An Improved Penalty Parameter Selection Procedure for the Lasso AR-X

William B. Nicholson[*], Xiaohan Yan[†]

August 26, 2015

Abstract

Many recent developments in the statistical time series literature have centered around incorporating the Lasso, a feature selection procedure originally designed for least squares problems, to applications with time dependent data. The Lasso requires the specification of a penalty parameter that determines the degree of sparsity to impose. The most popular cross validation approaches that respect time dependence are very computationally intensive and not appropriate for modeling certain classes of time series. We propose a novel adaptive penalty parameter selection procedure that takes advantage of the sequentially observed nature of time series data to improve both computational performance and forecast accuracy in comparison to existing methods.

1 Introduction

Selecting relevant features in a time series model is a longstanding open problem in both statistics and econometrics. Traditional feature selection approaches, such as those proposed by Box & Jenkins (1994) rely on heuristic methods, such as visual inspection of diagnostic plots. Alternatively, popular data-driven approaches select features based upon the minimization of information criterion, such as Akaike's Information Criterion (AIC, Akaike (1974)) or Bayesian Information Criterion (BIC, Schwarz et al. (1978)) over a subset of potential models. Typically, because the space of all possible models is extremely large, such methods require imposing substantial restrictions on the feature space.

More recent approaches have extended the Lasso Tibshirani (1996) to a time dependent setting. The Lasso has advantages over conventional methods in that it shrinks least squares estimates toward zero in addition to performing feature selection. It also allows for estimation under scenarios in which there are more potential features than observations.

The Lasso utilizes a penalty parameter that controls the degree of sparsity in a solution. In most scenarios there is no theoretical basis for choosing this parameter, so it must be estimated empirically. A standard selection approach, n-fold cross validation, does not respect time dependence. "Rolling" cross validation, a popular approach used by Song & Bickel (2011), Banbura et al. (2009), and Koop (2011) involves incrementing the data forward one observation at a time over a training period

[*]PhD Candidate, Department of Statistical Science, Cornell University, 301 Malott Hall, Ithaca, NY 14853, (E-mail: wbn8@cornell.edu; Webpage: http://www.wbnicholson.com)

[†]PhD Student Cornell University, 301 Malott Hall, Ithaca, NY 14850 USA E-mail: xy257@cornell.edu

and recording the loss over a fixed grid of penalty values. This procedure is very computationally intensive, as it requires recalculating the Lasso solution at every timepoint despite adding just one observation.

Moreover, the "optimal" value reported from a fixed grid of penalty parameters might not be an appropriate in modeling certain classes of time dependent problems; in particular, nonstationary time series in which the degree of dependence may vary across time. Penalty parameter selection procedures designed for nonstationary time series should be able to adapt to varying degrees of dependence.

By utilizing an online procedure proposed by Garrigues & El Ghaoui (2008) that updates the current Lasso solution as a new observation is received rather than completely reestimating, we are able to substantially improve upon the computational performance of the Lasso. While incorporating this procedure, we present an adaptive updating scheme that allows the regularization parameter to freely vary over the training period in order to better accommodate a larger class of time dependent problems.

Section 2 introduces the autoregressive with exogenous variables framework that is used throughout the paper, explains the incorporation of the Lasso penalty and addresses the issues with the conventional penalty parameter selection procedure, Section 3 presents the online updating scheme used to improve computational performance and our proposed adaptive regularization scheme. Section 4 details our results in both simulations and a macroeconomic data application, and Section 5 presents our conclusion.

2 Methodology

In this section, we will provide a brief overview of autoregressive modeling with exogenous variables. We start by demonstrating that the Lasso AR-X offers greater flexibility than information criterion based methods in selecting relevant features. We additionally provide an overview of rolling cross validation and detail some of its shortcomings.

2.1 Framework

Our analysis will operate in the context of autoregressive processes with exogenous variables (AR-X), a canonical time series model that is amenable toward applying a Lasso penalty, as it can be formulated as a least squares problem.

