Estimating sparse models from multivariate discrete data via transformed Lasso

Teemu Roos
Helsinki Institute for Information Technology HIIT
Univ. of Helsinki & Helsinki Univ. of Technology, Finland
Email: teemu.roos@hiit.fi

Bin Yu
Department of Statistics
University of California, Berkeley, USA
Email: binyu@stat.berkeley.edu

***Abstract*—The type of ℓ_1 norm regularization used in Lasso and related methods typically yields sparse parameter estimates where most of the estimates are equal to zero. We study a class of estimators obtained by applying a linear transformation on the parameter vector before evaluating the ℓ_1 norm. The resulting "transformed Lasso" yields estimates that are "smooth" in a way that depends on the applied transformation. The optimization problem is convex and can be solved efficiently using existing tools. We present two examples: the Haar transform which corresponds to variable length Markov chain (context-tree) models, and the Walsh-Hadamard transform which corresponds to linear combinations of XOR (parity) functions of binary input features.**

I. Introduction

In situations where the number of potentially relevant features is large relative to the sample size ("large p, small n"), leveraging a suitable bias is crucial for both model selection and prediction accuracy. Usually this is explained in terms of Occam's razor or other principles favoring *simple* models over *complex* ones. Naturally, in order to be concrete, we need to be specific about what we mean by simplicity. For models that can be expressed as a vector (or matrix) of real values, the number of non-zero coefficients is one popular measure of simplicity, used widely as a basis of various information criteria like the Akaike information criterion (AIC), and the Bayesian information criterion (BIC). One of the main drawbacks of this measure is that the resulting minimization problems tend to be hard: we cannot find the optimal model without exhaustive search.

Replacing the number of non-zero coefficients by a convex surrogate function greatly simplifies the optimization problem in many cases. In particular, using the sum of the absolute values of the coefficients, i.e., the ℓ_1 norm of the coefficient vector, has lead to the methods of basis pursuit [1] and Lasso [2], for which fast algorithms exist, see e.g. [3], [4]. The Lasso and its relatives have turned out to be hugely successful in combination with a wide range of statistical models, and in many different applications, see e.g. the review [5].

However, in some cases our preferred way to measure simplicity is based on other factors in addition, or instead of, the number of non-zero coefficients or the ℓ_1 norm. It is not clear how such problems can be made amenable to the Lasso approach. In this paper, we consider an approach based on applying a linear transformation to the parameter vector such that sparsity in the sense of few non-zero coefficients in the *transformed* parameter vector coincides with out conception of simplicity. We present two examples on applying the idea to learning sparse models from discrete multivariate data. In the first example, we establish a connection between Lasso with a Haar transformation applied on the parameters and learning variable length Markov chains (VLMCs), or context tree models. In the second example, we show that using the so called Walsh-Hadamard transformation corresponds to learning linear combinations of XOR (parity) functions of binary input features.

II. Model Formulation

Consider a data-set $\{(\mathbf{x}^{(i)}, y^{(i)})\}, 1 \leq i \leq n$, where $\mathbf{x}^i = (x_1^{(i)}, \ldots, x_k^{(i)})'$ is a k-dimensional covariate vector, and y^i is a discrete response. We denote the domain of the response variable by $\mathcal{Y}$. We use the logistic regression model where a distribution over $\mathcal{Y} = \{0, 1\}$ given the covariates is specified by the formula

$$P(y = 1 \mid \mathbf{x}\,;\, \boldsymbol{\beta}) = \frac{1}{1 + e^{-\boldsymbol{\beta}'\mathbf{x}}}, \tag{1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^k$ is a parameter vector. Generalization to larger alphabets, i.e., multiple logistic regression, can be dealt with similarly[1]. As such, the logistic model is rather restricted since the log-odds of the outcomes follow an additive model where only the individual effects of the covariates appear. However, by replacing the covariste vector $\mathbf{x}$ by a vector of indicator functions $\mathbf{z}(\mathbf{x})$, one for each combination of the covariates, as shown in Eq. (2), we obtain a model that is complete in the sense that any conditional distribution can be represented by choosing suitable parameter values.

