

# The group lasso for logistic regression

Lukas Meier, Sara van de Geer and Peter Bühlmann

*Eidgenössische Technische Hochschule, Zürich, Switzerland*

[Received March 2006. Final revision July 2007]

**Summary.** The group lasso is an extension of the lasso to do variable selection on (predefined) groups of variables in linear regression models. The estimates have the attractive property of being invariant under groupwise orthogonal reparameterizations. We extend the group lasso to logistic regression models and present an efficient algorithm, that is especially suitable for high dimensional problems, which can also be applied to generalized linear models to solve the corresponding convex optimization problem. The group lasso estimator for logistic regression is shown to be statistically consistent even if the number of predictors is much larger than sample size but with sparse true underlying structure. We further use a two-stage procedure which aims for sparser models than the group lasso, leading to improved prediction performance for some cases. Moreover, owing to the two-stage nature, the estimates can be constructed to be hierarchical. The methods are used on simulated and real data sets about splice site detection in DNA sequences.

**Keywords:** Categorical data; Co-ordinate descent algorithm; DNA splice site; Group variable selection; High dimensional generalized linear model; Penalized likelihood

## 1. Introduction

The lasso (Tibshirani, 1996), which was originally proposed for linear regression models, has become a popular model selection and shrinkage estimation method. In the usual linear regression set-up we have a continuous response  $\mathbf{Y} \in \mathbb{R}^n$ , an  $n \times p$  design matrix  $\mathbf{X}$  and a parameter vector  $\boldsymbol{\beta} \in \mathbb{R}^p$ . The lasso estimator is then defined as

$$\hat{\boldsymbol{\beta}}_\lambda = \arg \min_{\boldsymbol{\beta}} (\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \sum_{j=1}^p |\beta_j|),$$

where  $\|\mathbf{u}\|_2^2 = \sum_{i=1}^n u_i^2$  for a vector  $\mathbf{u} \in \mathbb{R}^n$ . For large values of the penalty parameter  $\lambda$ , some components of  $\hat{\boldsymbol{\beta}}_\lambda$  are set exactly to 0. The  $l_1$ -type penalty of the lasso can also be applied to other models as for example Cox regression (Tibshirani, 1997), logistic regression (Lokhorst, 1999; Roth, 2004; Shevade and Keerthi, 2003; Genkin *et al.*, 2007) or multinomial logistic regression (Krishnapuram *et al.*, 2005) by replacing the residual sum of squares by the corresponding negative log-likelihood function.

Already for the special case in linear regression when not only continuous but also categorical predictors (factors) are present, the lasso solution is not satisfactory as it only selects individual dummy variables instead of whole factors. Moreover, the lasso solution depends on how the dummy variables are encoded. Choosing different contrasts for a categorical predictor will produce different solutions in general. The **group lasso** (Yuan and Lin, 2006; Bakin, 1999; Cai, 2001; Antoniadis and Fan, 2001) overcomes these problems by introducing a suitable extension

*Address for correspondence:* Lukas Meier, Seminar für Statistik, Eidgenössische Technische Hochschule Zürich, Leonhardstrasse 27, CH-8092 Zürich, Switzerland.  
E-mail: meier@stat.math.ethz.ch

of the lasso penalty. The estimator is defined as

$$\hat{\beta}_\lambda = \arg \min_{\beta} (\|Y - X\beta\|_2^2 + \lambda \sum_{g=1}^G \|\beta_{\mathcal{I}_g}\|_2),$$

where  $\mathcal{I}_g$  is the index set belonging to the  $g$ th group of variables,  $g = 1, \dots, G$ . This penalty can be viewed as an intermediate between the  $l_1$ - and  $l_2$ -type penalty. It has the attractive property that it does variable selection at the group level and is invariant under (groupwise) orthogonal transformations like ridge regression (Yuan and Lin, 2006).

This paper deals with the group lasso penalty for logistic regression models. The logistic case calls for new computational algorithms. Kim *et al.* (2006) first studied the group lasso for logistic regression models and proposed a gradient descent algorithm to solve the corresponding constrained problem. We present methods which allow us to work directly on the penalized problem and whose convergence property does not depend on unknown constants as in Kim *et al.* (2006). Our algorithms are efficient in the sense that they can handle problems where  $p$  and  $n$  are large. Furthermore, they are also applicable to generalized linear models, beyond the case of logistic regression. We do not aim for an (approximate) [path following algorithm](#) (Rosset, 2005; Zhao and Yu, 2004; Park and Hastie, 2006, 2007) but our approaches are sufficiently fast for computing a whole range of solutions for varying penalty parameters on a (fixed) grid. Our approach is related to Genkin *et al.* (2007) which presented an impressively fast implementation ('the fastest') for large-scale logistic regression with the lasso; in fact, we can also deal with dimensionality  $p$  in the 10000s but now for the group lasso. Moreover, we present an asymptotic consistency theory for the group lasso in high dimensional problems where the predictor dimension is much larger than the sample size. This has neither been developed for linear nor for logistic regression. High dimensionality of the predictor space arises in many applications, in particular with higher order interaction terms or basis expansions for logistic additive models where the groups correspond to the basis functions for individual continuous covariates. Our application about the detection of splice sites, the regions between coding (exons) and non-coding (introns) deoxyribonucleic acid (DNA) segments involves the categorical predictor space  $\{A, C, G, T\}^7$  which has cardinality 16384.

The rest of this paper is organized as follows. In Section 2 we restate in more detail the idea of the group lasso for logistic regression models, present two efficient algorithms which are proven to solve the corresponding convex optimization problem and compare them with other optimization methods. Furthermore, we show that the group lasso estimator is statistically consistent for high dimensional sparse problems. In Section 3 we outline a two-stage procedure which often produces more adequate models in terms of both model size and prediction performance. Simulations follow in Section 4 and an application of the modelling of functional DNA sites can be found in Section 5. Section 6 contains the discussion. All proofs are given in Appendix A.

## 2. Logistic group lasso

### 2.1. Model set-up

Assume that we have independent and identically distributed observations  $(\mathbf{x}_i, y_i)$ ,  $i = 1, \dots, n$ , of a  $p$ -dimensional vector  $\mathbf{x}_i \in \mathbb{R}^p$  of  $G$  predictors and a binary response variable  $y_i \in \{0, 1\}$ . Both categorical and continuous predictors are allowed. We denote by  $\text{df}_g$  the degrees of freedom of the  $g$ th predictor and can thus rewrite  $\mathbf{x}_i = (\mathbf{x}_{i,1}^T, \dots, \mathbf{x}_{i,G}^T)^T$  with the group of variables  $\mathbf{x}_{i,g} \in \mathbb{R}^{\text{df}_g}$ ,  $g = 1, \dots, G$ . For example, the main effect of a factor with four levels has  $\text{df} = 3$  whereas a continuous predictor involves  $\text{df} = 1$  only.

Linear logistic regression models the conditional probability  $p_{\beta}(\mathbf{x}_i) = \mathbb{P}_{\beta}(Y = 1|\mathbf{x}_i)$  by

$$\log \left\{ \frac{p_{\beta}(\mathbf{x}_i)}{1 - p_{\beta}(\mathbf{x}_i)} \right\} = \eta_{\beta}(\mathbf{x}_i), \quad (2.1)$$

with

$$\eta_{\beta}(\mathbf{x}_i) = \beta_0 + \sum_{g=1}^G \mathbf{x}_{i,g}^T \boldsymbol{\beta}_g,$$

where  $\beta_0$  is the intercept and  $\boldsymbol{\beta}_g \in \mathbb{R}^{\text{df}_g}$  is the parameter vector corresponding to the  $g$ th predictor. We denote by  $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$  the whole parameter vector, i.e.  $\boldsymbol{\beta} = (\beta_0, \boldsymbol{\beta}_1^T, \dots, \boldsymbol{\beta}_G^T)^T$ .

The logistic group lasso estimator  $\hat{\boldsymbol{\beta}}_{\lambda}$  is given by the minimizer of the convex function

$$S_{\lambda}(\boldsymbol{\beta}) = -l(\boldsymbol{\beta}) + \lambda \sum_{g=1}^G s(\text{df}_g) \|\boldsymbol{\beta}_g\|_2, \quad (2.2)$$

where  $l(\cdot)$  is the log-likelihood function, i.e.

$$l(\boldsymbol{\beta}) = \sum_{i=1}^n y_i \eta_{\beta}(\mathbf{x}_i) - \log[1 + \exp\{\eta_{\beta}(\mathbf{x}_i)\}].$$

The tuning parameter  $\lambda \geq 0$  controls the amount of penalization. Note that we do not penalize the intercept. However, as shown in lemma 1, the minimum in equation (2.2) is attained. The function  $s(\cdot)$  is used to rescale the penalty with respect to the dimensionality of the parameter vector  $\boldsymbol{\beta}_g$ . Unless stated otherwise, we use  $s(\text{df}_g) = \text{df}_g^{1/2}$  to ensure that the penalty term is of the order of the number of parameters  $\text{df}_g$ . The same rescaling was used in Yuan and Lin (2006).

