

Recovering gene networks using l_1 and l_0 penalties

Johan J. de Rooi^{1,2,*} Paul H. C. Eilers²

¹Department of Bioinformatics, Erasmus Medical Centre, Rotterdam, the Netherlands

²Department of Biostatistics, Erasmus Medical Centre, Rotterdam, the Netherlands

*Email: j.derooi@erasmusmc.nl

Recently various approaches to recover genetic networks from expression data have been proposed. Often applied methods are Support Vector Machines, Bayesian networks or methods based on information theory. From a statistical point of view it is logical to use the covariance matrix of the genes to build the network. More elegant is to use its inverse (aka the precision matrix), because of its close relation with partial correlations. Although the final model should be sparse, the covariance matrix doesn't contain any zeros. Due to the large number of variables and a limited number of samples, the inverse often cannot be calculated. In order to derive an invertible covariance matrix and reach a sparse model, shrinkage procedures based on the so-called "bridge" penalty have been applied.

The approach we take here is based on 'running lasso regression'; this means that a regression model is fitted on one variable in the model with all others being the predictors. This procedure is repeated for all variables. There is a direct link between this model and the inverse covariance matrix: a non-zero regression coefficient corresponds to a non-zero element of the precision matrix. Probably the most often used penalty is the l_1 penalty, aka the lasso. This penalty is attractive because it is convex and does both shrinkage and variable selection by setting coefficients to zero. However in connection to recovering sparse networks the l_1 often leaves too many edges in the network. In order to further reduce the number of nodes with many relations, we use the l_0 penalty and in this way yield a model that better resembles the very sparse nature of genetic networks. Because of the non-convex nature of the l_0 penalty we adopt a two step strategy. As a first step the l_1 penalty is applied to compute an initial solution. In the second step the l_0 penalty is put to work on the remaining non-zero coefficients in the model. Preliminary results on both real data and small simulations show promising results.