In this way, for instance, the different coefficient vectors for the case $p = 3$ are given by

$\mathbf{x}$	$\mathbf{z}(\mathbf{x})$
$(0,0,0)$	$(1,0,0,0,0,0,0,0)$
$(0,0,1)$	$(0,1,0,0,0,0,0,0)$
$(0,1,0)$	$(0,0,1,0,0,0,0,0)$
$\vdots$	$\vdots$
$(1,1,1)$	$(0,0,0,0,0,0,0,1)$

(2)

[1]Although it will be necessary to use the group Lasso or related methods [6]–[8] in order to obtain interpretable models.

With this mapping, we can represent any conditional distribution $P(y \mid \mathbf{x})$ by defining parameters

$$\beta(\mathbf{x}) = \ln \frac{P(y=1 \mid \mathbf{x})}{P(y=0 \mid \mathbf{x})}$$

and concatenating these parameters into a single vector $\boldsymbol{\beta} \in \mathbb{R}^{2^k}$ in the same order as the contexts are listed in the above table, i.e., $\boldsymbol{\beta}' = (\beta(000), \beta(001), \beta(010), \ldots, \beta(111))$. The dot product $\boldsymbol{\beta}'\mathbf{z}(\mathbf{x})$ simply picks the correct parameter from vector $\boldsymbol{\beta}$, and Eq. (1) yields the desired probability.

It is well known that logistic regression models are exponential families, which implies that the log-likelihood function is concave, see e.g. [9].

For fully parameterized discrete models, the number of parameters needed to specify the model is $|\mathcal{X}|^k(|\mathcal{Y}|-1)$ where it is assumed that all the covariates take values in the set $\mathcal{X}$; in the binary case 2^k. This makes it hard to estimate the parameters accurately for $k \gg 1$.

III. Transformed Lasso

In the Lasso (least absolute shrinkage and smoothing operator) [2], the log-likelihood is penalized by the ℓ_1 norm of the parameter vector[2]:

$$\max_{\boldsymbol{\beta}} \log P(\mathbf{y} \mid X\ ;\ \boldsymbol{\beta}) - \lambda\|\boldsymbol{\beta}\|_1, \tag{3}$$

where $\mathbf{y} = (y_1, \ldots, y_n)$ is the response sequence, and $X = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)})$ is the design matrix. In the standard linear-quadratic case, the log-likelihood is a quadratic function of the parameters. The ℓ_1 penalty has the property that usually many of the parameter estimates are equal to zero. Assuming that the log-likelihood function is downwards convex, as it is in the logistic regression case, the optimization problem can be solved efficiently by convex optimization methods; for the linear–quadratic case, see [2]–[4].

Under the logistic parameterization (1), letting parameter $\beta(\mathbf{x})$ be (close to) zero, results in a (nearly) uniform conditional distribution for y given $\mathbf{x}$. Thus, ℓ_1 penalization tends to *smooth* the parameter estimates towards the uniform distribution. However, this is not necessarily the only kind of sparsity (or "simplicity") we expect. For instance, we would often also like the *differences* between parameters to be small. For instance, if $\beta(100) = \beta(101)$, then the third covariate, x_3, has no effect given that the other two covariates take values $x_1 = 1, x_2 = 0$.

In the *fused Lasso* [10], one also penalizes for the absolute difference of subsequent parameters (ordered in a suitable fashion):

$$\max_{\boldsymbol{\beta}} \log P(\mathbf{y} \mid X\ ;\ \boldsymbol{\beta}) - \lambda_1\|\boldsymbol{\beta}\|_1 - \lambda_2 \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|, \tag{4}$$

where p is the number of parameters, and λ_1 and λ_2 are regularization parameters. If the parameters are ordered as above (see Eq. (2)), this would indeed favor to some extent parameter vectors where subsequent contexts give identical conditional distributions. This handles the case where the identity $\beta(100) = \beta(101)$ eliminates the effect of x_3 in a given context. However, it is unclear what the meaning of, say, $\beta(011) = \beta(100)$ is: the covariate vectors $(1, 1, 0)$ and $(0, 0, 1)$ are at first sight the opposite of each other; why should we have a bias that encourages their giving the same distribution for y? On the other hand, penalizing only the pairs that differ in x_k but agree in the other symbols[3], will penalize *only* the effect of the last covariate x_k.

