical method used to perform the inference itself. The modeling, the pre- and post-processing of the data, possible bootstrapping (and number of bootstrap samples), the criteria used for evaluation may all have non-negligible impacts on the final results.

2.3.2.1 Post-processing

To analyze the importance of the post-processing used on the obtained results, we compared two other post-processing methods on one network. We used dataset 34. With a large sample, a sparse gene density and high heritability, it defines a playground where most methods (including our random forest approach) obtain comparable performances for the directed edges prediction. The Bayesian network approach is excluded from this comparison since it uses a specific pre-processing that could hinder the effect of the different post-processings.

Table 2.2 gives the AUPR obtained using the previous "genes-and-markers" post-processing (which combines expression data-based scores w^e and allelic states-based scores w^m by addition) with two extra post-processing (described in Sect. 2.2.6). The "filtered-markers" post-processing focuses on marker information by using only w^m scores. It also tries to weaken the influence of high correlations between close markers by keeping only markers with local undominated w^m score. The "genes-and-filtered-markers" post-processing combines this filtering process in the original "gene-and-markers" post-processing.

On this dataset, one can check that a change in the post-processing may change the ranking of the different methods. On dataset 34, retaining only allelic state-based scores gave much better results for linear regression methods but was rather disappointing for random forests. As we mentioned previously, the Boolean allelic state variables are probably introduced lately in the regression trees, giving priority to expression levels. This may explain the important effect of keeping only allelic state-based scores on random forests. The mixed "genes-and-filtered-markers" post-processing gave more even performances.

All three post-processing were tested on all configurations of the whole datasets. The "filtered-markers" post-processing gave the overall best result on dataset 34, but it gave extremely poor results on most other configurations (except for configurations 1 and 2, all AUPR being below 11 %). The "genes-and-filtered-markers" post-processing often offered performances that were comparable to our initial "genes-and-marker" post-processing, but was more often dominated by it than the converse.

Clearly, because of the dual nature of genetical genomics data that captures different effects through different types of variables, there are plenty of possible choices for combining the two types of information. A sensible conclusion at this point is that the optimal post-processing depends on features of the data that need to be carefully checked before proposing analysis results.

2.3.2.2 The AUPR Curve

To be consistent with the previous DREAM5 experiment, we used the AUPR criteria to compare the performances of the different approaches. In this section, we want to show that AUPR is a very high-level criterion. It summarizes the performance of every method as a unique number, but hides different behaviors.

In Fig. 2.1, we give the complete precision versus recall curve for all methods on dataset 34 using two different post-processings ("genes-and-markers" and the best post-processing from Table 2.2). Some methods (such as the Dantzig selector in the "filtered-markers" post-processing) tend to have a very high precision initially. This means that almost all of the first few hundred predicted edges are correct. On this same dataset, the Bayesian network learning presented a similar AUPR but a very different behavior. The initial precision decreased more rapidly. This means that false positive edges existed in the beginning of the list of ranked edges, making the output harder to use. Again, this comparison between the Dantzig selector and the Bayesian network learning is valid for this dataset and this post-processing but conclusion may differ on another dataset and with different post-processings.

2.3.2.3 Effect of the Chosen Number of Bootstraps

The use of bootstrapping is quite costly in terms of computation time: all cpu times are immediately multiplied by our $N_{\mathrm{boot}} = 100$ bootstraps. We checked whether this additional work is of interest first beyond theoretical grounds and whether it is sufficient. We therefore compared the results obtained using bootstrapping with an increasing number of bootstraps. The results obtained on dataset 34 are presented in Fig. 2.2 using the Lasso regression approach and two different post-processings. These curves give the observed recall at different precision levels. Precision levels

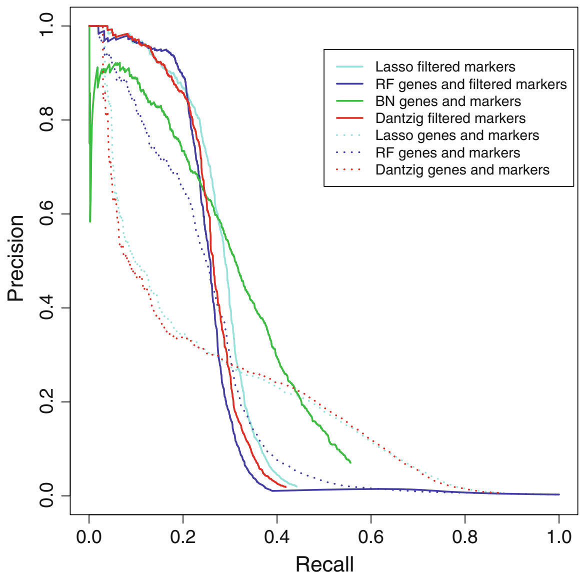

Fig. 2.1 PR *curve* for the different methods and different post-processings on dataset Net5-Conf2-DS34-900SH