We consider forecasting the length T series $\{y_t\}_{t=1}^{T} \in \mathbb{R}^T$ using its p most recent lagged values $(y_{t-1}, \ldots, y_{t-p})$ as well as s lagged values of $\{\boldsymbol{x}_t\}_{t=1}^{T} \in \mathbb{R}^{T\times k}$, which represent k unmodeled, exogenous series. In the context of this paper, a time series is considered *exogenous* if it is not modeled, but it aids in forecasting our series of interest y_t. For example, if we are forecasting the US Federal Funds Rate, potential exogenous series can include other relevant macroeconomic indicators, such as the Consumer Price Index or Gross Domestic Product growth rate.

Typically, maximal lag orders (i.e. the maximum number of lagged features included in the model) for both y_t and $\boldsymbol{x}_t$ are chosen according to the the frequency of the data. For example if the series are recorded quarterly, one might choose $p = s = 4$ to represent one year of past dependence; for monthly data, one might select $p = s = 12$.

A forecast from an AR-X(p, s) model at time t can be estimated by least squares via the objective function

$$\min_{\phi,\theta} \frac{1}{2}\|y_t - \sum_{i=1}^{p} \phi_i y_{t-i} - \sum_{i=1}^{k}\sum_{j=1}^{s} \theta_{ij} \boldsymbol{x}_{i,t-j}\|_2^2, \tag{1}$$

in which $\phi_i, \theta_{ij} \in \mathbb{R}$ denote least squares coefficients.

2.2 Feature Selection

In general, it is not desirable to report forecasts from an AR-X(p, s) if p and s are large, due to concerns of overfitting. Since the least squares objective function decreases monotonically as the number of features included increases, it is natural to apply a penalty that restricts the feature space.

Conventional penalized approaches involve the use of information criterion (henceforth IC). Rooted in information theory, IC based methods, such as AIC and BIC provide a coherent framework for feature selection. A standard approach involves fitting several nested models over a subset of the feature space and selecting the model that minimizes a chosen IC.

As stated in Hsu et al. (2008), the space of all potential models can be extremely large. Even if p, k, and s are relatively small, it is not computationally tractable to fit each of the possible 2^{p+ks} subset models.

In the IC setting, the model space is typically restricted by assuming that all coefficients are nonzero up to maximal lag orders $\hat{p}$ and $\hat{s}$. These lag orders are typically selected by fitting an AR-X by least squares (using equation (1)) for $1 \leq \ell \leq p$, $1 \leq j \leq s$ and selecting $\hat{p}$ and $\hat{s}$ based on the minimization of an IC. The AIC and BIC of an AR-X(ℓ, j) are defined as:

$$\begin{aligned} \text{AIC}(\ell, j) &= \log(\hat{\sigma}_u^{\ell,j}) + \frac{2(kj + \ell)}{T}, \\ \text{BIC}(\ell, j) &= \log(\hat{\sigma}_u^{\ell,j}) + \frac{\log(T)(kj + \ell)}{T}, \end{aligned}$$

in which $\hat{\sigma}_u^{\ell,j}$ is the squared residual obtained from using Equation (1) to fit an AR-X(ℓ, j). As AIC penalizes model coefficients uniformly by a factor of 2 whereas BIC scales penalties according to series length, BIC will tend to select more parsimonious models than will AIC.

Since we are limited to a subset of potential models, IC based methods can be very restrictive. In particular, it is typically assumed that every exogenous series in $\boldsymbol{x}_t$ has the same maximal lag order, although as shown by Penm et al. Penm et al. (1993), this rarely holds empirically.

In contrast, the Lasso solution can be obtained by adding the penalty

$$\lambda\left(\sum_{i=1}^{p} \|\phi_i\|_1 + \sum_{i=1}^{k}\sum_{j=1}^{s} \|\theta_{ij}\|_1\right)$$

to equation (1). We denote $\lambda \geq 0$ as the parameter that controls the tradeoff between the least squares fit and the penalty. As opposed to other penalized regression methods, such as ridge regression, due to the inclusion of the L_1 penalty, the Lasso returns a sparse solution vector. Larger values of λ tend to lead to more sparse solutions. In contrast to IC methods, the Lasso performs estimation and feature selection in one step and does not require imposing any restrictions on the feature space.