More generally, we should of course penalize for the absolute difference between any two parameters which we like to be (almost) equal. Adding very many penalty terms will, however, slow down the optimization procedure. Our approach, which we outline next, is based on penalization of linear combinations of the logistic parameters and avoids the explicit use of additional pair-wise and higher-order penalties.

What we propose to do is to perform a suitable linear transformation on the parameters and use Lasso (ℓ_1) penalization on the *transformed* parameters. This yields the optimization problem:

$$\max_{\boldsymbol{\beta}} \log P(\mathbf{y} \mid X\ ;\ \boldsymbol{\beta}) - \lambda\|T\boldsymbol{\beta}\|_1, \tag{5}$$

where $\lambda > 0$ is a regularization parameter, and T is a linear transformation. We call this method the *transformed Lasso*.

The idea is that if the (original) parameter vector $\boldsymbol{\beta}$ has some smoothness properties captured by the transformation T, then $T\boldsymbol{\beta}$ is sparse, i.e., it has only a few non-zero coefficients. When estimating the parameters from data using (5), the estimates of these parameters tend to be set to zero. Since T is a linear transformation, the concavity of the penalized log-likelihood is retained.

In fact the transformed Lasso was proposed already by Tibshirani *et al.* [10] (using the Haar transformation) for the linear–quadratic case, but the authors found it ill-suited, mainly because in their example, the predictor structure was not 'dyadic', like it is in our case: while the predictors were ordered so that they formed blocks, i.e., consecutive runs of identical coefficients, the block boundaries did not occur near powers of two. This implies that the parameter vector is not necessarily sparse in the Haar domain. Due to the way we order the parameters, we automatically get such dyadic structures. We illustrate the idea below.

For orthogonal transformations, for which the transpose of the transformation matrix T gives the *inverse* transformation, $T'T = I$, the transformed Lasso problem (5) can be easily solved by existing Lasso techniques. Denoting the transformed parameters by $\boldsymbol{\eta} = T\boldsymbol{\beta}$, the problem can be re-written as

$$\max_{\boldsymbol{\eta}} \log P(\mathbf{x}\ ;\ T'\boldsymbol{\eta}) - \lambda\|\boldsymbol{\eta}\|_1, \tag{6}$$

where we used $T'\boldsymbol{\eta} = T'T\boldsymbol{\beta} = \boldsymbol{\beta}$. Now, consider the

[2]Alternatively, we can maximize the log-likelihood subject to an upper-bound on the ℓ_1 norm of $\boldsymbol{\beta}$.

[3]For $k = 3$, this means that we penalize by the sum $|\beta(000) - \beta(001)| + |\beta(010) - \beta(011)| + |\beta(100) - \beta(101)| + |\beta(110) - \beta(111)|$; compare this to the last term of the fused Lasso, Eq. (4).

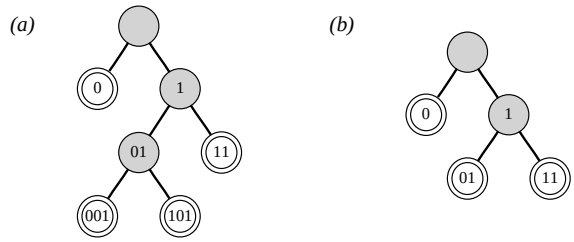

Fig. 1. Examples of context trees. In each tree, left and right branches correspond to zero and one, respectively. The shaded nodes indicate the relevant symbols, and the resulting contexts are given in the terminal nodes. If $y_{i-1} = 0$, then the context is simply 0 in both trees. On the other hand, if $y_{i-1} = 1$ and $y_{i-2} = 0$, then the context is 01 in tree *(b)*; in tree *(a)* one more bit, y_{i-2}, is needed determine the context.

likelihood function written in terms of the $\boldsymbol{\eta}$-parameters,

$$P(y = 1 \mid \mathbf{x}\,;\,\boldsymbol{\eta}) = 1/\left(1 + e^{-(T'\boldsymbol{\eta})'\mathbf{z}(\mathbf{x})}\right) = 1/\left(1 + e^{-\boldsymbol{\eta}'T\mathbf{z}(\mathbf{x})}\right). \quad (7)$$

From (6) and (7) we see that the transformed problem is equivalent to the usual Lasso problem with the inputs given by $T\mathbf{z}(\mathbf{x})$, i.e., the inputs are simply mapped through the transform. Having solved the optimization problem, the optimal parameter vector $\hat{\boldsymbol{\beta}}_\lambda$ can be obtained by the inverse transform, $\hat{\boldsymbol{\beta}}_\lambda = T'\hat{\boldsymbol{\eta}}_\lambda$, where $\hat{\boldsymbol{\eta}}_\lambda$ is the solution of (6).