*Lemma 1.* Assume that  $0 < \sum_{i=1}^n y_i < n$ . For  $\lambda > 0$  and  $s(d) > 0$  for all  $d \in \mathbb{N}$ , the minimum in optimization problem (2.2) is attained.

The first condition in lemma 1 is a minimal requirement for the observed data. If the design matrix  $X$  has full rank, the minimizer of  $S_{\lambda}(\cdot)$  is unique. Otherwise, the set of minimizers is a convex set whose elements correspond to the same minimum value of  $S_{\lambda}(\cdot)$ .

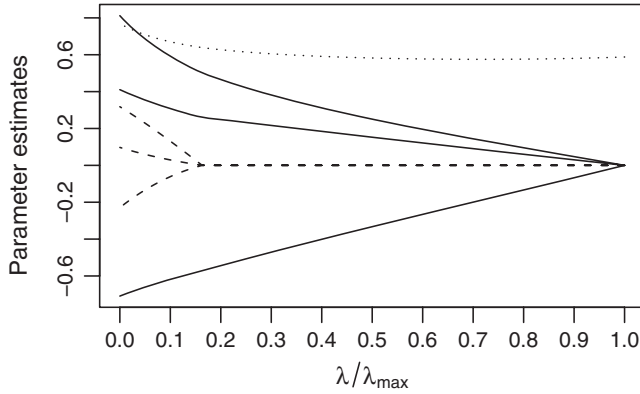
The ‘groupwise’  $l_2$ -norm in equation (2.2) is an intermediate between the lasso and the ridge penalty function. It encourages that in general either  $\hat{\boldsymbol{\beta}}_g = \mathbf{0}$  or  $\hat{\boldsymbol{\beta}}_{g,j} \neq 0$  for all  $j \in \{1, \dots, \text{df}_g\}$ , where we have omitted the index  $\lambda$  for easier notation. A geometrical interpretation of this special sparsity property was given in Yuan and Lin (2006). An example of a solution path  $\{\hat{\boldsymbol{\beta}}_{\lambda}\}_{\lambda \geq 0}$  for a model consisting of an intercept and two factors having 3 degrees of freedom each is depicted in Fig. 1.

Let the  $n \times \text{df}_g$  matrix  $X_g$  be the columns of the design matrix corresponding to the  $g$ th predictor. If we assume that the block matrices  $X_g$  are of full rank, we can perform a (blockwise) orthonormalization—e.g. by a  $QR$ -decomposition—to obtain  $X_g^T X_g = I_{\text{df}_g}$ ,  $g = 1, \dots, G$ . Using such a design matrix, the group lasso estimator does not depend on the encoding scheme of the dummy variables. We choose a rescaled version  $X_g^T X_g = n I_{\text{df}_g}$  to ensure that the parameter estimates are on the same scale when varying the sample size  $n$ . After parameter estimation, the estimates must be transformed back to correspond to the original encoding.

## 2.2. Algorithms for the logistic group lasso

### 2.2.1. Block co-ordinate descent

Parameter estimation is computationally more demanding than for linear regression models. The algorithm that was presented in Yuan and Lin (2006) sequentially solves a system of (neces-



**Fig. 1.** Solution path  $\{\hat{\beta}_\lambda\}_{\lambda \geq 0}$  for a model consisting of an intercept ( $\cdots$ ) and two factors having 3 degrees of freedom each ( $- - -$ ,  $—$ ):  $\lambda_{\max}$  is the value of the penalty parameter  $\lambda$  such that no penalized group is active in the model

sary and sufficient) non-linear equations which corresponds to a groupwise minimization of the penalized residual sum of squares. Hence, the algorithm is a special case of a block co-ordinate descent algorithm. No result on numerical convergence was given in Yuan and Lin (2006).

For the more difficult case of logistic regression, we can also use a block co-ordinate descent algorithm and we prove numerical convergence by using the results of Tseng (2001) as shown in proposition 1. The key lies in the separable structure of the non-differentiable part in  $S_\lambda(\cdot)$ . These properties of course also apply to the group lasso for linear and generalized linear regression models.

We cycle through the parameter groups and minimize the objective function  $S_\lambda(\cdot)$ , keeping all except the current parameter group fixed. This leads us to the algorithm that is presented in Table 1, where we denote by  $\beta_{-g}$  the parameter vector  $\beta$  when setting  $\beta_g$  to  $\mathbf{0}$  while all other components remain unchanged. In step 3 we first check whether the minimum is at the non-differentiable point  $\beta_g = \mathbf{0}$ . If not, we can use a standard numerical minimizer, e.g. a Newton-type algorithm, to find the optimal solution with respect to  $\beta_g$ . In such a case the values of the last iteration can be used as starting values to save computing time. If the group was not in the model in the last iteration, we first go a small step in the opposite direction of the gradient of the negative log-likelihood function to ensure that we start at a differentiable point.

**Table 1.** Logistic group lasso algorithm using block co-ordinate descent minimization

Step	Algorithm
1	Let $\beta \in \mathbb{R}^{p+1}$ be an initial parameter vector
2	$\beta_0 \leftarrow \arg \min_{\beta_0} \{S_\lambda(\beta)\}$
3	For $g = 1, \dots, G$ <ul style="list-style-type: none"> <li>if <math>\ X_g^T(y - \mathbf{p}_{\beta_{-g}})\ _2 \leq \lambda s(df_g)</math> <ul style="list-style-type: none"> <li><math>\beta_g \leftarrow \mathbf{0}</math></li> </ul> </li> <li>else <ul style="list-style-type: none"> <li><math>\beta_g \leftarrow \arg \min_{\beta_g} \{S_\lambda(\beta)\}</math></li> </ul> </li> <li>end</li> </ul>
4	Repeat steps 2 and 3 until some convergence criterion is met

*Proposition 1.* Steps 2 and 3 of the block co-ordinate descent algorithm perform groupwise minimizations of  $S_\lambda(\cdot)$  and are well defined in the sense that the corresponding minima are attained. Furthermore, if we denote by  $\hat{\beta}^{(t)}$  the parameter vector after  $t$  block updates, then every limit point of the sequence  $\{\hat{\beta}^{(t)}\}_{t \geq 0}$  is a minimum point of  $S_\lambda(\cdot)$ .

Because the iterates can be shown to stay in a compact set, the existence of a limit point is guaranteed.

The main drawback of such an algorithm is that the blockwise minimizations of the active groups must be performed numerically. However, for small and moderate-sized problems in the dimension  $p$  and the group sizes  $\text{df}_g$  this turns out to be sufficiently fast. For large-scale applications it would be attractive to have a closed form solution for a block update as in Yuan and Lin (2006). This will be discussed in the next subsection.

### 2.2.2. Block co-ordinate gradient descent

The key idea of the block co-ordinate gradient descent method of Tseng and Yun (2007) is to combine a quadratic approximation of the log-likelihood with an additional line search. Using a second-order Taylor series expansion at  $\hat{\beta}^{(t)}$  and replacing the Hessian of the log-likelihood function  $l(\cdot)$  by a suitable matrix  $H^{(t)}$  we define

$$\begin{aligned} M_\lambda^{(t)}(\mathbf{d}) &= -\{l(\hat{\beta}^{(t)}) + \mathbf{d}^T \nabla l(\hat{\beta}^{(t)}) + \frac{1}{2} \mathbf{d}^T H^{(t)} \mathbf{d}\} + \lambda \sum_{g=1}^G s(\text{df}_g) \|\hat{\beta}_g^{(t)} + \mathbf{d}_g\|_2 \\ &\approx S_\lambda(\hat{\beta}^{(t)} + \mathbf{d}), \end{aligned} \quad (2.3)$$

where  $\mathbf{d} \in \mathbb{R}^{p+1}$ . Now we consider the minimization of  $M_\lambda^{(t)}(\cdot)$  with respect to the  $g$ th penalized parameter group. This means that we restrict ourselves to vectors  $\mathbf{d}$  with  $\mathbf{d}_k = \mathbf{0}$  for  $k \neq g$ . Moreover, we assume that the corresponding  $\text{df}_g \times \text{df}_g$  submatrix  $H_{gg}^{(t)}$  is of the form  $H_{gg}^{(t)} = h_g^{(t)} I_{\text{df}_g}$  for some scalar  $h_g^{(t)} \in \mathbb{R}$ .

