BIOINFORMATICS ORIGINAL PAPER Vol. 25 no. 14 2009, pages 1775–1781
doi:10.1093/bioinformatics/btp322

Gene expression

Gradient lasso for Cox proportional hazards model

Insuk Sohn[1], Jinseog Kim[2], Sin-Ho Jung[1] and Changyi Park[3,*]

[1]Department of Biostatistics & Bioinformatics, Duke University, NC 27705, USA, [2]Department of Statistics and Information Science, Dongguk University, Gyeongju 780-714 and [3]Department of Statistics, University of Seoul, Seoul 130-743, Korea

Received on February 9, 2009; Revised on May 7, 2009; accepted on May 12, 2009

Advance Access publication May 15, 2009

Associate Editor: John Quackenbush

ABSTRACT

Motivation: There has been an increasing interest in expressing a survival phenotype (e.g. time to cancer recurrence or death) or its distribution in terms of a subset of the expression data of a subset of genes. Due to high dimensionality of gene expression data, however, there is a serious problem of collinearity in fitting a prediction model, e.g. Cox's proportional hazards model. To avoid the collinearity problem, several methods based on penalized Cox proportional hazards models have been proposed. However, those methods suffer from severe computational problems, such as slow or even failed convergence, because of high-dimensional matrix inversions required for model fitting. We propose to implement the penalized Cox regression with a lasso penalty via the gradient lasso algorithm that yields faster convergence to the global optimum than do other algorithms. Moreover the gradient lasso algorithm is guaranteed to converge to the optimum under mild regularity conditions. Hence, our gradient lasso algorithm can be a useful tool in developing a prediction model based on high-dimensional covariates including gene expression data.

Results: Results from simulation studies showed that the prediction model by gradient lasso recovers the prognostic genes. Also results from diffuse large B-cell lymphoma datasets and Norway/Stanford breast cancer dataset indicate that our method is very competitive compared with popular existing methods by Park and Hastie and Goeman in its computational time, prediction and selectivity.

Availability: R package `glcoxph` is available at http://datamining.dongguk.ac.kr/R/glcoxph.

Contact: park463@uos.ac.kr

1 INTRODUCTION

DNA microarray is a biotechnology allowing simultaneous monitoring of tens of thousands of gene expression in cells (Brown and Botstein, 1999). It has important applications in pharmaceutical and clinical research, including tumor classification, molecular pathway modeling and functional genomics. The identification of genes related to survival may provide new information on pathogenesis and etiology, and may further aid in the search for new targets for drug design.

There has been an increasing interest in relating gene expression profiles to survival phenotypes such as time to cancer recurrence or death. Recent works include semi-supervised methods (Bair and Tibshirani, 2004), supervised principal components (Bair *et al*., 2006), a Cox regression based on threshold gradient descent (Gui and Li, 2005a), the LARS-Cox (Gui and Li, 2005b), the residual finesse (Segal, 2006) and cancer survival prediction using automatic relevance determination (Kaderali *et al*., 2006).

In particular, we are interested in the Cox regression (Cox, 1972) because of its popularity in the analysis of survival data with censoring. However, due to high dimensionality of gene expression data, there is a serious problem of collinearity in applying the Cox proportional hazards model directly. To avoid the collinearity problem, several methods based on penalized Cox proportional hazards models have been proposed. Fan and Li (2002) adopted the smoothly clipped absolute deviation (SCAD) penalty in the Cox model and Zou (2008) proposed a path-based variable selection method to construct an adaptive sparse shrinkage path of the coefficients. Because the SCAD penalty is non-convex, the optimization with the SCAD penalty has a non-convex problem. Zhang and Lu (2007) suggested an adaptive lasso method with an adaptively weighted penalty on coefficients. Note that the adaptive lasso has an extra tuning parameter and thus the computation is much heavier than usual one step estimations. Park and Hastie (2007) generalized the LARS algorithm to generalized linear models and the Cox proportional hazards model.

The LARS-Cox procedure by Gui and Li (2005b) suffers from severe computational problems in its optimization because it requires inversion of high-dimensional matrices as discussed in Segal (2006) or Goeman (2008). To remedy the computational problem of the LARS-Cox, Segal (2006) proposed a method, called the residual finesse, to speed up the computation by replacing the survival times by the deviance residuals and Goeman (2008) proposed to combine gradient descent with the Newton–Raphson algorithm.

In this article, we apply the gradient lasso algorithm proposed by Kim *et al*. (2008) to a Cox proportional hazards model with a lasso penalty. Unlike the algorithm in Gui and Li (2005b), the gradient lasso algorithm does not require a matrix inversion, so that it is scalable to high-dimensional data and yields faster convergence to the global optimum. Theorem 1 in Kim *et al*. (2008) guarantees the convergence of the gradient lasso estimator to the optimum under mild regularity conditions. Unlike other methods such as Gui and Li (2005b), Segal (2006) and Zou (2008), the gradient lasso yields a solution to the exact penalized partial likelihood, not to an approximate penalized partial likelihood. Hence, our method can be a useful alternative to those methods in building a statistical model

*To whom correspondence should be addressed.

to predict the patient's survival time based on high-dimensional gene expression data.