IV. Variable length Markov chains

The model class of variable length Markov chains (VLMC), or context tree models, is extensively studied in information theory, see e.g. [11]–[14]. VLMC models can be characterized by *context trees*, examples of which are shown in Fig. 1. Given a context tree, the number of parameters needed is $L(|\mathcal{X}|-1)$, where L is the number of leaf nodes in the tree. Ordinary Markov models are a special case of VLMCs where the context tree is a balanced tree of depth k with $L = |\mathcal{X}|^k$ leaf nodes.

We now show that the transformed Lasso with the Haar transformation corresponds to learning VLMC models. For details of the Haar and other wavelet transformations, see e.g. [15]. The Haar transform matrix of order 8 (to be used for models with 8 parameters), which gives the *basis vectors* as its rows, is given by Eq. (8). The multipliers on the left make sure that each vector $\mathbf{u}_i$ is of unit length. The basis vectors are also orthogonal in the sense that $\mathbf{u}_i'\mathbf{u}_j = 0$ for all $i \neq j$. The (*forward*) Haar transform is mathematically equivalent to multiplying a vector by the Haar matrix, $\mathbf{v} \mapsto \mathcal{H}\mathbf{v}$, whereas the *inverse* Haar transform is equivalent to multiplying by the inverse of the matrix, which is by orthogonality equivalent to the transpose, $\mathbf{w} \mapsto \mathcal{H}'\mathbf{w}$. In practice, matrix multiplications are not used, since a simple and fast (linear-time) algorithm exist for the Haar transform and its inverse.

In order to apply the logistic regression model, we let the covariate vector $\mathbf{x}^{(i)}$ be determined by the k symbols preceding the ith symbol in the sequence $\mathbf{y}$. This needs to be done in *reverse* order, so that for $k = 3$, the covariate vector is given by $\mathbf{x}^{(i)} = (y_{i-1}, y_{i-2}, y_{i-3})$ — this way the most significant bit of the covariate vector is the immediate predecessor y_{i-1}. The initial part where $i \leq k$ can be dealt with, for instance, by assuming a fixed initial sequence $(\ldots, y_{-2}, y_{-1}, y_0)$ prefixed to the actual sequence $\mathbf{y}$ so that the preceding k symbols are always well defined.

The linear transform T in (5) is then the Haar transform $\boldsymbol{\beta} \mapsto \mathcal{H}\boldsymbol{\beta}$. This gives a representation of the parameter vector in terms of the Haar basis vectors. The first basis vector represents the mean of all the parameters, which gives the general "bias" (not statistical bias, however) towards $y_i = 1$. The second basis function gives the *difference* in bias towards $y_i = 1$ between the cases where $y_{i-1} = 0$ and those where $y_{i-1} = 1$. The third one gives the difference in bias between the cases where $y_{i-2} = 0, y_{i-1} = 0$ and those where $y_{i-2} = 1, y_{i-1} = 0$, and so on. In a given context, the bias is obtained as a sum of (possibly negated) coefficient values.

For instance, any Markovian model of order 2 can be represented using basis vectors 1,2,3, and 4 only. Any distribution compatible with the three-node context tree *(b)* in Fig. 1, where the different contexts are $\{0, 01, 11\}$, can be represented using three basis vectors, namely vectors 1, 2, and 4. In the binary case, the number of basis vectors needed to represent any VLMC is equal to L, the number of leaf nodes in the corresponding context tree, which is optimal.