If  $\|\nabla l(\hat{\beta}^{(t)})_g - h_g^{(t)} \hat{\beta}_g^{(t)}\|_2 \leq \lambda s(\text{df}_g)$ , the minimizer of equation (2.3) is

$$\mathbf{d}_g^{(t)} = -\hat{\beta}_g^{(t)}.$$

Otherwise

$$\mathbf{d}_g^{(t)} = -\frac{1}{h_g^{(t)}} \left\{ \nabla l(\hat{\beta}^{(t)})_g - \lambda s(\text{df}_g) \frac{\nabla l(\hat{\beta}^{(t)})_g - h_g^{(t)} \hat{\beta}_g^{(t)}}{\|\nabla l(\hat{\beta}^{(t)})_g - h_g^{(t)} \hat{\beta}_g^{(t)}\|_2} \right\}.$$

If  $\mathbf{d}^{(t)} \neq \mathbf{0}$ , an inexact line search using the Armijo rule must be performed: let  $\alpha^{(t)}$  be the largest value in  $\{\alpha_0 \delta^l\}_{l \geq 0}$  such that

$$S_\lambda(\hat{\beta}^{(t)} + \alpha^{(t)} \mathbf{d}^{(t)}) - S_\lambda(\hat{\beta}^{(t)}) \leq \alpha^{(t)} \sigma \Delta^{(t)},$$

where  $0 < \delta < 1$ ,  $0 < \sigma < 1$  and  $\alpha_0 > 0$ , and  $\Delta^{(t)}$  is the improvement in the objective function  $S_\lambda(\cdot)$  when using a linear approximation for the log-likelihood, i.e.

$$\Delta^{(t)} = -\mathbf{d}^{(t)T} \nabla l(\hat{\beta}^{(t)}) + \lambda s(\text{df}_g) \|\hat{\beta}_g^{(t)} + \mathbf{d}_g^{(t)}\|_2 - \lambda s(\text{df}_g) \|\hat{\beta}_g^{(t)}\|_2.$$

Finally, we define

$$\hat{\beta}^{(t+1)} = \hat{\beta}^{(t)} + \alpha^{(t)} \mathbf{d}^{(t)}.$$

**Table 2.** Logistic group lasso algorithm using block co-ordinate gradient descent minimization

Step	Algorithm
1	Let $\beta \in \mathbb{R}^{p+1}$ be an initial parameter vector
2	For $g=0, \dots, G$ $H_{gg} \leftarrow h_g(\beta) I_{df_g}$ $\mathbf{d} \leftarrow \arg \min_{\mathbf{d}   \mathbf{d}_k=0, k \neq g} \{M_\lambda(\mathbf{d})\}$ if $\mathbf{d} \neq \mathbf{0}$ $\alpha \leftarrow \text{line search}$ $\beta \leftarrow \beta + \alpha \mathbf{d}$ end end
3	Repeat step 2 until some convergence criterion is met

The algorithm is outlined in Table 2. When minimizing  $M_\lambda^{(t)}(\cdot)$  with respect to a penalized group, we first must check whether the minimum is at a non-differentiable point as outlined above. For the (unpenalized) intercept this is not necessary and the solution can be directly computed:

$$d_0^{(t)} = -\frac{1}{h_0^{(t)}} \nabla l(\hat{\beta}^{(t)})_0.$$

For a general matrix  $H^{(t)}$  the minimization with respect to the  $g$ th parameter group depends on  $H^{(t)}$  only through the corresponding submatrix  $H_{gg}^{(t)}$ . To ensure a reasonable quadratic approximation in equation (2.3),  $H_{gg}^{(t)}$  is ideally chosen to be close to the corresponding submatrix of the Hessian of the log-likelihood function. Restricting ourselves to matrices of the form  $H_{gg}^{(t)} = h_g^{(t)} I_{df_g}$ , a possible choice is (Tseng and Yun, 2007)

$$h_g^{(t)} = -\max[\text{diag}\{-\nabla^2 l(\hat{\beta}^{(t)})_{gg}\}, c_*], \quad (2.4)$$

where  $c_* > 0$  is a lower bound to ensure convergence (see proposition 2). The matrix  $H^{(t)}$  does not necessarily have to be recomputed in each iteration. Under some mild conditions on  $H^{(t)}$  convergence of the algorithm is assured, as can be seen from Tseng and Yun (2007) and from the proof of proposition 2.

Standard choices for the tuning parameters are for example  $\alpha_0 = 1$ ,  $\delta = 0.5$  and  $\sigma = 0.1$  (Bertsekas, 2003; Tseng and Yun, 2007). Other definitions of  $\Delta^{(t)}$  as for example to include the quadratic part of the improvement are also possible. We refer the reader to Tseng and Yun (2007) for more details and proofs that  $\Delta^{(t)} < 0$  for  $\mathbf{d}^{(t)} \neq \mathbf{0}$  and that the line search can always be performed.

*Proposition 2.* If  $H_{gg}^{(t)}$  is chosen according to equation (2.4), then every limit point of the sequence  $\{\hat{\beta}^{(t)}\}_{t \geq 0}$  is a minimum point of  $S_\lambda(\cdot)$ .

*Remark 1.* When cycling through the co-ordinate blocks, we could restrict ourselves to the current active set and visit the remaining blocks for example every 10th iteration to update the active set. This is especially useful for very high dimensional settings and it easily allows for  $p \approx 10^4$ – $10^5$ . For the high dimensional example in Section 2.3, this modification decreases the computational times by about 40% of what is reported in Fig. 2 there. Moreover, it is also possible to update the co-ordinate blocks in a non-cyclic manner or all at the same time, which would allow for a parallelizable approach with the convergence result still holding.

*Remark 2.* The block co-ordinate gradient descent algorithm BCGD can also be applied to the group lasso in other models. For example, any generalized linear model where the response  $y$  has a distribution from the exponential family falls into this class. This is available in our R-package `grplasso`.

A related algorithm is found in Krishnapuram *et al.* (2005), where a global upper bound on the Hessian is used to solve the lasso problem for multinomial logistic regression. This approach can also be used with the group lasso penalty, resulting in a closed form solution for a block update. However, the upper bound is not sufficiently tight for moderate and small values of  $\lambda$ , which leads to too slow convergence in general. Genkin *et al.* (2007) overcame this problem by working with an updated local bound on the second derivative and by restricting the change in the current parameter to a local neighbourhood.

For linear models, the least angle regression algorithm lars (Efron *et al.*, 2004; Osborne *et al.*, 2000) is very efficient for computing the path of lasso solutions  $\{\hat{\beta}_\lambda\}_{\lambda \geq 0}$ . For logistic regression, approximate path following algorithms have been proposed (Rosset, 2005; Zhao and Yu, 2007; Park and Hastie, 2007). But, with the group lasso penalty, some of them are not applicable (Rosset, 2005) or do not necessarily converge to a minimum point of  $S_\lambda(\cdot)$  (Zhao and Yu, 2007), and none seem to be computationally faster than working iteratively on a fixed grid of penalty parameters  $\lambda$ . This has been observed as well by Genkin *et al.* (2007) for logistic regression with the lasso in large-scale applications.

To calculate the solutions  $\hat{\beta}_\lambda$  on a grid of the penalty parameter  $0 \leq \lambda_K < \dots < \lambda_1 \leq \lambda_{\max}$  we can for example start at

$$\lambda_{\max} = \max_{g \in \{1, \dots, G\}} \left\{ \frac{1}{s(\text{df}_g)} \|X_g^T(\mathbf{y} - \bar{\mathbf{y}})\|_2 \right\},$$

where only the intercept is in the model. We then use  $\hat{\beta}_{\lambda_K}$  as a starting value for  $\hat{\beta}_{\lambda_{K+1}}$  and proceed iteratively until  $\hat{\beta}_{\lambda_K}$  with  $\lambda_K$  equal or close to 0. Instead of updating the approximation of the Hessian  $H^{(t)}$  in each iteration, we can use a constant matrix based on the previous parameter estimates  $\hat{\beta}_{\lambda_K}$  to save computing time, i.e.

$$H_{gg}^{(t)} = h_g(\hat{\beta}_{\lambda_K}) I_{\text{df}_g},$$

for the estimation of  $\hat{\beta}_{\lambda_{K+1}}$ . Some cross-validation can then be used for choosing the parameter  $\lambda$ . Most often, we aim for minimal test sample negative log-likelihood score.

### 2.3. Comparison with other algorithms

In this subsection we compare the BCGD algorithm with the blockwise sparse regression (BSR) algorithm of Kim *et al.* (2006). After an earlier version of this manuscript, Park and Hastie (2006) (algorithm PH) also applied their methodology of Park and Hastie (2007) to group lasso models which we also include in our comparison.

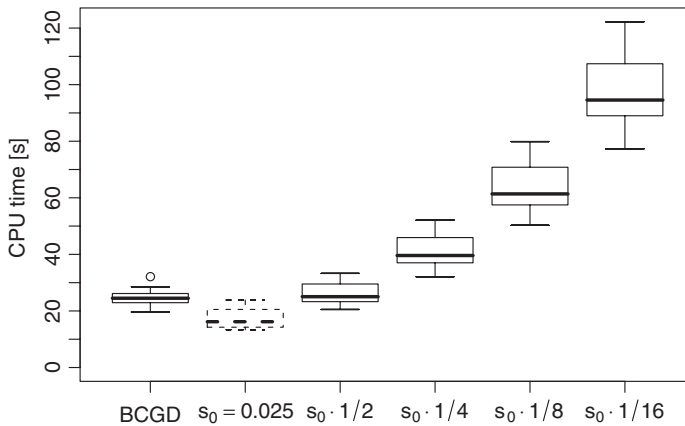
We emphasize that BSR is a method which requires the specification of an algorithmic tuning parameter, which is denoted by  $s$ . It is shown in Kim *et al.* (2006) that numerical convergence of BSR only holds if  $s$  is chosen sufficiently small (depending on the unknown Lipschitz constant of the gradient). Moreover, a small parameter  $s$  slows down the computational speed of BSR, and vice versa for a large  $s$ . Thus, we are in a situation of trading off numerical convergence against computational speed. Our BCGD method does not require the specification of an algorithmic tuning parameter to ensure convergence, and we view this as a very substantial advantage for practical use.