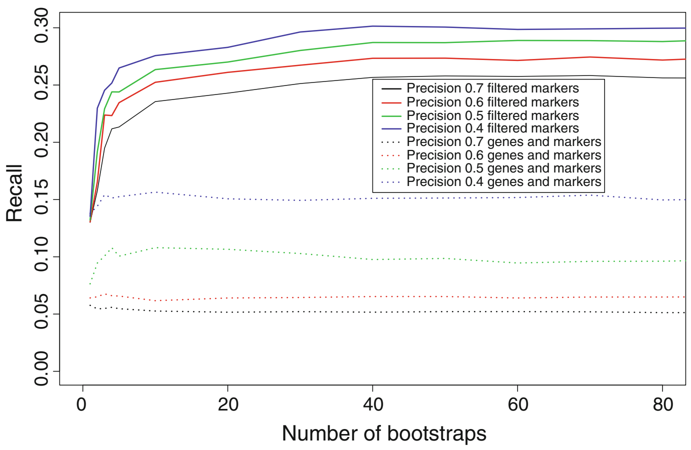

Fig. 2.2 Effect of the number of bootstrapped samples on dataset Net5-Conf2-DS34-900SH for different levels of precision with the Lasso regression method using the "genes-and-markers" and the "genes-and-filtered-markers" post-processings

range from a nearly acceptable degree of precision of 0.4 to a good degree of precision of 0.7. One can see that bootstrapping enhances performance, but the importance of the improvement varies a lot depending on the post-processing used. These curves

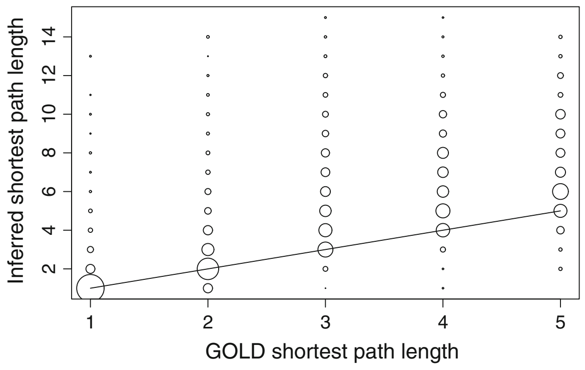

Fig. 2.3 Normalized frequency for shortest path lengths in the predicted network as a function of shortest path lengths in the Gold Standard network. Disc areas are proportional to normalized frequencies. Dataset 34 was analyzed with the Lasso regression method using the "genes-and-filtered-markers" post-processing. The predicted network was built with the 1,000 first predicted edges. Raw total column frequencies are 540, 2,283, 2,407, 1,098, and 82. For clarity, we do not represent shortest paths that are not recovered (virtually of length ∞)

also confirm that the number of bootstrap sub-samples we performed is sufficient, an asymptotic behavior being reached before 50 sub-samples in most cases.

2.3.2.4 Shortest Path Lengths

Another way to gain knowledge about the topology of the predicted network (without edge orientations) is to compare its shortest paths to the shortest paths of the true (Gold Standard) network. In a graph, the shortest path between two nodes is defined as the path of minimum length connecting these two nodes. For example, shortest paths of length 1 are direct edges between two nodes. If the network was perfectly recovered, the length of shortest paths for any pair of nodes in the predicted graph would perfectly match the length of the corresponding shortest paths in the true network. We depict in Fig. 2.3 the normalized distribution of shortest path lengths in one of the predicted network as a function of the shortest path lengths in the corresponding Gold Standard network. Note that shortest paths which exist in the true network but whose extremities are not connected in the predicted network are not accounted for here. The rationale here is to check whether the distribution of distances between genes of the two compared networks are close. If this is the case, the distribution should concentrate around the $y = x$ axis.