On the other hand, since the number of context trees of maximum depth k is less than the number of subsets of k basis vectors, there are some subsets that correspond to no context tree. For instance, by setting certain coefficients to zero and retaining others, it is possible to eliminate the *individual* effect of y_{i-1} while retaining the effect of y_{i-2} given that $y_{i-1} = 0$, etc. In other words, it is possible for the effect of a symbol to cancel out *on the average* (when averaged over all contexts) even when symbols further away in the context have a non-

$$\mathcal{H} = \begin{bmatrix} 1/\sqrt{8} & \times & (1, & 1, & 1, & 1, & 1, & 1, & 1, & 1) \\ 1/\sqrt{8} & \times & (1, & 1, & 1, & 1, & -1, & -1, & -1, & -1) \\ 1/2 & \times & (1, & 1, & -1, & -1, & 0, & 0, & 0, & 0) \\ 1/2 & \times & (0, & 0, & 0, & 0, & 1, & 1, & -1, & -1) \\ 1/\sqrt{2} & \times & (1, & -1, & 0, & 0, & 0, & 0, & 0, & 0) \\ 1/\sqrt{2} & \times & (0, & 0, & 1, & -1, & 0, & 0, & 0, & 0) \\ 1/\sqrt{2} & \times & (0, & 0, & 0, & 0, & 1, & -1, & 0, & 0) \\ 1/\sqrt{2} & \times & (0, & 0, & 0, & 0, & 0, & 0, & 1, & -1) \end{bmatrix} \quad (8)$$

zero effect. We defer further discussion to the full version of the paper about when this is useful and when not.

In our application, it turns out that it is better to omit the scaling multipliers in (8). This is because the higher-order basis functions, like the four bottom rows in (8), are multiplied by a factor which is exponential (in the order of the effect) with respect to the factor of the lowest-level basis functions. This causes many spurious high-order effects to enter the model. The problem can be fixed by ignoring the multipliers in the transformation matrix (and adjusting the inverse transformation accordingly). We omit further details.

We now describe a simple experiment. Data was generated by sampling random binary sequences of given length from a fixed VLMC model described below. The maximum order of the effects in the transformed Lasso method was restricted to $k = 7$. We compare the estimated models to the generating model, and also estimate the negative log-likelihood by evaluating the per-symbol logarithmic prediction errors in a test sequence sampled from the same distribution.

For solving the transformed Lasso problem (5), we used the `glmpath` package[4] [16], which also gives a regularization path, i.e., the set of solutions obtained by letting the regularization parameter λ vary between some maximum value and zero. Having computed the regularization path, we selected the level of regularization, λ, by the Bayesian information criterion (BIC), see e.g. [17].

Example 1: Let the model be

$$P(y_i = 1 \mid \mathbf{y}_{i-k}^{i-1}) = \begin{cases} 0.2 & \text{if } \mathbf{y}_{i-3}^{i-1} = 000 \\ 0.4 & \text{if } \mathbf{y}_{i-3}^{i-1} = 100 \\ 0.55 & \text{if } \mathbf{y}_{i-3}^{i-1} = 010 \\ 0.4 & \text{if } \mathbf{y}_{i-5}^{i-1} = 00110 \\ 0.55 & \text{if } \mathbf{y}_{i-5}^{i-1} = 10110 \\ 0.55 & \text{if } \mathbf{y}_{i-5}^{i-1} = 01110 \\ 0.4 & \text{if } \mathbf{y}_{i-5}^{i-1} = 11110 \\ 0.2 & \text{if } \mathbf{y}_{i-2}^{i-1} = 01 \\ 0.35 & \text{if } \mathbf{y}_{i-2}^{i-1} = 11 \end{cases},$$

where $\mathbf{y}_{i-l}^{i-1} = (y_{i-l}, \ldots, y_{i-1})$.

Figure 2 shows the regularization path and the BIC curve in a representative run with sample size $n = 2048$, together with the obtained maximum likelihood and penalized parameter estimates. It can be seen that the maximum likelihood estimates in panel *(c)* (obtained by setting $\lambda = 0$) are very noisy; many of them are either zero or one since they are associated with contexts where only one of the outcomes has occurred. The transformed Lasso estimates, panel *(d)*, are much more stable and give a better estimate of the true parameters.

Compared to existing methods for learning VLMC models, the transformed Lasso typically produces similar model selection and compression (prediction) performance. Further details will be provided in a full version of this paper.

[4]R package available from CRAN, http://cran.R-project.org/.