For comparing the various algorithms, we use a random-design matrix where the predictors are simulated according to a centred multivariate normal distribution with covariance matrix  $\Sigma_{i,j} = \rho^{|i-j|}$ . If not stated otherwise,  $\rho = 0.5$  is used. For the penalty parameter  $\lambda$  multiplicative grids between  $\lambda_{\max}$  and  $\lambda_{\max}/100$  are used.

For BCGD we use the R package `grplasso` and for BSR our own implementation in R. As BSR works with a constraint instead of a penalty, we use the result of BCGD as constraint value. We use an equivalent stopping criterion as in the package `grplasso`, i.e. the relative function improvement and the relative change in the parameter vector must be sufficiently small. Although this slowed down the algorithms, it is necessary to identify the correct active set of the solution. For both algorithms we make use of the preceding solution of the path as starting value for the next grid point. For BCGD we update the Hessian at each fifth grid point and we use an ‘ordinary’ cycling through the co-ordinate blocks. For the path following algorithm PH of Park and Hastie (2006) we use the corresponding MATLAB implementation that is available at <http://www.stanford.edu/~mypark/glasso.htm>. As recommended, the step length on the  $\lambda$ -scale is chosen adaptively. However, we could run PH with reasonable computing time on very small data sets only.

One of them is motivated by the user guide of PH. It consists of  $n = 200$  observations of  $G = 3$  groups each having  $\text{df} = 3$ , i.e.  $p = 10$  (with intercept). For the design matrix we use  $\rho = 0$  and the whole parameter vector is set to zero, i.e. there is no signal. 20 grid points are used for  $\lambda$ . The corresponding central processor unit times (in seconds) based on 20 simulation runs are 0.093 (0.01), 0.041 (0.0054) and 5.96 (1.23) for BCGD, BSR and PH respectively. Standard deviations are given in parentheses. We used the tuning parameter  $s = 0.01$  for BSR. Already for such a simple, low dimensional problem, BCGD and BSR were substantially faster than PH. As mentioned above, we could not run PH for larger problems (this is probably because of implementation, but we also think that an optimized implementation of PH, involving potentially large active sets, would be slower than BCGD or BSR).

As a second example, we use a higher dimensional setting with  $n = 100$  and  $G = 250$  groups each having  $\text{df} = 4$  ( $p = 1001$ ). The first 10 groups are active with coefficient 0.2, resulting in a Bayes risk of approximately 0.2. The computing times based on 20 simulation runs are depicted in Fig. 2, where we have used 100 grid points for  $\lambda$ . The boxplot for  $s_0 = 0.025$  is shown in outline because the probability for numerical convergence was only 20%. BSR with  $s$  suitably chosen is not faster in this example. The success for numerical convergence depends heavily on the choice



**Fig. 2.** Central processor unit times for BCGD (left-hand side) and for BSR with various values of the parameter  $s$ : see the text for more details



**Table 3.** Central processor unit times on the splice site data set for BCGD and for BSR with various values for the parameter  $s^\dagger$ 

Time (s) for BCGD	Times (s) for BSR and the following values of $s$ :			
	$s = 5 \times 10^{-4}$	$s = 2.5 \times 10^{-4}$	$s = 1.25 \times 10^{-4}$	$s = 6.125 \times 10^{-5}$
948	2737	4273	6688	10581

$^\dagger$ The algorithm did not converge for larger values of  $s$ .

of  $s$  and additional time is needed to find an optimal value for  $s$ . For some single  $\lambda$ -values, BSR sometimes turned out to be faster, but the difference between the computing times when calculating the whole path is much smaller owing to good starting values and the fact that BSR slows down for small values of  $\lambda$ .

We also applied BSR to the splice site data set in Section 5. The running times are reported in Table 3. We summarize that BCGD is often almost as fast as or even faster than (as for the real splice site data set) BSR with the optimal algorithmic tuning parameter  $s$ . This tuning parameter varies very much from problem to problem, and it is highly unrealistic to have reasonable *a priori* knowledge about a good parameter. Thus, the user needs to do some trial and error first, which can be very unpleasant. In contrast, BCGD runs fully automatically and is proved to converge, as described in Section 2.2.2.

Because of implementational issues it can be difficult to compare different algorithms. But the fact that co-ordinatewise approaches for sparse models are efficient for high dimensional data has also been noticed by Genkin *et al.* (2007) or Balakrishnan and Madigan (2006). They have successfully applied related algorithms for the lasso even when the number of variables was in the hundreds of thousands. For the co-ordinatewise approaches in general, already after a few sweeps through all variables both the objective function and the number of selected variables is close to the optimal solution.

#### 2.4. Consistency

A reasonable choice of the tuning parameter  $\lambda$  will depend on the sample size  $n$ , as well as on the number of groups  $G$ , and the degrees of freedom within each group. Assuming that the degrees of freedom per group are kept fixed, the smoothing parameter  $\lambda$  can be taken of order  $\log(G)$ . Then the group lasso can be shown to be globally consistent under some further regularity and sparseness conditions. This section gives more details on this asymptotic result.

Let us consider the data (before rescaling)  $(\mathbf{x}_i, y_i)$  as independent copies of the population variable  $(\mathbf{x}, y)$ . The negative log-likelihood function is used as the loss function, which we denote for easier notation by

$$\gamma_\beta(\mathbf{x}, y) = -(y\eta_\beta(\mathbf{x}) - \log[1 + \exp\{\eta_\beta(\mathbf{x})\}]).$$

The theoretical risk is defined as

$$R(\beta) = \mathbb{E}[\gamma_\beta(\mathbf{x}, y)],$$

and the empirical counterpart as

$$R_n(\beta) = \frac{1}{n} \sum_{i=1}^n \gamma_\beta(\mathbf{x}_i, y_i).$$

With this notation, the logistic group lasso estimator  $\hat{\beta}_\lambda$  is the minimizer of

$$\frac{S_\lambda(\beta)}{n} = R_n(\beta) + \frac{\lambda}{n} \sum_{g=1}^G s(\text{df}_g) \|\beta_g\|_2.$$

Let us consider a minimizer

$$\beta^0 \in \arg \min_{\beta} \{R(\beta)\}.$$

If the model is well specified it holds that

$$\mathbb{E}[y|\mathbf{x}] = p_{\beta^0}(\mathbf{x}).$$

There are various ways to measure the quality of the estimation procedure. We shall use the global measure

$$d^2(\eta_{\hat{\beta}_\lambda}, \eta_{\beta^0}) = \mathbb{E}[|\eta_{\hat{\beta}_\lambda}(\mathbf{x}) - \eta_{\beta^0}(\mathbf{x})|^2].$$

The following assumptions are made.

- (a) We shall suppose that for some constant  $0 < \varepsilon \leq \frac{1}{2}$

$$\varepsilon \leq p_{\beta^0}(\mathbf{x}) \leq 1 - \varepsilon$$

for all  $\mathbf{x}$ .

- (b) We shall require that the matrix

$$\Sigma = \mathbb{E}[\mathbf{x}\mathbf{x}^T]$$

is non-singular. We denote the smallest eigenvalue of  $\Sigma$  by  $\nu^2$ .

- (c) Let  $\mathbf{x}_g$  denote the  $g$ th predictor in  $\mathbf{x}$ . We normalize  $\mathbf{x}_g$  such that it has identity inner product matrix  $\mathbb{E}[\mathbf{x}_g \mathbf{x}_g^T] = I_{\text{df}_g}$ . With this normalization, we assume in addition that, for some constant  $L_n$ ,

$$\max_{\mathbf{x}} \max_g (\mathbf{x}_g^T \mathbf{x}_g) \leq n L_n^2.$$

The smallest possible order for  $L_n^2$  is  $L_n^2 = O(1/n)$ , since we use the normalization  $\mathbb{E}[\mathbf{x}_g \mathbf{x}_g^T] = I_{\text{df}_g}$ . For categorical predictors,  $L_n^2 = O(1/n)$  corresponds to the balanced case where in each category the probability of finding an individual in that category is bounded away from 0 and 1.