Figure 1 depicts an example highlighting the differences in features selected by IC minimization and the Lasso. The Lasso can select any subset of features, hence it can more accurately capture the non-monotonic feature relationships that are common in time series, such as seasonal dependence (e.g. nonzero coefficients at every jth lag). In addition, the Lasso can allow the dependence among exogenous features within a lag to vary.

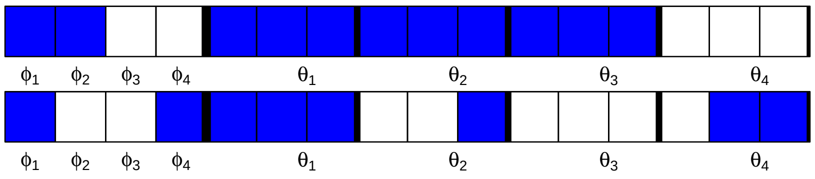
ϕ_1 ϕ_2 ϕ_3 ϕ_4 θ_1 θ_2 θ_3 θ_4
ϕ_1 ϕ_2 ϕ_3 ϕ_4 θ_1 θ_2 θ_3 θ_4

Figure 1: Example features selected (shaded) from an AR-X(4,4) with 3 exogenous series. The top figure corresponds to features selected by IC minimization whereas the bottom corresponds to the Lasso.

2.3 Penalty Parameter Selection

In a time-dependent context, penalty parameter selection is not well suited to traditional n-fold cross-validation. Instead, *rolling* cross validation, put forth by Banbura et al. (2009) selects the optimal penalty parameter as the minimizer of a loss function after iterating forward one observation at a time over a training period. This procedure divides the data into three periods: one for initialization, one for training, and one for forecast evaluation. For the purposes of this paper, we will define time indices $T_1 = \lfloor \frac{T}{3} \rfloor, T_2 = \lfloor \frac{2T}{3} \rfloor$. The period $T_1 + 1$ through T_2 is used for training and $T_2 + 1$ through T is a holdout set reserved for evaluation of forecast accuracy. The procedure is illustrated in Figure 2.

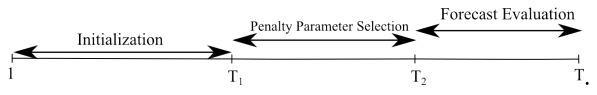

Figure 2: Illustration of Rolling Cross-Validation

Define $\hat{\mathbf{y}}_{t+1}^{\lambda_i}$ as the one-step ahead forecast based on all observations from $1, \ldots, t$ corresponding to a given penalty parameter λ_i. Rolling cross validation then chooses λ from a predetermined grid of n values $(\lambda_1, \ldots, \lambda_n)$ as the minimizer of one step ahead mean squared forecast error (MSFE) averaged over the training period

$$\text{MSFE}(\lambda_i) = \frac{1}{(T_2 - T_1 - 1)} \sum_{t=T_1}^{T_2-1} (\hat{y}_{t+1}^{\lambda_i} - y_{t+1})^2,$$
$$i = 1, \ldots, n.$$

Other loss functions could be considered, but MSFE is the most natural given our use of a least squares objective function. One of the major drawbacks of rolling cross validation procedure its computational burden; it requires solving $T_2 - T_1 - 1$ Lasso optimization problems over a grid of n penalty values. Though the Lasso can be solved efficiently via an iterative procedure such as coordinate descent Friedman et al. (2010), a single pass has complexity O($(p + ks)T$), which can be burdensome if p, s, k and T are large.

2.3.1 Issues with nonstationarity

Following Friedman et al. Friedman et al. (2010), the grid of penalty values is constructed by starting with the smallest value in which all features will be set to zero, and then decrementing in log linear increments. The depth of the grid as well as the number of gridpoints are usually left to user input.