V. Linear Combinations of XOR Functions

As a second example, we consider the Walsh-Hadamard (WH) transformation, see [18]. It gives a decomposition of the parameter vector in terms of exclusive-OR (parity) functions of increasing order. The WH matrix of order 2^k is given by the recursion

$$H_{2^k} = \frac{1}{\sqrt{2}} \begin{pmatrix} H_{2^{k-1}} & H_{2^{k-1}} \\ H_{2^{k-1}} & -H_{2^{k-1}} \end{pmatrix}$$

with $H_1 = [1]$. For instance, the order 2 and 4 WH matrices are given by

$$H_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_4 = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}. \quad (9)$$

Due to the construction, the basis vectors correspond to all the 2^k possible XOR (parity) functions of k Boolean (binary) variables. For instance, consider the order 4 matrix in (9). If we associate with each column an instantiation of two binary variables, x_1 and x_2, in the order $(x_1, x_2) = ((0,0),(0,1),(1,0),(1,1))$, then the rows correspond, from top to bottom, to functions XOR(1), $\mathrm{XOR}(x_2)$, $\mathrm{XOR}(x_1)$, $\mathrm{XOR}(x_1, x_2)$, where XOR(1) denotes the constant function.

The WH transformation corresponds to multiplying a vector by the WH matrix, $\boldsymbol{\beta} \mapsto H\boldsymbol{\beta}$, and the inverse transform is obtained as the transpose. Again, in practice no matrix multiplications are necessary since fast and simple algorithms exist. The transformed Lasso problem with the WH transformation can be solved using standard tools by mapping the vector of indicator functions, $\mathbf{z}(\mathbf{x})$, Eq. (2), through the WH transformation.

It may seem that a representation in terms of XOR functions is the worst possible alternative for modeling naturally occurring data. However, the key property of the WH basis is that any pattern is decomposed into sub-patterns of increasing order in such a way that if the actual pattern can be represented as a sum of *any* low order Boolean functions — which do not have to be XOR functions —, then none of the higher-order XOR functions are used. For instance, if $I_0 \subseteq \{1, \ldots, k\}$ is a subset of indices, then the conjuction of the corresponding covariates can be represented as the sum

$$\mathrm{AND}(\{x_{I_0}\}) = \sum_{m=1}^{|I_0|} (-1)^{m-1} \sum_{\substack{I' \subseteq I_0 \\ |I'|=m}} \mathrm{XOR}(\{x_{I'}\}),$$

where $\mathrm{AND}(\{x_I\})$ and $\mathrm{XOR}(\{x_I\})$ denote the conjunction and exclusive-OR of covariates with indices in set I respectively. For example, $\mathrm{AND}(x_1, x_2) = \mathrm{XOR}(x_1) + \mathrm{XOR}(x_2) - \mathrm{XOR}(x_1, x_2)$. As can be seen from the expression, the terms in the sum are at most of order $|I_0|$. Similar expressions apply to other Boolean functions, see [19].

Example 2: Seven covariates, $x_1, \ldots, x_7$, are generated independently with uniform probability over $\{0, 1\}$. The model is given by

$$P(y = 1 \mid \mathbf{x}) = 1/(1 + e^{x_1 - 0.5\,\mathrm{XOR}(x_2,x_3) + 2\,\mathrm{OR}(x_4,x_5,x_6)}).$$

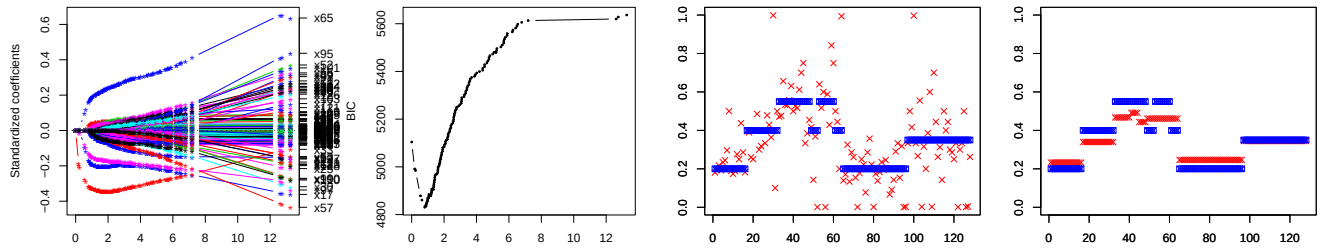

Fig. 2. *(a)* The regularization path obtained by `glmpath`. Penalized coefficients are plotted against optimization step. *(b)* The BIC criterion plotted against optimization step. *(c)* Maximum likelihood and *(d)* penalized estimates. The true parameter values, which are the same in both panels, are shown with blue squares (□), the estimates with red crosses (×). Data was generated from the model in Example 1, with sample size $n = 2048$.