One can then show consistency in the following sense. Let  $\beta_g^0$  denote the elements in the vector  $\beta^0$  corresponding to the  $g$ th group. Let  $N_0$  be the number of non-zero group effects, i.e. the number of vectors  $\beta_g^0$  satisfying  $\|\beta_g^0\|_2 \neq 0$ . Then there exist universal constants  $C_1, C_2, C_3$  and  $C_4$ , and constants  $c_1$  and  $c_2$  depending on  $\varepsilon, \nu$  and  $\max_g(\text{df}_g)$ , such that whenever

- (d)  $C_1(1 + N_0^2)L_n^2 \log(G) \leq c_1$  and  $C_1 \log(G) \leq \lambda \leq c_1/(1 + N_0^2)L_n^2$

the following probability inequality holds:

$$\mathbb{P}\left\{d^2(\eta_{\hat{\beta}_\lambda}, \eta_{\beta^0}) \geq c_2 \frac{(1 + N_0)\lambda}{n}\right\} \leq C_2 \left\{\log(n) \exp\left(-\frac{\lambda}{C_3}\right) + \exp\left(-\frac{1}{C_4 L_n^2}\right)\right\}.$$

This result follows from arguments that are similar to those which were used in van de Geer (2003) and Tarigan and van de Geer (2006). An outline of the proof is given in Appendix A.

For the asymptotic implications, let us assume that  $\varepsilon, \nu$  and  $\max_g(\text{df}_g)$  are kept fixed as

$n \rightarrow \infty$  and that  $G \gg \log(n)$ . Take  $\lambda \asymp \log(G)$ , i.e.  $\lambda$  is of the order  $\log(G)$ . When for example the number of non-zero group effects  $N_0$  satisfies  $N_0 = O(1)$ , and when  $L_n^2 = O\{1/\log(G)\}$ , then we find the, almost parametric, rate

$$d^2(\eta_{\hat{\beta}_\lambda}, \eta_{\beta^0}) = O_P\{\log(G)/n\}.$$

With  $L_n^2 = O(1/n)$ , the maximal rate of growth for  $N_0$  is  $N_0 = O[\sqrt{\{n/\log(G)\}}]$ , and when  $N_0$  is exactly of this order we arrive at the rate

$$d^2(\eta_{\hat{\beta}_\lambda}, \eta_{\beta^0}) = O_P[\sqrt{\{\log(G)/n\}}].$$

*Remark 3.* We may improve the result by replacing  $N_0$  by the number of non-zero coefficients of ‘the best’ approximation of  $\eta_{\beta^0}$ , which is the approximation that balances estimation error and approximation error.

*Remark 4.* A similar consistency result can be obtained for the group lasso for Gaussian regression.

### 3. Logistic group lasso–ridge hybrid

#### 3.1. General case

As can be observed in the simulation study in Yuan and Lin (2006), the models that are selected by the group lasso are large compared with the underlying true models. For the ordinary lasso, smaller models with good prediction performance can be obtained by using the lasso with relaxation (Meinshausen, 2007). This idea can also be incorporated in the (logistic) group lasso approach and our proposal will also allow us to fit hierarchical models.

Denote by  $\hat{\mathcal{I}}_\lambda \subseteq \{0, \dots, G\}$  the index set of predictors that are selected by the group lasso with penalty parameter  $\lambda$  and by  $\hat{\mathcal{M}}_\lambda = \{\beta \in \mathbb{R}^{p+1} \mid \beta_g = \mathbf{0} \text{ for } g \notin \hat{\mathcal{I}}_\lambda\}$  the set of possible parameter vectors of the corresponding submodel. The group lasso–ridge hybrid estimator is defined as

$$\hat{\beta}_{\lambda, \kappa} = \arg \min_{\beta \in \hat{\mathcal{M}}_\lambda} \left\{ -l(\beta) + \kappa \sum_{g=1}^G \frac{s(\text{df}_g)}{\sqrt{\text{df}_g}} \|\beta_g\|_2^2 \right\} \quad (3.1)$$

for  $\lambda, \kappa \geq 0$ . The penalty in equation (3.1) is rescaled with  $1/\sqrt{\text{df}_g}$  to ensure that it is of the same order as the group lasso penalty. The special case  $\kappa = 0$  is analogous to the lars–ordinary least squares hybrid in Efron *et al.* (2004) and is denoted as the group lasso–maximum likelihood estimator (MLE) hybrid. In this case, we only need the group lasso to select a candidate model  $\hat{\mathcal{M}}_\lambda$ . The parameters are estimated with the (unpenalized) MLE. Optimization problem (3.1) can be solved with a Newton-type algorithm. For large-scale applications co-ordinatewise approaches as used in Genkin *et al.* (2007) may be more appropriate. The reason why we choose a ridge-type penalty follows in the next subsection.

#### 3.2. Restriction to hierarchical models

When working with interactions between predictors (e.g. factors), the group lasso solutions are not necessarily hierarchical. An interaction may be present even though (some) corresponding main effects are missing. In most applications hierarchical models are preferred because of their interpretability. The above two-stage procedure (3.1) can be used to produce hierarchical models by expanding the model class  $\hat{\mathcal{M}}_\lambda$  to  $\hat{\mathcal{M}}_\lambda^{\text{hier}}$ , where  $\hat{\mathcal{M}}_\lambda^{\text{hier}}$  is the hierarchical model class that is

induced by  $\hat{\mathcal{M}}_\lambda$ :  $\hat{\beta}_{\lambda,\kappa}^{\text{hier}}$  is then defined as in estimator (3.1), but with the minimum taken over  $\hat{\mathcal{M}}_\lambda^{\text{hier}}$ .

Instead of using a ridge-type penalty, we could have also used again the group lasso penalty and proceed exactly as in Meinshausen (2007), using a ‘relaxed’ penalty  $\kappa \leq \lambda$  in the second stage. Although this works well for the general (non-hierarchical) case, there are problems if we restrict ourselves to hierarchical models. Even if we choose  $\kappa \leq \lambda$  the solutions may not be hierarchical owing to the expansion of  $\hat{\mathcal{M}}_\lambda$  to  $\hat{\mathcal{M}}_\lambda^{\text{hier}}$ . In other words: some variable selection may happen in addition in the second stage. Using a ridge-type penalty, we prevent any further model selection and just do shrinkage.

#### 4. Simulation

We use a simulation scheme that is similar to that of Yuan and Lin (2006) but with larger models. In each simulation run we first sample  $n_{\text{train}}$  instances of a nine-dimensional multivariate normal distribution  $(T_1, \dots, T_9)^T$  with mean vector  $\mathbf{0}$  and covariance matrix  $\Sigma_{i,j} = \rho^{|i-j|}$ . Each component  $T_k, k = 1, \dots, 9$ , is subsequently transformed into a four-valued categorical random variable by using the quartiles of the standard normal distribution. For the main effects, this results in (non-orthogonalized) predictors  $\mathbf{x}_i = (\mathbf{x}_{i,1}^T, \dots, \mathbf{x}_{i,9}^T)^T \in \mathbb{R}^{9 \times 3}, i = 1, \dots, n_{\text{train}}$ . We use the sum constraint as encoding scheme for the dummy variables, i.e. the coefficients must add up to 0. The entire predictor space has dimension  $4^9 = 262\,144$ . The corresponding responses  $y_i$  are simulated according to a Bernoulli distribution with model-based probabilities.

The parameter vector  $\beta_g$  of a predictor with  $\text{df}_g$  degrees of freedom is set up as follows to conform to the encoding scheme. We simulate  $\text{df}_g + 1$  independent standard normal distributions resulting in  $\tilde{\beta}_{g,1}, \dots, \tilde{\beta}_{g,\text{df}_g+1}$  and define

$$\beta_{g,j} = \tilde{\beta}_{g,j} - \frac{1}{\text{df}_g + 1} \sum_{k=1}^{\text{df}_g+1} \tilde{\beta}_{g,k}$$

for  $j \in \{1, \dots, \text{df}_g\}$ . The intercept is set to 0. The whole parameter vector  $\beta$  is finally rescaled to adjust the empirical Bayes risk  $r$  at the desired level, where

$$r = \frac{1}{n} \sum_{i=1}^n \min\{p_\beta(\mathbf{x}_i), 1 - p_\beta(\mathbf{x}_i)\}$$

for some large  $n$ . For all simulation runs for a given setting of  $\rho$  and  $r$ , the same parameter vector is reused.

The four different cases studied are as follows.

- The main effects and the two-way interaction between the first two factors  $\mathbf{x}_{i,1}$  and  $\mathbf{x}_{i,2}$  build the true model, which has a total of four terms or 16 parameters.  $n_{\text{train}} = 500$  observations are used in each simulation run.
- The underlying true model consists of all main effects and two-way interactions between the first five factors  $\mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,5}$ , resulting in 16 terms or 106 parameters.  $n_{\text{train}} = 500$ .
- Case (c) is as case (b) but with  $n_{\text{train}} = 1000$ .
- All main effects and two-way interactions between  $\mathbf{x}_{i,k}$  and  $\mathbf{x}_{i,l}, |k-l| = 1$ , are active. In addition the two-way interactions between  $\mathbf{x}_{i,1}, \mathbf{x}_{i,5}$  and  $\mathbf{x}_{i,3}, \mathbf{x}_{i,9}$  are present. This makes a total of 20 terms or 118 parameters.  $n_{\text{train}} = 1000$ .