For example, we found 2,341 shortest paths of length one (i.e., edges) that were not connected at all in the predicted network. It means that among the 2,882 true edges to recover, 439 (implying a recall of 15.2 %) were correctly identified by the predicted network while 102 were found with at least another intermediate gene in the path between them, and 2,341 pair of genes were not connected in the predicted network. This is due to the fact that the predicted network was not fully connected, and that only 1,000 edges were predicted while the true network contained three times more edges.

Focusing on the first two columns of Fig. 2.3, we see that the predicted graph had a distribution of shortest paths of lengths 1 and 2 that matched with the actual distribution of such edges in the true network. This was not the case anymore for paths of length 3 and over: the predicted network tended to overestimate the length of such shortest paths when it was able to identify them. Again, the fact that we only studied the first 1,000 predicted edges plays an important role in this observation. Longer paths are needed to cope with correct paths of a given length that are composed of true edges only. However, even those few edges and some of them that were not correct (the precision level was here below 0.5), the distributions of shortest path lengths in the predicted network and in the true network were not too different.

2.4 Conclusion

In contrast with our previous experiments in Vignes et al. (2011), which relied on a subset of the same algorithms (the Lasso regression, the Dantzig selector and Bayesian network structure learning) on data generated using the same generator but with simplified settings, the results we obtained here with additional effort (computational cost of the bootstrapping process and integration of a random forest approach) are rather disappointing in the hardest situations, in terms of AUPRs. On these hardest problems, whether because of limited number of individuals, or because of nonlinear relationships induced by high heritability, these low AUPR often hide rapidly decreasing precision, which means that the list of predicted edges quickly becomes hard to exploit to predict gene regulations.

Given that boostrapping seems to be able to provide improved prediction quality, the most likely reason of these results lies in the generated data itself which relies on new parameters and new networks. Clearly, the specific characteristics of these new datasets could probably be accommodated, to some extent, by the same methods, using different modeling or pre/post-processing strategies. In the case of configuration 2, the use of the "filtered-marker" post-processing strategy provided significant improvements in the AUPR of the linear regression-based approaches, leading to respectable performances.

These results point also to the ambitious aim of trying to identify which statistical method would perform best for GRN learning in a genetical genomics context, based on a large scope of methods and on large datasets of simulated data. The overwhelming size of combinations that can be imagined using different methods, models and pre and post-processing procedures to deal with a large amount of simulated data which is itself parameterized by several parameters defines a daunting task. It is quite obvious for us that we have only explored a small fraction of all possible combinations and that much work remains to be done to identify an ideal performer, if it exists. Indeed, our results show that every method, even those that tend to give the worst results, can outperform other methods in specific situations. A combination of methods may therefore be, ultimately, the best approach.

These experiments also show the possible difficulty in the generation of realistic simulated data for genetical genomics. Given the radically different results obtained depending on the change of parameters used for problem generation, it is natural to wonder which of these different settings could be considered as the most realistic model for real genetical genomics data and for which organism. First elements of response to this question may finally open the door to tackle real datasets.

Acknowledgments We are very grateful to the staff of the GenoToul (Toulouse, France) Bioinformatics plateform for the computational support it provided during this work. We would also like to thank our colleagues from CRS4 Bioinformatica for creating the datasets of the present study.

References

Asif MS, Romberg JK (2010) Dynamic updating for ℓ_1 minimization. J Sel Top Sig Process 4(2):421–434

Aten JE, Fuller TF, Lusis AJ, Horvath S (2008) Using genetic markers to orient the edges in quantitative trait networks: the NEO software. BMC Bioinform 2:34

Bach F (2008) Bolasso: model consistent lasso estimation through the bootstrap. In: Cohen WW, McCallum A, Roweis ST (eds) Proceedings of the twenty-fifth international conference on machine learning (ICML), ACM international conference proceeding series, vol 307. Helsinki, Finland, pp 25–32

Bansal M, di Bernardo D (2007) Inference of gene networks from temporal gene expression profiles. IET Syst Biol 1(5):306–312

Box GEP, Cox DR (1964) An analysis of transformations. J Roy Stat Soc Ser B (Methodological), 26(2):211–252