TABLE I
FREQUENCIES OF CORRECTLY IDENTIFYING TERMS IN THE TRUE MODEL OF EXAMPLE 2 OUT OF 100 TRIALS, AND THE AVERAGE FREQUENCY OF FALSE POSITIVES ('OTHERS').

	FUNCTION	FREQUENCY
1.	$\mathrm{XOR}(x_1)$	100
2.	$\mathrm{XOR}(x_2, x_3)$	100
3.	$\mathrm{XOR}(x_4)$	100
4.	$\mathrm{XOR}(x_5)$	100
5.	$\mathrm{XOR}(x_6)$	100
6.	$\mathrm{XOR}(x_4, x_5)$	100
7.	$\mathrm{XOR}(x_4, x_6)$	100
8.	$\mathrm{XOR}(x_5, x_6)$	100
9.	$\mathrm{XOR}(x_4, x_5, x_6)$	100
	others (avg.)	3.53

Under the WH basis, the true model includes nine basis functions: the functions $\mathrm{XOR}(x_1)$, $\mathrm{XOR}(x_2, x_3)$, and all the seven functions involving variables x_4, x_5, x_6 (items 3–9 in Table I). The latter ones result from the decomposition of the OR function in terms of XOR functions.

Table I lists the frequencies with which these functions are included in the estimated model in 100 trials when the models were learned by transformed Lasso from samples of size $n = 1600$; model complexity was chosen using BIC. All the effects were identified correctly in all trials. The average number of spurious effects (false positives) per trial was 3.53.

ACKNOWLEDGMENT

TR was supported in part by the Academy of Finland (project Modest). BY was supported in part by grants NSF DMS-0605165, NSFC (60628102), and an NSF CDI award.

REFERENCES

[1] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[2] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Statist. Soc. B*, vol. 58, no. 1, pp. 267–288, 1996.

[3] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *J. Num. Analysis*, vol. 20, pp. 389–403, 2000.

[4] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, 2004.

[5] T. Hesterberg, N. H. Choi, L. Meier, and C. Fraley, "Least angle and ℓ_1 penalized regression: A review," *Statist. Surveys*, vol. 2, pp. 61–93, 2008.

[6] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Royal Statist. Soc. B*, vol. 68, no. 1, pp. 49–67, 2005.

[7] L. Meier, S. van de Geer, and P. Bühlmann, "The group lasso for logistic regression," *J. Royal Statist. Soc. B*, vol. 70, no. 1, pp. 53–71, 2008.

[8] P. Zhao, G. V. Rocha, and B. Yu, "The composite absolute penalties family for grouped and hierarchical variable selection," *Annals of Statistics*, to appear.

[9] G. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, New York, 1992.

[10] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *J. Royal Statist. Soc. B*, vol. 67, no. 1, pp. 91–108, 2005.

[11] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Trans. Inform. Theory*, vol. 30, pp. 629–636, 1983.

[12] M. Weinberger, J. Rissanen, and M. Feder, "A universal finite memory source," *IEEE Trans. Inform. Theory*, vol. 41, pp. 643–652, 1995.

[13] P. Bühlmann and A. Wyner, "Variable length Markov chains," *Ann. Statist.*, vol. 27, pp. 480–513, 1998.

[14] F. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inform. Theory*, vol. 41, pp. 653–664, 1995.

[15] S. Mallat, *A Wavelet Tour of Signal Processing*. San Diego, CA: Academic Press, 1998.

[16] M. Y. Park and T. Hastie, "L1 regularization path algorithm for generalized linear models," *J. Royal Statist. Soc. B*, vol. 69, pp. 659–677, 2007.

[17] H. Zou, T. Hastie, and R. Tibshirani, "On the "degrees of freedom" of the Lasso," *Ann. Statist.*, vol. 35, no. 5, pp. 2173–2192, 2007.

[18] T. Beer, "Walsh transforms," *Am. J. Phys.*, vol. 49, no. 5, 1981.

[19] M. G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*. New York, NY: John Wiley & Sons, 1976.