For estimation, the candidate models that were used for the logistic group lasso and its variants consist always of all main effects and all two-way interactions from the nine factors, which

**Table 4.** Average test set negative log-likelihood and average number of selected terms based on 100 simulation runs†

Case	$\rho$	$r$	Test set negative log-likelihood			Number of terms		
			Group lasso	Group lasso–ridge hybrid	Group lasso–MLE hybrid	Group lasso	Group lasso–ridge hybrid	Group lasso–MLE hybrid
(a)	0.00	0.15	185.57	185.76	212.74	20.73	4.11	3.96
			(10.85)	(26.42)	(37.28)	(8.02)	(0.40)	(0.49)
		0.25	273.57	269.35	278.76	16.39	4.30	4.01
			(8.28)	(19.63)	(29.57)	(6.74)	(0.83)	(1.01)
	0.20	0.15	185.85	182.31	207.24	20.08	4.28	3.90
			(11.62)	(20.82)	(37.63)	(7.15)	(1.05)	(0.56)
		0.25	274.92	269.34	273.89	15.21	4.17	4.03
			(8.48)	(17.63)	(22.11)	(6.22)	(0.53)	(0.58)
	0.50	0.15	194.67	191.90	207.16	18.73	4.17	3.79
			(12.49)	(24.68)	(28.25)	(7.61)	(0.73)	(0.50)
		0.25	279.59	275.35	283.90	14.34	4.26	3.83
			(9.37)	(13.34)	(16.78)	(6.38)	(0.80)	(0.88)
(b)	0.00	0.15	233.84	229.69	291.39	35.90	16.86	8.93
			(12.76)	(15.15)	(27.06)	(3.98)	(3.37)	(3.22)
		0.25	300.08	306.07	325.53	30.95	15.78	6.02
			(9.50)	(10.92)	(18.61)	(5.77)	(6.85)	(3.39)
	0.20	0.15	235.29	232.13	283.32	36.20	16.65	9.19
			(12.71)	(16.29)	(21.19)	(3.94)	(3.60)	(2.83)
		0.25	299.65	303.82	322.22	30.93	16.37	6.85
			(8.94)	(11.78)	(11.53)	(5.27)	(7.75)	(3.36)
	0.50	0.15	232.63	230.48	285.17	35.11	15.67	8.47
			(11.35)	(15.07)	(29.88)	(3.73)	(4.02)	(2.82)
		0.25	296.56	298.68	321.80	29.62	14.11	6.61
			(7.78)	(11.09)	(23.76)	(5.88)	(5.55)	(3.14)
(c)	0.00	0.15	408.63	385.03	423.05	40.95	15.78	14.72
			(16.55)	(19.17)	(34.11)	(2.52)	(1.06)	(1.13)
		0.25	569.56	557.99	581.94	36.34	15.41	13.63
			(14.13)	(17.31)	(24.89)	(3.83)	(2.27)	(1.85)
	0.20	0.15	408.27	385.64	424.42	40.54	15.89	14.49
			(18.53)	(20.43)	(30.87)	(2.79)	(1.56)	(1.18)
		0.25	566.46	556.70	584.23	36.65	15.92	13.25
			(15.17)	(19.44)	(28.48)	(4.11)	(2.75)	(2.01)
	0.50	0.15	397.35	376.20	421.71	39.90	15.74	14.29
			(19.83)	(24.79)	(41.32)	(2.85)	(1.49)	(1.16)
		0.25	559.28	550.82	579.71	35.86	15.85	13.09
			(15.57)	(18.55)	(27.67)	(4.00)	(2.77)	(1.98)
(d)	0.00	0.15	419.61	394.37	445.84	42.74	20.17	19.01
			(18.02)	(21.81)	(39.15)	(2.19)	(0.82)	(1.18)
		0.25	574.58	566.77	600.09	39.43	19.59	17.36
			(13.78)	(14.90)	(20.92)	(3.07)	(2.32)	(2.31)
	0.20	0.15	418.67	394.51	447.09	42.46	20.26	18.94
			(17.91)	(20.54)	(34.93)	(2.26)	(0.79)	(1.31)
		0.25	574.10	565.83	601.39	38.88	19.91	16.58
			(14.50)	(17.68)	(27.49)	(3.47)	(2.41)	(2.48)
	0.50	0.15	416.96	394.84	448.87	42.50	20.28	18.78
			(18.02)	(21.28)	(38.49)	(2.60)	(1.29)	(1.49)
		0.25	568.63	562.49	599.59	38.65	20.08	15.78
			(15.09)	(17.46)	(29.26)	(3.38)	(2.39)	(3.28)

†Standard deviations are given in parentheses.

make a total of 46 terms or  $p = 352$  parameters. We use the restriction for hierarchical model fitting for the two-stage procedures.

In each simulation run, all models are fitted on a training data set of size  $n_{\text{train}}$ . A multiplicative grid of the form  $\{\lambda_{\text{max}}, 0.96\lambda_{\text{max}}, \dots, 0.96^{148}\lambda_{\text{max}}, 0\}$  is used for the penalty parameter  $\lambda$  of the logistic group lasso. For the logistic group lasso–ridge hybrid, we consider values  $\kappa \in \{1.5^{11}, 1.5^{10}, \dots, 1.5^{-5}, 0\}$ . The penalty parameters  $\lambda$  and  $\kappa$  are selected according to the (unpenalized) log-likelihood score on an independent validation set of size  $n_{\text{train}}/2$ . Finally, the models are evaluated on an additional test set of size  $n_{\text{train}}$ .

Table 4 reports the average test set negative log-likelihood and the average number of terms selected on the basis of 100 simulation runs for each setting. The corresponding standard deviations are given in parentheses. The group lasso produces the largest models followed by the group lasso–ridge hybrid and group lasso–MLE hybrid. Compared with the underlying true models, the group lasso seems to select unnecessarily large models with many noise variables resulting in a low true discovery rate (which is not shown), which is defined as the ratio of the number of correctly selected terms and the total number of selected terms. On the other side, the group lasso–MLE hybrid is very conservative in selecting terms, resulting in a large true discovery rate at the cost of a low true positive rate (which is not shown). The group lasso–ridge hybrid seems to be the best compromise.

The prediction performance measured in terms of the (unpenalized) log-likelihood score on the test set is in most cases best for the group lasso–ridge hybrid, followed by the group lasso and the group lasso–MLE hybrid. The group lasso–ridge hybrid seems to be able to benefit from the good prediction performance of the group lasso with the advantage of producing reasonably sized models.

## 5. Application to splice site detection

The prediction of short DNA motifs plays an important role in many areas of computational biology. Gene finding algorithms such as GENIE (Burge and Karlin, 1997) often rely on the prediction of splice sites. Splice sites are the regions between coding (exons) and non-coding (introns) DNA segments. The 5' end of an intron is called a donor splice site and the 3' end an acceptor splice site. A donor site whose first two intron positions are the letters 'GT' is called canonical, whereas an acceptor site is called canonical if the corresponding intron ends with 'AG'. An overview of the splicing process and of some models that are used for detecting splice sites can be found in Burge (1998).

### 5.1. MEMset donor data set

The MEMset donor data set consists of a training set of 8415 true (encoded as  $Y = 1$ ) and 179 438 false (encoded as  $Y = 0$ ) human donor sites. An additional test set contains 4208 true and 89 717 false donor sites. A sequence of a real splice site consists of the last three bases of the exon and the first six bases of the intron. False splice sites are sequences on the DNA which match the consensus sequence at positions 4 and 5. Removing the consensus GT results in a sequence length of 7 with values in  $\{A, C, G, T\}^7$ : thus, the predictor variables are seven factors, each having four levels. The data are available at <http://genes.mit.edu/burgelab/maxent/ssdata/>. A more detailed description can be found in Yeo and Burge (2004).

The original training data set is used to build a smaller balanced training data set (5610 true and 5610 false donor sites) and an unbalanced validation set (2805 true and 59 804 false donor sites). All sites are chosen randomly without replacement such that the two sets are disjoint.

The additional test set remains unchanged. Note that the ratios of true to false sites are equal for the validation and the test set.

### 5.2. Procedure

All models are fitted on the balanced training data set. As the ratio of true to false splice sites strongly differs from the training to the validation and the test set, the intercept is corrected as follows (King and Zeng, 2001):

$$\hat{\beta}_0^{\text{corr}} = \hat{\beta}_0 - \log\left(\frac{\bar{y}}{1 - \bar{y}}\right) + \log\left(\frac{\pi}{1 - \pi}\right),$$

where  $\pi$  is the proportion of true sites in the validation set.

Penalty parameters  $\lambda$  and  $\kappa$  are selected according to the (unpenalized) log-likelihood score on the validation set by using the corrected intercept estimate.

For a threshold  $\tau \in (0, 1)$  we assign observation  $i$  to class 1 if  $p_{\hat{\beta}}(\mathbf{x}_i) > \tau$  and to class 0 otherwise. Note that the class assignment can also be constructed without intercept correction by using a different threshold.