A recent method proposed by Park and Hastie (2007) provides the entire penalization path for the Cox model in a forward stage-wise manner. Both their method and our gradient lasso act only on active sets. Because Park and Hastie (2007) require matrix inversions only on active sets, the computation is faster and more stable than other methods in the literature. Ma and Huang (2005) adopted the earlier version of the gradient lasso by Kim and Kim (2004). In this article, we adopt the refined version of Kim and Kim (2004) with a deletion step to improve the computational speed and provide more stable solutions than the earlier version. Another recent method by Goeman (2008) is a gradient algorithm with an option of switching to the Newton–Raphson algorithm to avoid slow convergence. Both our method and the algorithm by Goeman (2008) are gradient algorithms. However, our method directly solves the optimization problem of the lasso penalized Cox model in the constrained form, whereas the methods of Park and Hastie (2007) and Goeman (2008) solve the optimization problem equally in an unconstrained form using a Lagrange multiplier.

The rest of the article is organized as follows. In Section 2, we describe the Cox proportional hazards model with lasso penalty and present the gradient lasso algorithm for the model. We briefly review Park and Hastie (2007), too. In Section 3, we compare the methods in Park and Hastie (2007) and Goeman (2008) with the gradient lasso on simulated and real microarray datasets. Finally, we give brief discussions of our method and results.

2 METHODS

2.1 Cox model with lasso penalty

We investigate the relationship between the survival time and the covariates such as gene expression levels and clinical variables. For patient i (=1, …, N), we observe $(t_i, \delta_i, x_{i1}, \ldots, x_{ip})$, where δ_i is the censoring indicator taking 1 if an event is observed and 0 otherwise, t_i denotes the survival time if $\delta_i = 1$ or censoring time otherwise, and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$ is the vector of covariates.

By the Cox's proportional hazards model, the hazard function for patient i is given as

$$\lambda_i(t) = \lambda_0(t)\exp(\boldsymbol{x}_i^T\boldsymbol{\beta}), \tag{1}$$

where $\lambda_0(t)$ is a unknown baseline hazard function and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ is the regression coefficient vector. Then the partial log-likelihood (Cox, 1972) is expressed as

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{N}\delta_i\left\{\boldsymbol{x}_i^T\boldsymbol{\beta} - \log\left(\sum_{j\in R_i}\exp(\boldsymbol{x}_j^T\boldsymbol{\beta})\right)\right\},$$

where R_i is the set of indices of the patients at risk at time t_i-.

Typically for microarray data we have $p \gg N$, so that there exists a serious collinearity problem when applying the partial likelihood estimation to the Cox model directly. Tibshirani (1997) proposed to estimate the parameters in (1) under the L_1 constraint:

$$\hat{\boldsymbol{\beta}} = \arg\max l(\boldsymbol{\beta}),\ \text{subject to}\ \|\boldsymbol{\beta}\|_1 \le s, \tag{2}$$

where $\|\cdot\|_1$ denotes the L_1 norm and s is a positive number. The optimization problem (2) is good for dimension reduction of covariates, but is computationally difficult because the L_1 objective function is not differentiable (Fan and Li, 2002).

The equivalent Lagrange multiplier version of (2) is

$$\hat{\boldsymbol{\beta}} = \arg\min\{-l(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1\} \tag{3}$$

for $\lambda > 0$. Equation (3) can be interpreted as the posterior mode of $\boldsymbol{\beta}$ for the double exponential prior with parameter λ. The tuning parameters s and λ can be determined, for example, by cross-validated partial likelihood (CVPL) as in Gui and Li (2005b) and Segal (2006).

Tibshirani (1997) proposed an iterative procedure to solve (2). Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ denote the $p \times N$ gene expression matrix, $\boldsymbol{\eta} = \boldsymbol{\beta}^T\boldsymbol{X}$, $\boldsymbol{\mu} = -\frac{\partial l}{\partial \boldsymbol{\eta}}$, $\boldsymbol{A} = -\frac{\partial^2 l}{\partial\eta\partial\eta^T}$, and $\boldsymbol{z} = \boldsymbol{\eta} + \boldsymbol{A}^-\boldsymbol{\mu}$, where $\boldsymbol{A}^-$ is a generalized inverse of $\boldsymbol{A}$. By the Taylor expansion of $l(\boldsymbol{\beta})$, the partial log-likelihood is approximated by

$$(\boldsymbol{z} - \boldsymbol{\eta})^T\boldsymbol{A}(\boldsymbol{z} - \boldsymbol{\eta}). \tag{4}$$

Tibshirani (1997) suggested to replace $\boldsymbol{A}$ with a diagonal matrix $\boldsymbol{D}$ having the same diagonal elements as $\boldsymbol{A}$ and solve (4) iteratively using a quadratic programming. See Tibshirani (1997) for more details on the iterative procedure. Since the quadratic programming cannot be applied directly to the cases with $p \gg N$, Gui and Li (2005b) applied the Choleski decomposition: $\boldsymbol{T} = \boldsymbol{A}^{1/2}$ and $\boldsymbol{T}^T\boldsymbol{T} = \boldsymbol{A}$. Note that, with the restriction $\text{rank}(\boldsymbol{A}) = n - 1$ in place, the quadratic form (4) is invariant to the choice of the generalized inverse. The LARS-Cox procedure can be solved using the LARS procedure in Efron *et al*. (2004). See Gui and Li (2005b) for more details.

As noted earlier, these algorithms can be very computationally intensive and sometimes may fail to converge to the optimum because they involve matrix inversions. Other methods in the literature such as Segal (2006) and Zou (2008) speed up the computation by solving approximate objective functions. The gradient lasso algorithm presented below solves the optimization problem (2) directly and does not resort to a matrix inversion. Thus, the gradient lasso can be a useful alternative. Another recent method of Goeman (2008) is a gradient algorithm solving (3). While the gradient lasso updates a single coordinate at a time, the method in Goeman (2008) utilizes the full gradient at each step and can switch to a Newton–Raphson algorithm to speed up its convergence around the optimum. As the gradient lasso is scalable to high-dimensional data, so are the algorithms in Park and Hastie (2007) and Goeman (2008). Interested readers may see Park and Hastie (2007) and Goeman (2008) for more details on the algorithms.