Breiman L (2001) Random forests. Mach Lear 45(1):5–32

Candès E, Tao T (2007) The Dantzig selector: Statistical estimation when *p* is much larger than *n*. Ann Stat 35(6):2313–2351

Chickering D, Heckerman D, Meek C (2004) Large-sample learning of Bayesian networks is NP-hard. J Mach Learn Res 5:1287–1330

Efron B, Tibshirani R (1997) Improvements on cross-validation: The. 632+ bootstrap method. J Am Stat Assoc 92(438):548–560

Efron Bradley (1981) Nonparametric estimates of standard errors - the jackknife, the bootstrap and other methods. Biometrika 68(3):589–599

Friedman J, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models. J Stat Softw 33(1):1–22

Friedman N, Nachman I, Peér D (1999) Learning bayesian network structure from massive datasets: The “sparse candidate” algorithm. In: Proceedings of the 15th conference on uncertainty in artificial intelligence, Stockholm, Sweden, pp 206–215

Friedman N, Linial M, Nachman I, Peer D (2000) Using Bayesian networks to analyse expression data. J Comput Biol 7(3):601–620

Geurts P, Huynh-Thu V-A (2012) Personal communication

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction. Series in Statistics, 2nd edn. Springer, New York

Heckerman D, Geiger D, Chickering DM (1995) Learning Bayesian networks: the combination of knowledge and statistical data. Mach Learn 20(3):197–243

Huynh-Thu VA, Irrthum A, Wehenkel L, Geurts P (2010) Inferring regulatory networks from expression data using tree-based methods. PLoS ONE 5(9):e12776

Jansen RC, Nap NP (2001) Genetical genomics : the added value from segregation. Trends Genet 17(7):388–391

Koller D, Friedman N (2009) Probabilistic graphical models: principles and techniques. MIT press, Cambridge

Lèbre S, Becq J, Devaux F, Stumpf MH, Lelandais G (2010) Statistical inference of the time-varying structure of gene-regulation networks. BMC Systems Biology 4:130

Leclerc RD (2008) Survival of the sparsest: robust gene networks are parsimonious. Mol Syst Biol 4:213

Liaw A, Wiener M (2002) Classification and regression by randomforest. R News 2(3):18–22

Liu B, de la Fuente A, Hoeschele I (2008) Gene network inference via structural equation modeling in genetical genomics experiments. Genetics 178(3):1763–1776

Marbach D, Mattiussi C, Floreano D (2009) Replaying the evolutionary tape: biomimetic reverse engineering of gene networks. Ann New York Acad Sci 1158(1):234–245

Pinna A, Soranzo N, Hoeschele I, de la Fuente A (2011) Simulating systems genetics data with SysGenSIM. Bioinformatics 27(17):2459–2462

Rau A, Jaffrezic F, Fouley J-L, Doerge RW (2010) An empirical Bayesian method for estimating biological networks from temporal microarray data. Stat Appl Genet Mol Biol 9(1):art.9

Strobl C, Boulesteix A-L, Zeileis A, Hothorn T (2007) Bias in random forest variable importance measures: Illustrations, sources and a solution. BMC Bioinform 8:25

Thomas R (1973) Boolean formalization of genetic control circuits. J Theor Biol 42(3):563–585

Tibshirani R (1996) Regression shrinkage and selection via the lasso. J Roy Stat Soc Ser B (Methodological), 58(1):267–288

Vandel J, Mangin B, de Givry S (2012) New local move operators for Bayesian network structure learning. In: Proceedings of PGM-12, Granada, Spain

Vignes M, Vandel J, Allouche D, Ramadan-Alban N, Cierco-Ayrolles C, Schiex T, Mangin B, de Givry S (2011) Gene regulatory network reconstruction using Bayesian networks, the Dantzig selector, the lasso and their meta-analysis. PloS one 6(12):e29165

Yvert G, Brem RB, Whittle J, Akey JM, Foss E, Smith EN, Mackelprang R, Kruglyak L (2003) Trans-acting regulatory variation in saccharomyces cerevisiae and the role of transcription factors. Nat Genet 35(1):57–64

Zhu J, Wiener MC, Zhang C, Fridman A, Minch E, Lum PY, Sachs JR, Schadt EE (2007) Increasing the power to detect causal associations by combining genotypic and expression data in segregating populations. PLoS Comput Biol 3(4):e69