The correlation coefficient  $\rho_{\tau}$  corresponding to a threshold  $\tau$  is defined as the Pearson correlation between the binary random variable of the true class membership and the binary random variable of the predicted class membership. In Yeo and Burge (2004) the maximal correlation coefficient

$$\rho_{\max} = \max\{\rho_{\tau} | \tau \in (0, 1)\}$$

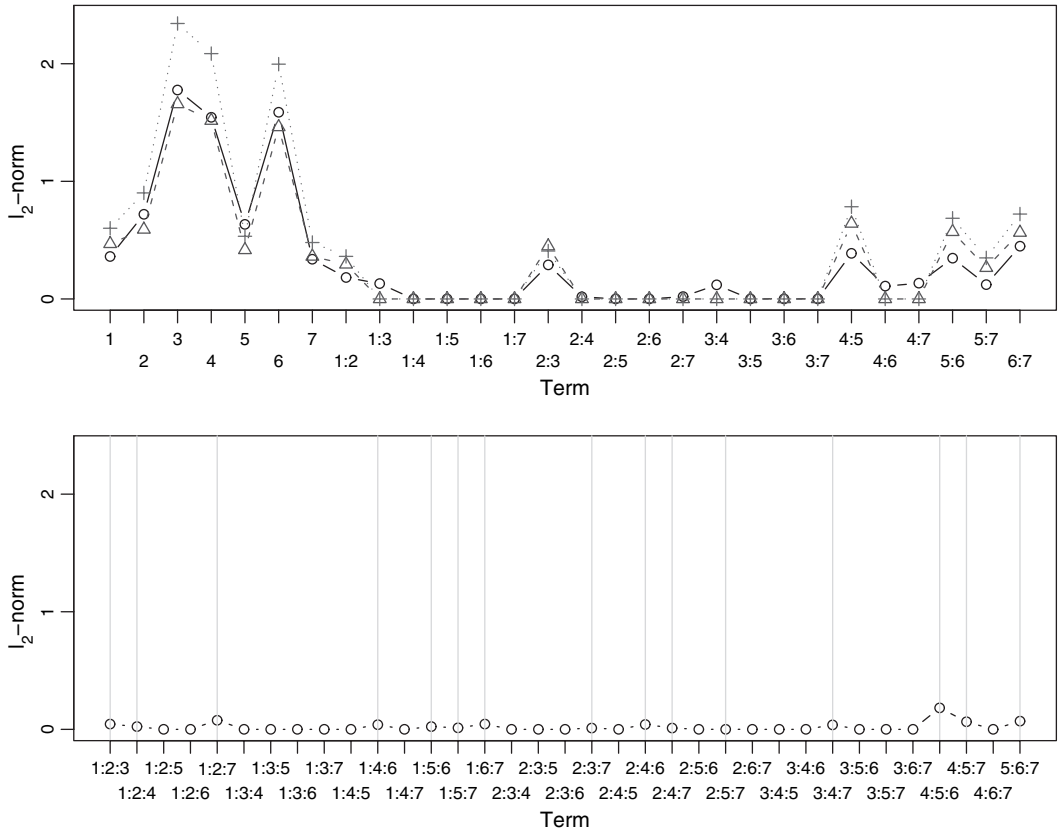
is used as a goodness-of-fit statistic on the test set.

The candidate model that was used for the logistic group lasso consists of all three-way and lower order interactions involving 64 terms or  $p = 1156$  parameters. In addition to the standard logistic group lasso estimator, the hierarchical group lasso–ridge hybrid and group lasso–MLE hybrid estimators are considered.

### 5.3. Results

The best model with respect to the log-likelihood score on the validation set is the group lasso estimator. It is followed by the group lasso–ridge hybrid and the group lasso–MLE hybrid. The corresponding values of  $\rho_{\max}$  on the test set are 0.6593, 0.6569 and 0.6541 respectively. They are all competitive with the results from Yeo and Burge (2004) whose best  $\rho_{\max}$  equals 0.6589. Whereas the group lasso solution has some active three-way interactions, the group lasso–ridge hybrid and the group lasso–MLE hybrid contain only two-way interactions. Fig. 3 shows the  $l_2$ -norm of each parameter group for the three estimators. The three-way interactions of the group lasso solution seem to be very weak. Considering also the non-hierarchical models for the two-stage procedures yields the same selected terms. Decreasing the candidate model size to contain only two-way interactions gives similar results.

In summary, the prediction performance of the group lasso and its variants is competitive with the maximum entropy models that were used in Yeo and Burge (2004) which have been viewed as (among) the best for short motif modelling and splice site prediction. Advantages of the group lasso and variants thereof include selection of terms. In addition, other (possibly continuous) predictor variables as for example global sequence information could be naturally included in the group lasso approach to improve the rather low correlation coefficients (Yeo and Burge, 2004).



**Fig. 3.**  $l_2$ -norms  $\|\hat{\beta}_g\|_2$ ,  $g \in \{1, \dots, G\}$ , of the parameter groups with respect to the blockwise orthonormalized design matrix when using a candidate model with all three-way interactions ( $i:j:k$  denotes the three-way interaction between the  $i$ th,  $j$ th and  $k$ th sequence position; the same scheme applies to the two-way interactions and the main effects):  $\circ$ , group lasso;  $\triangle$ , group lasso–ridge hybrid;  $+$ , group lasso–MLE hybrid;  $|$ , active three-way interaction

## 6. Discussion

We study the group lasso for logistic regression. We present efficient algorithms, which are especially suitable for very high dimensional problems, for solving the corresponding convex optimization problem, which is inherently more difficult than  $l_1$ -penalized logistic regression. The algorithms rely on recent theory and developments for block co-ordinate and block co-ordinate gradient descent minimization (Tseng, 2001; Tseng and Yun, 2007). In contrast with the algorithm in Kim *et al.* (2006), our procedure is fully automatic and does not require the specification of an algorithmic tuning parameter to ensure convergence. Moreover, our algorithm is much faster than the recent proposal from Park and Hastie (2006). An implementation can be found in our R-package `grplasso`. Additionally, we present a statistical consistency theory for the setting where the predictor dimension is potentially much larger than the sample size but assuming that the true underlying logistic regression model is sparse. The algorithms with the supporting mathematical optimization theory as well as the statistical consistency theory also apply directly to the group lasso in other generalized linear models.

Furthermore, we propose the group lasso–ridge hybrid method, which often yields better pre-



dictions and better variable selection than the group lasso. In addition, our group lasso–ridge hybrid allows for hierarchical model fitting.

Finally, we apply the group lasso and its variants to short DNA motif modelling and splice site detection. Our general methodology performs very well in comparison with the maximum entropy method which is considered to be among the best for this task.

## Acknowledgements

We thank two referees and Sylvain Sardy for constructive comments.

## Appendix A

### A.1. Proof of lemma 1

We shall first eliminate the intercept. Let  $\beta_1, \dots, \beta_G$  be fixed. To obtain the estimate for the intercept we must minimize a function of the form

$$g(\beta_0) = - \sum_{i=1}^n [y_i(\beta_0 + c_i) - \log\{1 + \exp(\beta_0 + c_i)\}]$$

with derivative

$$g'(\beta_0) = - \sum_{i=1}^n \left\{ y_i - \frac{\exp(\beta_0 + c_i)}{1 + \exp(\beta_0 + c_i)} \right\},$$

where  $c_i = \sum_{g=1}^G \mathbf{x}_{i,g}^T \beta_g$  is a constant. It holds that  $\lim_{\beta_0 \rightarrow \infty} \{g'(\beta_0)\} = n - \sum_{i=1}^n y_i > 0$  and  $\lim_{\beta_0 \rightarrow -\infty} \{g'(\beta_0)\} = -\sum_{i=1}^n y_i < 0$ . Furthermore  $g'(\cdot)$  is continuous and strictly increasing. Therefore there is a unique  $\beta_0^* \in \mathbb{R}$  such that  $g'(\beta_0^*) = 0$ . By the implicit function theorem the corresponding function  $\beta_0^*(\beta_1, \dots, \beta_G)$  is continuously differentiable. By replacing  $\beta_0$  in  $S_\lambda(\beta)$  by the function  $\beta_0^*(\beta_1, \dots, \beta_G)$  and using duality theory, we can rewrite equation (2.2) as an optimization problem under the constraint  $\sum_{g=1}^G \|\beta_g\|_2 \leq t$  for some  $t > 0$ . This is an optimization problem of a continuous function over a compact set; hence the minimum is attained.