2.2 Gradient lasso algorithm

Kim *et al*. (2008) proposed gradient lasso algorithm, which is a refined version of the early algorithm by Kim and Kim (2004). The algorithm can be applied to general convex loss functions. Moreover the algorithm is scalable to high-dimensional data because it requires only a univariate optimization at each iteration and its convergence speed is independent of the number of input variables.

For a given convex function $C(\boldsymbol{\beta})$ of $\boldsymbol{\beta} \in \mathbb{R}^p$, consider the minimization of $C(\boldsymbol{\beta})$ on $\mathcal{S} = \{\boldsymbol{\beta} : \|\boldsymbol{\beta}\|_1 \le s\}$. Note that $C(\boldsymbol{\beta}) = -l(\boldsymbol{\beta})$ in the lasso penalized Cox model. The gradient lasso solves the minimization problem by iteratively updating its solution through the addition and deletion steps.

In the addition step, a coordinate vector $\boldsymbol{v}$ at which $C(\boldsymbol{\beta})$ decreases most rapidly is sought. Let $\nabla C(\boldsymbol{\beta}) = (\partial C(\boldsymbol{\beta})/\partial\beta_1, \ldots, \partial C(\boldsymbol{\beta})/\partial\beta_p)^T$ be the gradient of $C(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$. By Taylor expansion, we have

$$C((1-\alpha)\boldsymbol{\beta} + \alpha\boldsymbol{v}) \approx C(\boldsymbol{\beta}) + \alpha\nabla C(\boldsymbol{\beta})^T(\boldsymbol{v} - \boldsymbol{\beta})$$

for $\boldsymbol{v} \in \mathcal{S}$ and $\alpha \in [0, 1]$, and

$$\min_{\boldsymbol{v}\in\mathcal{S}}\nabla C(\boldsymbol{\beta})^T\boldsymbol{v} = \min_{j=1,\ldots,p}\min\left\{\frac{\partial C(\boldsymbol{\beta})}{\partial\beta_j}, -\frac{\partial C(\boldsymbol{\beta})}{\partial\beta_j}\right\}.$$

Let

$$\hat{j} = \arg\max_{j=1,\ldots,p}\left\{\left|\frac{\partial C(\boldsymbol{\beta})}{\partial\beta_j}\right|\right\} \text{and}\ \hat{\gamma} = -s \times \text{sign}\left(\frac{\partial C(\boldsymbol{\beta})}{\partial\beta_j}\right).$$

Let $\hat{\boldsymbol{v}}$ be a vector such that its $\hat{j}$-th coordinate of $\boldsymbol{v}$ is $\hat{\gamma}$ and the other coordinates are zeros. The current solution $\boldsymbol{\beta}$ is updated as $(1-\hat{\alpha})\boldsymbol{\beta} + \hat{\alpha}\hat{\boldsymbol{v}}$, where the weight $\hat{\alpha}$ is determined by

$$\hat{\alpha} = \arg\min_{\alpha\in[0,1]} C((1-\alpha)\boldsymbol{\beta} + \alpha\hat{\boldsymbol{v}}). \tag{5}$$

Since we take a convex combination, the updated solution always satisfies the constraint $\mathcal{S}$.

1. **Initialize**: $\boldsymbol{\beta}=\mathbf{0}$ and $m=0$.
2. **Do** until convergence
   (a) **Addition:**
      i. Compute the gradient $\nabla C(\boldsymbol{\beta})$.
      ii. Find the $\hat{j}$ maximizing $|\partial C(\boldsymbol{\beta})/\partial\beta_j|$ for $j=1,\ldots,p$ and $\hat{\gamma}=-s\times\text{sign}(\partial C(\boldsymbol{\beta})/\partial\beta_{\hat{j}})$.
      iii. Let $\boldsymbol{v}$ be a p dimensional vector such that its $\hat{j}$-th element $\hat{\gamma}$ and the other elements are zeros.
      iv. Find $\hat{\alpha}=\text{arg}\min_{\alpha\in[0,1]}C\big((1-\alpha)\boldsymbol{\beta}+\alpha\boldsymbol{v}\big)$.
      v. Update $\boldsymbol{\beta}=(1-\hat{\alpha})\boldsymbol{\beta}+\hat{\alpha}\boldsymbol{v}$.
   (b) **Deletion:**
      i. Calculate $\boldsymbol{h}_\sigma$ as in (6).
      ii. Find $\hat{\delta}$ as in (7).
      iii. Update $\boldsymbol{\beta}=\boldsymbol{\beta}+\hat{\delta}\boldsymbol{h}$.
   (c) Set $m=m+1$.
3. **Return** $\boldsymbol{\beta}$.

Fig. 1. The gradient lasso algorithm.

The earlier version of the algorithm (Kim and Kim, 2004) consisting of only the addition step may converge slowly near the optimum. To resolve this problem, Kim *et al*. (2008) introduced the deletion step. The deletion step speeds up the algorithm by updating all the non-zero coordinate of $\boldsymbol{\beta}$ simultaneously. The active set σ is defined as $\sigma=\{j:\beta_j\neq 0\}$. Denote the current solution and corresponding gradient of the current solution with respect to an active set as $\boldsymbol{\beta}_\sigma$ and $\nabla C(\boldsymbol{\beta}_\sigma)$, respectively. Then we move $\boldsymbol{\beta}_\sigma$ toward the direction of $-\nabla C(\boldsymbol{\beta}_\sigma)$. If the current solution is placed on the surface of $\mathcal{S}$ with redundant non-zero coefficients, such a move may not satisfy the constraint $\mathcal{S}$. In order to make the move to satisfy the constraint, we replace $\nabla C(\boldsymbol{\beta}_\sigma)$ by

$$\boldsymbol{h}_\sigma=-\nabla C(\boldsymbol{\beta}_\sigma)+\theta_\sigma\nabla C(\boldsymbol{\beta}_\sigma)^T\theta_\sigma/|\sigma|, \tag{6}$$

where θ_σ is the sign vector of $\boldsymbol{\beta}_\sigma$ and $|\sigma|$ is the cardinality of σ. Let $U=\min_{k\in\sigma}\{-\beta_k/h_k:\beta_k h_k<0\}$. Let $\boldsymbol{P}$ denote the permutation matrix that collects the non-zero elements of $\boldsymbol{\beta}$ in the first $|\sigma|$ elements. Then we move $\boldsymbol{\beta}_\sigma$ toward $\boldsymbol{h}_\sigma$ until one of the non-zero coordinates becomes zero. That is, $\boldsymbol{\beta}$ is updated as $\boldsymbol{\beta}+\hat{\delta}\boldsymbol{h}$, where

$$\hat{\delta}=\text{arg}\min_{\delta\in[0,U]}C(\boldsymbol{\beta}+\delta\boldsymbol{h}) \tag{7}$$

and $\boldsymbol{h}=\boldsymbol{P}^T\begin{pmatrix}\boldsymbol{h}_\sigma\\ \mathbf{0}\end{pmatrix}$. When $\hat{\delta}=U$, one of non-zero coordinates is deleted. Figure 1 summarizes the gradient lasso algorithm.

2.3 Performance measures

In the microarray survival literature, the predictive performance of a model is typically assessed as follows: (i) risk scores based on the fitted model are computed for patients in a (withheld) test dataset, (ii) strata (usually two) are created based on thresholding these scores and (iii) log-rank testing of between-strata survival differences is performed. The greater the achieved significance the more predictive the model is deemed to be. In our data analysis, the threshold value of the predictive score is set to zero for dividing patients into low- and high-risk groups. Limitations of this approach include not just the arbitrariness of the imposed stratification but, more importantly, the familiar shortcoming of *P*-values not necessarily capturing effect size/explained variation. A more refined approach is afforded by the use of time-dependent ROC curves, proposed by Heagerty *et al*. (2000) and used in the present context by Gui and Li (2005b). Let $f(X)$ be the predictive model, where $X=(X_1,\ldots,X_p)$ denotes a p-dimensional random vector measuring the gene expression levels of an individual. Define time-dependent sensitivity and specificity functions at a cutoff point c as

$$\text{sensitivity}(c,t|f(X))=\mathbb{P}(f(X)>c|\Delta(t)=1) \tag{8}$$

$$\text{specificity}(c,t|f(X))=\mathbb{P}(f(X)\leq c|\Delta(t)=0), \tag{9}$$

where $\Delta(t)$ is the event indicator at time t. Throughout the article, the cutoff point c is set to zero as in Gui and Li (2005b) and Segal (2006). The corresponding time-dependent ROC curve at time t, ROC$(t|f(X))$, is then just the plot of sensitivity$(c,t|f(X))$ versus $1-$specificity$(c,t|f(X))$ over different values of cutoff point c. We adopt the analogous area under the time-dependent ROC$(t|f(X))$ curve, denoted as AUC$(t|f(X))$, in Section 3. In order to estimate the conditional probabilities in (8) and (9) accounting for possible censoring, we employ a nearest neighbor estimator for the bivariate distribution function (Akritas, 1994) as in Heagerty *et al*. (2000).

3 RESULTS

We compared the performance of a recent algorithm by Park and Hastie (2007), called the coxpath, the gradient algorithm by Goeman (2008), called the penalized, and our gradient lasso algorithm, called the glcoxph. Note that the coxpath and the penalized are available in the R package `glmpath` and `penalized`, respectively. The glcoxph package in R is available upon request from the authors. The glcoxph directly solves the constrained optimization problem in (2), while the coxpath and the penalized solve the equivalent unconstrained optimization (3).

Although there exists a one-to-one correspondence between the tuning parameters s in (2) and λ in (3), there is no closed conversion formula between them. Hence, we used different grids for tuning the coxpath and glcoxph. For the coxpath and the penalized, we conducted a grid search. More specifically, we extracted the maximum value, $\hat{\lambda}_{\max}$, of λ after fitting the coxpath on the training dataset, obtained the 100 grid points by dividing the interval $1\leq\lambda\leq\hat{\lambda}_{\max}$ equally in the log scale, and used those points in tuning. Note that the coefficients for $\lambda\geq\hat{\lambda}_{\max}$ are almost the same and thus the penalization path taken to be flat over the range of λ in the coxpath. For the glcoxph, the grid for the tuning parameter s was $0.05,0.1,\ldots,5$ as available in the previous studies such as Gui and Li (2005b). We adopted 5-fold CVPL as our criterion for the selection of tuning parameter.

3.1 Real microarray data

The diffuse large B-cell lymphoma (DLBCL) dataset consists of 240 samples from patients having DLBCL with expression measurements on 7399 genes. The clinical outcome was survival time, either observed or censored. We analyzed two different sets of the DLBCL data: one from Rosenwald *et al*. (2002) and the other from Bair *et al*. (2006). Both of the datasets consist of 160 training samples and 80 test samples.