### A.2. Proof of proposition 1

We first show that the groupwise minima of  $S_\lambda(\cdot)$  are attained. For  $g = 0$  this follows from the proof of lemma 1. The case  $g \geq 1$  corresponds to a minimization of a continuous function over a compact set; hence the minimum is attained. We now show that step 3 minimizes the convex function  $S_\lambda(\beta_g)$  for  $g \geq 1$ . Since  $S_\lambda(\beta_g)$  is not differentiable everywhere, we invoke subdifferential calculus (Bertsekas, 2003). The subdifferential of  $S_\lambda(\cdot)$  with respect to  $\beta_g$  is the set

$$\begin{aligned} \partial S_\lambda(\beta_g) &= \{-X_g^T(\mathbf{y} - \mathbf{p}_\beta) + \lambda \mathbf{e} \mid \mathbf{e} \in E(\beta_g)\}, \\ E(\beta_g) &= \left\{ \mathbf{e} \in \mathbb{R}^{\text{df}_g} : \mathbf{e} = s(\text{df}_g) \frac{\beta_g}{\|\beta_g\|_2} \text{ if } \beta_g \neq \mathbf{0} \text{ and } \|\mathbf{e}\|_2 \leq s(\text{df}_g) \text{ if } \beta_g = \mathbf{0} \right\}. \end{aligned}$$

The parameter vector  $\beta_g$  minimizes  $S_\lambda(\beta_g)$  if and only if  $\mathbf{0} \in \partial S_\lambda(\beta_g)$ , which is equivalent to the formulation of step 3. Furthermore conditions (A1), (B1)–(B3) and (C2) in Tseng (2001) hold. By lemma 3.1 and proposition 5.1 in Tseng (2001) every limit point of the sequence  $\{\hat{\beta}^{(t)}\}_{t \geq 0}$  is a stationary point of the convex function  $S_\lambda(\cdot)$ , and hence a minimum point.

### A.3. Proof of proposition 2

Proposition 2 directly follows from theorem 1(e) in section 4 of Tseng and Yun (2007). We must show that  $-H^{(t)}$  is bounded by above and away from zero. The Hessian of the negative log-likelihood function is

$$N = \sum_{i=1}^n p_\beta(\mathbf{x}_i) \{1 - p_\beta(\mathbf{x}_i)\} \mathbf{x}_i \mathbf{x}_i^T \leq \frac{1}{4} X^T X$$

in the sense that  $N - \frac{1}{4} X^T X$  is negative semidefinite. For the block matrix  $N_{gg}$  corresponding to the  $g$ th predictor it follows from the blockwise orthonormalization that  $N_{gg} \leq (n/4) I_{\text{df}_g}$  and hence  $\max\{\text{diag}(N_{gg})\} \leq n/4$ . An upper bound on  $-H^{(i)}$  is therefore always guaranteed. The lower bound is enforced by the choice of  $H_{gg}^{(i)}$ . By the choice of the line search we ensure that  $\alpha^{(i)}$  is bounded by above and therefore theorem 1(e) in section 4 of Tseng and Yun (2007) can be applied.

#### A.4. Outline of the proof of the consistency result

The proof of the consistency result follows the arguments that were used in Tarigan and van de Geer (2006), who considered hinge loss instead of logistic loss, but, as they pointed out, a large part of their results can be easily extended because only the Lipschitz property of the loss is used there. Furthermore, under assumption (a) in Section 2.4, logistic loss has the usual ‘quadratic’ behaviour near its overall minimum. This means that it does not share the problem of unknown margin behaviour with hinge loss, i.e. the situation is in that respect simpler than in Tarigan and van de Geer (2006).

The group lasso reduces to the  $l_1$ -penalty (the usual lasso) when there is only 1 degree of freedom in each group. The extension of consistency results to more degrees of freedom is straightforward, provided that  $\max_g(\text{df}_g)$  does not depend on  $n$ . We furthermore note that the group lasso uses a normalization involving the design matrix of the observed predictors. For the consistency result, we need to prove that this empirical normalization is uniformly close to the theoretical normalization. This boils down to showing that empirical and theoretical eigenvalues of the design matrix per group cannot be too far from each other, uniformly over the groups. Here, we invoke that assumption (d) in Section 2.4 implies that  $L_n^2$  is no larger than  $c_1/C_1 \log(G)$ . We then apply assumption (c) in Bernstein’s inequality to bound the difference in eigenvalues.

A technical extension compared with Tarigan and van de Geer (2006) is that we do not assume an *a priori* bound on the functions  $\eta_\beta(\cdot)$ . This is now handled by using convexity arguments (similar to van de Geer (2003)), and again part of the assumption (d), namely that  $\lambda$  is smaller than  $c_1/(1 + N_0^2)L_n^2$ . This assumption ensures that for all  $n$ , with high probability, the difference between  $\eta_{\beta^0}$  and the estimated regression  $\eta_{\hat{\beta}_\lambda}$  is bounded by a constant that is independent of  $n$ . For similar results see van de Geer (2007).

## References

- Antoniadis, A. and Fan, J. (2001) Regularization of wavelet approximations (with discussion). *J. Am. Statist. Ass.*, **96**, 939–967.
- Bakin, S. (1999) Adaptive regression and model selection in data mining problems. *PhD Thesis*. Australian National University, Canberra.
- Balakrishnan, S. and Madigan, D. (2006) Algorithms for sparse linear classifiers in the massive data setting. Rutgers University, Piscataway. (Available from <http://www.stat.rutgers.edu/~madigan/PAPERS/>.)
- Bertsekas, D. P. (2003) *Nonlinear Programming*. Belmont: Athena Scientific.
- Burge, C. (1998) Modeling dependencies in pre-mrna splicing signals. In *Computational Methods in Molecular Biology* (eds S. Salzberg, D. Searls and S. Kasif), ch. 8, pp. 129–164. New York: Elsevier Science.
- Burge, C. and Karlin, S. (1997) Prediction of complete gene structures in human genomic DNA. *J. Molec. Biol.*, **268**, 78–94.
- Cai, T. T. (2001) Discussion of “Regularization of wavelet approximations” (by A. Antoniadis and J. Fan). *J. Am. Statist. Ass.*, **96**, 960–962.
- Efron, B., Hastie, T., Johnstone, I. and Tibshirani, R. (2004) Least angle regression. *Ann. Statist.*, **32**, 407–499.
- van de Geer, S. (2003) Adaptive quantile regression. In *Recent Advances and Trends in Nonparametric Statistics* (eds M. Akritas and D. Politis), pp. 235–250. New York: Elsevier.
- van de Geer, S. (2007) High-dimensional generalized linear models and the lasso. *Ann. Statist.*, to be published.
- Genkin, A., Lewis, D. D. and Madigan, D. (2007) Large-scale bayesian logistic regression for text categorization. *Technometrics*, **49**, 291–304.
- Kim, Y., Kim, J. and Kim, Y. (2006) Blockwise sparse regression. *Statist. Sin.*, **16**, 375–390.
- King, G. and Zeng, L. (2001) Logistic regression in rare events data. *Polit. Anal.*, **9**, 137–163.
- Krishnapuram, B., Carin, L., Figueiredo, M. A. and Hartemink, A. J. (2005) Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Trans. Pattern Anal. Mach. Intell.*, **27**, 957–968.
- Lokhorst, J. (1999) The lasso and generalised linear models. *Honors Project*. University of Adelaide, Adelaide.
- Meinshausen, N. (2007) Lasso with relaxation. *Computnl Statist. Data Anal.*, **52**, 374–393.
- Osborne, M., Presnell, B. and Turlach, B. (2000) A new approach to variable selection in least squares problems. *IMA J. Numer. Anal.*, **20**, 389–403.

- Park, M.-Y. and Hastie, T. (2006) Regularization path algorithms for detecting gene interactions. Stanford University, Stanford. (Available from <http://www-stat.stanford.edu/~hastie/pub.htm>.)
- Park, M. Y. and Hastie, T. (2007)  $L_1$ -regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, **69**, 659–677.
- Rosset, S. (2005) Following curved regularized optimization solution paths. In *Advances in Neural Information Processing Systems*, vol. 17 (eds L. K. Saul, Y. Weiss and L. Bottou), pp. 1153–1160. Cambridge: MIT Press.
- Roth, V. (2004) The generalized lasso. *IEEE Trans. Neur. Netwks*, **15**, 16–28.
- Shevade, S. and Keerthi, S. (2003) A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, **19**, 2246–2253.
- Tarigan, B. and van de Geer, S. (2006) Classifiers of support vector machine type with  $l_1$  complexity regularization. *Bernoulli*, **12**, 1045–1076.
- Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *J. R. Statist. Soc. B*, **58**, 267–288.
- Tibshirani, R. (1997) The lasso method for variable selection in the cox model. *Statist. Med.*, **16**, 385–395.
- Tseng, P. (2001) Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optimizn Theory Appl.*, **109**, 475–494.
- Tseng, P. and Yun, S. (2007) A coordinate gradient descent method for nonsmooth separable minimization. *Math. Programmng B*, to be published.
- Yeo, G. W. and Burge, C. B. (2004) Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals. *J. Computnl Biol.*, **11**, 475–494.
- Yuan, M. and Lin, Y. (2006) Model selection and estimation in regression with grouped variables. *J. R. Statist. Soc. B*, **68**, 49–67.
- Zhao, P. and Yu, B. (2007) Stagewise lasso. *J. Mach. Learn. Res.*, to be published.