Table 1 lists the genes selected by the glcoxph on the datasets from Rosenwald *et al*. (2002) and Bair *et al*. (2006). For the datasets defined in Rosenwald *et al*. (2002), eight genes were selected by the glcoxph. Four of those genes belong to the three gene expression signature groups, including Germinal-center B-cell signature, MHC class II signature and Lymph-node signature, defined in Rosenwald *et al*. (2002). Note that these four genes belong to the 10 genes selected by the LARS-Cox in Gui and Li (2005b) and are in fact

Table 1. Genes selected by the glcoxph on DLBCL datasets

Dataset	GenBank ID	$\hat{\beta}$	Signature	Description
	AA805575	−0.190	Germ	Thyroxine-binding globulin precursor
	X00452	−0.163	MHC	Major histocompatibility complex, class II, DQ alpha 1
	LC_29222	−0.148	Lymph	
Rosenwald *et al*. (2002)	X59812	−0.076	Lymph	Cytochrome P450, subfamily XXVIIA polypeptide 1
	AA729003	−0.062		T-cell leukemia/lymphoma 1A
	M15800	−0.052		Mal, T-cell differentiation protein
	AA291844	0.016		Immunoglobulin kappa constant
	AA760674	0.111		COX15 homolog, cytochrome c oxidase assembly protein (yeast)
	AA805575	−0.609	Germ	Thyroxine-binding globulin precursor
	M81750	−0.117		Myeloid cell nuclear differentiation antigen
Bair *et al*. (2006)	AA729055	−0.086	MHC	Major histocompatibility complex, class II, DR alpha
	AA809474	0.042		Immunoglobulin heavy constant mu
	AA485725	0.140		Immunoglobulin kappa constant

Table 2. The number of genes selected and the partial log-likelihood on DLBCL training datasets

	Rosenwald *et al*. (2002)			Bair *et al*. (2006)		
	coxpath	glcoxph	penalized	coxpath	glcoxph	penalized
No. of genes	14	8	16	5	5	5
$l(\boldsymbol{\beta})$	−391.1635	−383.8169	−387.0456	−432.3464	−429.5923	−432.3153

Table 3. Log-rank test results on DLBCL test datasets

	Rosenwald *et al*. (2002)			Bair *et al*. (2006)		
	coxpath	glcoxph	penalized	coxpath	glcoxph	penalized
No. of high risk	38	31	32	42	33	35
Log-rank *P*-value	0.0005	0.0000	0.0009	0.0040	0.0000	0.0027

the top four genes selected by the residual finesse in Segal (2006). The other four genes do not belong to the signature groups defined in Rosenwald *et al*. (2002). Among those four genes, AA729003, a protein coding TCL1A gene and AA760674, a COX15 homolog, were also selected by the LARS-Cox in Gui and Li (2005b). The glcoxph has selected five genes from the datasets by Bair *et al*. (2006). Two of those genes belong to the two gene expression signature groups, including Germinal-center B-cell signature and MHC class II signature, defined in Rosenwald *et al*. (2002).

Table 2 summarizes the number of selected genes and the partial log-likelihood by the coxpath, glcoxph and penalized on DLBCL training datasets. For the training data from Rosenwald *et al*. (2002), the coxpath and the glcoxph selected 14, 8 and 16 genes, respectively, and seven genes were selected by all the methods. Only by the coxpath, glcoxph and penalized algorithms, 1, 1 and 3 genes were selected, respectively. Interestingly, all the methods selected the same five genes on the training data from Bair *et al*. (2006). Our method yielded a sparser model than the other methods for both of DLBCL datasets. Comparing the values of $l(\boldsymbol{\beta})$ on the training data, we observe that the glcoxph yielded slightly larger value of $l(\boldsymbol{\beta})$ than the other methods. The optimal values of tuning parameters on the training datasets from Rosenwald *et al*. (2002) were 0.8, 25.2068 and 23.1014 for the glcoxph, the coxpath and the penalized, respectively. For the second training dataset from Bair *et al*. (2006), the optimal values were 0.2 for the glcoxph and 31.2733 for the coxpath and the penalized.

We compared the predictive performances of the methods. We stratified the test data into low- and high-risk groups. Table 3 shows the results by the log-rank test. The *P*-value of the glcoxph was a bit smaller than the others on both of the DLBCL datasets.

Time-dependent AUC curves in Figure 2 indicate that the glcoxph outperforms the other methods over the time span and the predictive performance of the penalized and the coxpath are almost the same. Note that the curve for the coxpath and the penalized in Figure 2b are identical. In summary, for the DLBCL datasets, the glcoxph predicts the survival distribution better with a smaller number of genes than the penalized or the coxpath.

Kaplan–Meier curves on the DLBCL datasets are shown in Figure 3. For the first dataset, the Kaplan–Meier curves have similar patterns except that the survival probability of the coxpath for the

high-risk patients has a longer tail than the glcoxph or the penalized. For the second dataset, the three methods show almost the same patterns for the low-risk patients. For the high-risk patients, the patterns of the methods are very similar up to 10 years. After 10 years, the survival curve for the coxpath decays very slowly, while that of the penalized drops quickly.

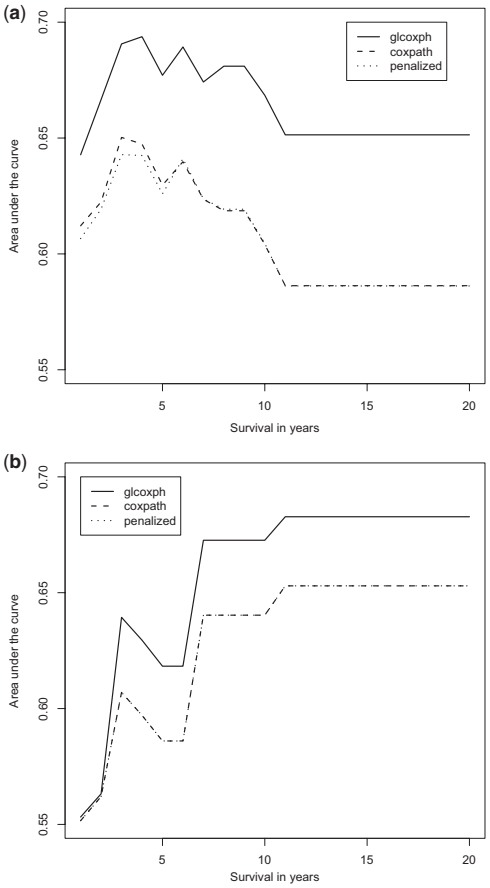

Fig. 2. AUC on DLBCL test datasets (**a**) Rosenwald *et al*. (2002) and (**b**) Bair *et al*. (2006).

The experiments were implemented in a Windows R environment on a computer with 2 GHz Intel Pentium processor and 1 GB RAM. We compared the computational times on the DLBCL dataset from Rosenwald *et al*. (2002). It took 4449.72 s (3094.24 s in tuning and 1355.38 s in training at the optimal value of tuning parameter), 2904.53 s (2903.84 s in tuning and 0.69 s in training), and 59432.29 s (59361.04 s in tuning and 71.25 s in training) seconds for the coxpath, the glcoxph and the penalized, respectively. While glcoxph fits the model at each fixed value of tuning parameter, the coxpath computes the entire penalization path over the range of s values with a specific step size. Consequently, a direct comparison of their computing times may not be so meaningful. However, at least we can say that the glcoxph is very competitive in terms of computing time.

In order to compare the performance of these methods further, we randomly partitioned the DLBCL data and the Norway/Stanford breast cancer data from Sørlie *et al*. (2003). The latter consist of 115 samples from women with breast cancer with expression measurements on 549 genes. The DLBCL dataset was randomly partitioned into 160 training samples and 80 test samples, and the Norway/Stanford breast cancer dataset was partitioned into 76 training samples and 39 test samples. We compared the predictive performance of the methods over 50 random partitions. Table 4 summarizes the average performance measures (and their standard errors in the parentheses) over the chosen random partitions.

In terms of the P-value and the AUC, the differences of the methods were negligible because the numbers are within the error range. However, the penalized was worse than the other methods in terms of the P-value on both datasets. The numbers of selected genes by the glcoxph and the coxpath were not much different, but fewer than that by the penalized. In summary, the glcoxph showed similar performance as the coxpath and the glcoxph outperformed the penalized in prediction and in sparsity of the resulting predictive model.

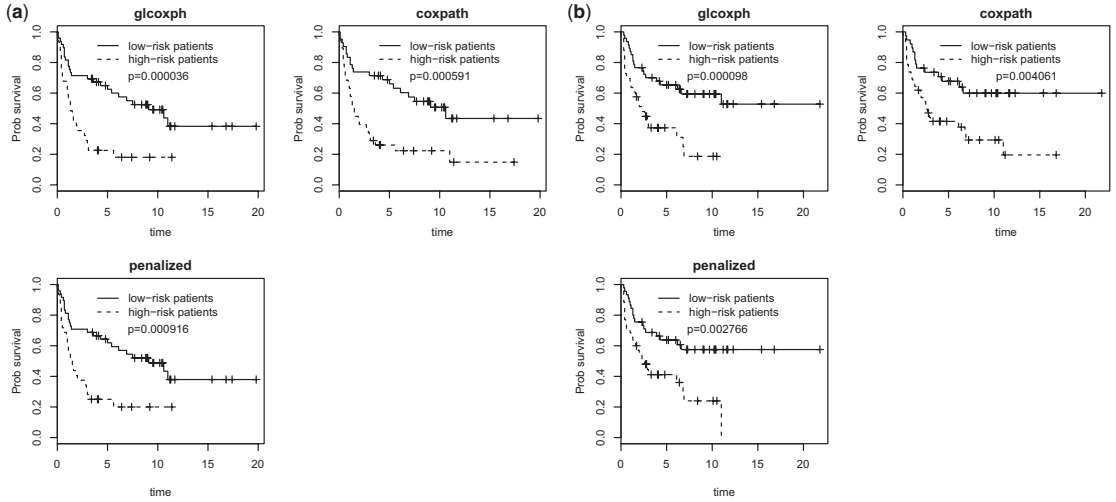

Fig. 3. Kaplan–Meier curves on DLBCL test datasets (**a**) Rosenwald *et al*. (2002) and (**b**) Bair *et al*. (2006).

Table 4. Results on the DLBCL and Norway/Stanford breast cancer datasets

	Average no. of genes selected			Average *P*-value			Average AUC		
Dataset	coxpath	glcoxph	penalized	coxpath	glcoxph	penalized	coxpath	glcoxph	penalized
DLBCL	13.30 (1.6311)	13.54 (1.2438)	23.12 (1.7976)	0.0590 (0.0123)	0.0672 (0.0143)	0.1020 (0.0275)	0.5917 (0.0185)	0.6015 (0.0188)	0.5678 (0.0216)
Norway/Stanford breast cancer	9.36 (0.8411)	9.12 (0.7332)	13.56 (0.9652)	0.0653 (0.0202)	0.0474 (0.0147)	0.1081 (0.0248)	0.6131 (0.0098)	0.6119 (0.0101)	0.6120 (0.0091)

Table 5. Average number of genes and prognostic genes on the simulated data

		coxpath			glcoxph			penalized		
Censoring (%)	ρ	No. of genes	No. of prognostic genes	Recovery rate	No. of genes	No. of prognostic genes	Recovery rate	No. of genes	No. of prognostic genes	Recovery rate
30	0	21.04	2.86	0.13	16.32	2.86	0.18	30.16	2.98	0.09
	0.3	23.22	2.96	0.12	16.80	2.96	0.18	30.22	2.98	0.09
	0.6	22.54	2.94	0.13	15.70	2.94	0.19	28.68	2.94	0.10
15	0	23.29	2.96	0.12	17.28	2.96	0.17	29.90	2.94	0.09
	0.3	24.24	2.92	0.12	15.82	2.92	0.18	28.26	2.92	0.10
	0.6	22.22	2.92	0.13	15.26	2.92	0.19	30.64	2.98	0.09

The recovery rate is defined as the ratio of the average number of prognostic genes to the average number of genes selected.

3.2 Simulated data

To evaluate the performance of the glcoxph empirically, we carried out a simulation study. We have modified the simulation scheme in Owzar *et al*. (2007) slightly. The data generation scheme is as follows. For $i=1,\ldots,N$, we generate $\epsilon_{i0},\epsilon_{i1},\ldots,\epsilon_{ip}$ independently from standard normal distribution and set

$$x_{ij}=\epsilon_{ij}\sqrt{1-\rho}+\epsilon_{i0}\sqrt{\rho}, \quad j=1,\ldots,p.$$

The gene expression data have a block exchangeable correlation structure (i.e. genes in the same block are correlated and in different blocks uncorrelated) with the correlation coefficient ρ and the block size 10. Survival times are generated from the Cox regression model (1) in the following configuration of coefficients:

$$\beta_j = \begin{cases} -0.7, & j=1,\ldots,D \\ 0, & j=D+1,\ldots,p, \end{cases}$$

where D denotes the number of prognostic genes.

For our experiment, we have set $p=1000$, $N=200$ and $D=3$. We considered the cases with 15% and 30% of censoring and $\rho=0,0.3$ and 0.6. A censoring time was generated from $U(0,a)$, where a was chosen to yield ~30% of censoring. With a fixed at this value, a censoring variable was generated from $U(b,a+b)$. Here, b was chosen to yield about 15% of censoring. We have generated 200 random samples. Hundred samples were used for training and the others for testing. To assess the variability of the experiment, we have replicated the above process 100 times.

We compared our method with the coxpath and the penalized. The tuning parameters were chosen by 5-fold CVPL as stated at the beginning of Section 3. Table 5 shows the average number of genes and prognostic genes selected by the coxpath and the glcoxph. Overall the penalized selected the largest number of genes and the glcoxph selected the least. All the methods selected almost all the three prognostic genes. The recovery rate of glcoxph was the highest, the coxpath was the second and the penalized was the lowest.

Table 6 shows average *P*-value of the log-rank test and AUC on the simulated data with standard errors in parentheses. All three methods perform better at 15% censoring than at 30% censoring as expected. In terms of *P*-value, the glcoxph performed a bit better than the other methods. The penalized yielded slightly larger AUC values than the others. However, their differences seem to be marginal. Combined with the results reported in Table 6, we conclude that the glcoxph showed similar predictive performance as the other methods with a more succinct model and a shorter computing time.

4 DISCUSSION

We applied the gradient lasso algorithm by Kim *et al*. (2008) to the Cox proportional hazards model. Most of the algorithms for the Cox model with lasso penalty suffer from severe computational problems. Due to matrix inversions, those algorithms converge very slowly or even fail to converge to the optimum. Meanwhile, the gradient lasso algorithm is scalable to high-dimensional data because it does not require matrix inversions. Also the gradient lasso tackles the exact penalized partial likelihood directly, while many other methods solve approximated penalized partial likelihoods. We have compared the gradient lasso algorithm with recent algorithms by Park and Hastie (2007) and Goeman (2008) on DLBCL datasets from Rosenwald *et al*. (2002) and Bair *et al*. (2006), Norway/Stanford breast cancer dataset from Sørlie *et al*. (2003) and also on a simulated dataset. Results indicate that our gradient lasso algorithm is very competitive in analyzing high-dimensional survival data in terms of

Table 6. Average *P*-values of log-rank test and AUC on the simulated data with standard errors in parentheses

		coxpath		glcoxph		penalized	
Censoring (%)	ρ	*P*-value	AUC	*P*-value	AUC	*P*-value	AUC
30	0	0.0183 (0.0090)	0.6452 (0.0132)	0.0113 (0.0071)	0.6444 (0.0052)	0.0232 (0.0096)	0.6546 (0.0018)
	0.3	0.0038 (0.0017)	0.6524 (0.0110)	0.0039 (0.0032)	0.6582 (0.0046)	0.0028 (0.0071)	0.6677 (0.0136)
	0.6	0.0013 (0.0004)	0.6588 (0.0102)	0.0010 (0.0004)	0.6603 (0.0051)	0.0026 (0.0013)	0.6659 (0.0172)
15	0	0.0028 (0.0008)	0.6525 (0.0091)	0.0015 (0.0007)	0.6768 (0.0046)	0.0092 (0.0064)	0.6685 (0.0086)
	0.3	0.0020 (0.0013)	0.6798 (0.0041)	0.0011 (0.0005)	0.6851 (0.0051)	0.0008 (0.0004)	0.6691 (0.0065)
	0.6	0.0002 (0.0001)	0.6679 (0.0092)	0.0005 (0.0003)	0.6779 (0.0046)	0.0005 (0.0004)	0.6827 (0.0061)

sparsity of the final prediction model, predictability and computing time. Since our algorithm can yield a more stable solution to the penalized Cox model in a faster manner than other methods in the literature, it will provide an efficient tool in developing a prediction model for survival time based on high-dimensional microarray data.

There are several issues yet to be investigated. We considered relating only gene expression profiles to survival phenotypes in our study. In the Cox proportional hazards model, the incorporation of some categorical clinical variables is anticipated. One may consider borrowing ideas from the blockwise sparse regression, an extension of the group lasso in Yuan and Lin (2006) to general loss functions, proposed by Kim *et al*. (2006). A non-parametric extension of the penalized Cox regression is another direction of interest. Liu (2008) proposed a garrote method of feature selection in the penalized partial likelihood of additive Cox models. An advantage of the garrote method is that it can deal with continuous as well as categorical variables. However, it seems that there is room for improvement in the computational aspect of the garrote method.

ACKNOWLEDGEMENTS

The authors are grateful to Dr M. Y. Park for personal communications on coxpath.

Funding: Korea Research Foundation Grant (KRF-2008-331-C00055) by the Korean Government.

Conflict of Interest: none declared.

REFERENCES

Akritas,M.G. (1994) Nearest neighbor estimation of a bivariate distribution under random censoring. *Ann. Stat.*, **22**, 1299–1327.

Bair,E. and Tibshirani,R. (2004) Semi-supervised methods to predict patient survival from gene expression data. *PLoS Biol.*, **2**, 511–522.

Bair,E. *et al*. (2006) Prediction by supervised principal components. *J. Am. Stat. Assoc.*, **101**, 119–137.

Brown,P.O. and Botstein,D. (1999) Exploring the new world of the genome with DNA microarrays. *Nat. Genet.*, **21** (Suppl. 1), 33–37.

Cox,D.R. (1972) Regression models and life-tables. *J. R. Stat. Soc. B*, **34**, 187–220.

Efron,B. *et al.* (2004) Least angle regression. *Ann. Stat.*, **32**, 407–499.

Fan,J. and Li,R. (2002) Variable selection for Cox's proportional hazards model and frailty model. *Ann. Stat.*, **30**, 74–99.

Goeman,J.J. (2008) An efficient algorithm for L_1-penalized estimation. Preprint, Department of Medical Statistics and Bioinformatics, Leiden University, Netherlands.

Gui,J. and Li,H. (2005a) Threshold gradient descent method for censored data regression, with Applications in Pharmacogenomics. In *Pacific Symposium on Biocomputing*, World Scientific, Singapore, pp. 272–283.

Gui,J. and Li,H. (2005b) Penalized Cox regression analysis in the high-dimensional and low-sample size settings, with applications to microarray gene expression data. *Bioinformatics*, **21**, 3001–3008.

Heagerty,P.J. *et al*. (2000) Time dependent ROC curves for censored survival data and a diagnostic marker. *Biometrics*, **56**, 337–344.

Kaderali,L. *et al*. (2006) CASPAR: a hierarchical Bayesian approach to predict survival times in cancer from gene expression data. *Bioinformatics*, **22**, 1495–1502.

Kim,J. and Kim,Y. (2004) Gradient lasso for feature selection. In *Proceedings of the 21th International Conference on Machine Learning*. Morgan Kaufmann, ACM, New York, pp. 473–480.

Kim,Y. *et al.* (2006) Blockwise sparse regression. *Stat. Sin.*, **16**, 375–390.

Kim,J. *et al*. (2008) A gradient-based optimization algorithm for lasso. *J. Comput. Graph. Stat.*, **17**, 994–1009.

Liu, S. (2008) Variable selection in semi-parametric additive models with extensions to high dimensional data and addtive Cox models. Ph.D. Thesis, Department of Statistics, North Carolina State University.

Ma,S. and Huang,J. (2005) Lasso method for additive risk models with high dimensional covariates. *Technical Report No. 347*, Department of Statistics and Actuarial Science, The University of Iowa.

Owzar,K. *et al*. (2007) A coupula approach for detecting prognostic genes associated with survival outcome in microarray studies. *Biometrics*, **63**, 1089–1098.

Park,M.Y. and Hastie,T (2007) L1 regularization path algorithm for generalized linear models. *J. R. Stat. Soc. B*, **69**, 659–677.

Rosenwald,A. *et al*. (2002) The use of molecular profiling to predict survival after chemotherapy for diffuse large B-cell lymphoma. *New Engl. J. Med.*, **346**, 1937–1946.

Segal,M.R. (2006) Microarray gene expression data with linked survival phenotypes: diffuse large-B-cell lymphoma revisited. *Biostatistics*, **7**, 268–285.

Sørlie,T. *et al*. (2003) Repeated observation of breast tumor subtypes in independent gene expression datasets. *Proc. Natl Acad. Sci. USA*, **100**, 8418–8423.

Tibshirani,R. (1997) The lasso method for variable selection in the Cox model. *Stat. Med.*, **16**, 385–395.

Yuan,M. and Lin,Y. (2006) Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. B*, **68**, 49–67.

Zhang,H.H. and Lu,W. (2007) Adaptive lasso for Cox's proportional hazards model. *Biometrika*, **94**, 691–703.

Zou,H. (2008) A note on path-based variable selection in the penalized proportional hazards model. *Biometrika*, **95**, 241–247.