Under many scenarios, optimizing based on a fixed grid of penalty parameters may not be appropriate. For example, most time series models implictly assume that the series is *stationary*, which, as defined by Shumway $\&$ Stoffer Shumway & Stoffer (2010), requires that the conditional mean of the series is constant across all observations and the covariance between two observations t_0 and t_1 is a function of their distance $(t_1 - t_0)$. Such assumptions are very restrictive and eliminate series exhibiting trendlike behavior as well as those with a substantial degree of variability.

There is a widespread belief that most financial Fama (1965) and macroeconomic Litterman (1979) time series exhibit nonstationary behavior. As an illustration, consider Figure 3, which depicts the log transformed US Gross Domestic Product Growth Rate (GDP), Consumer Price Index (CPI), and Federal Funds Rate (FFR). Note that both the FFR and CPI appear to have a distinct linear trend, implying that the mean of each series is not constant across time.

Under nonstationarity, selecting a penalty parameter as the minimizer of MSFE averaged over a training period seems inappropriate, as the optimal degree of regularization may vary at different points in the series. In the next section, we detail an adaptive approach that can more accurately capture the nonstationary tendencies of these time series.

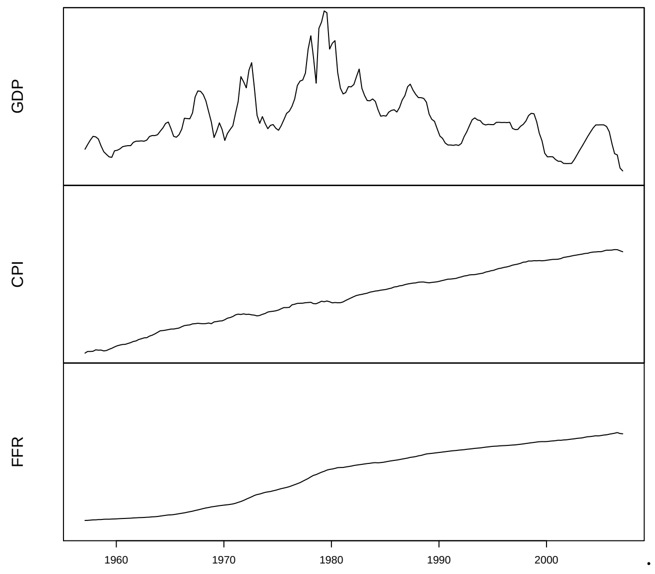

Figure 3: Plots exhibiting the nonstationary tendencies of the (log transformed) US Gross Domestic Product Growth Rate (GDP, top), Consumer Price Index (CPI, middle), and Federal Funds Rate (FFR, bottom). The data is collected quarterly and runs from Quarter 2 of 1957 to Quarter 4 of 2007.

3 Proposed Framework

In this section we describe the recursive Lasso algorithm adopted from Garrigues & El Ghaoui (2008) used for online updating of the Lasso solution. In addition, we propose a penalty parameter selection scheme that combines the conventional rolling approach with an adaptive updating scheme.

3.1 Compact Matrix Notation

In describing our algorithm, it will be convenient to reparameterize the Lasso problem from Section 2 in compact matrix notation. Let

$$\boldsymbol{z}_t = [y_{t-1}, \ldots, y_{t-p}, \boldsymbol{x}_{1,t-1}, \ldots, \boldsymbol{x}_{k,t-s}]$$

represent all lagged observations at time t, and let

$$\boldsymbol{B} = [\phi_1, \ldots, \phi_p, \ldots, \theta_{11}, \ldots, \theta_{ks}]^\top$$

denote the vector of coefficients. Then, we can express the Lasso AR-X optimization problem as

$$\arg\min_{\boldsymbol{B}} \frac{1}{2} \sum_{t=1}^{T} \|y_t - \boldsymbol{z}_t \boldsymbol{B}\|_2^2 + \lambda \|\boldsymbol{B}\|_1. \tag{2}$$

3.2 Online Updating Using RecLasso

Online updating is performed throughout the training period, extending the Recursive Lasso (RecLasso) algorithm from Garrigues & El Ghaoui (2008) to a time dependent setting. RecLasso is heavily influenced by the Least Angle Regression (Lars) algorithm proposed by Efron et al. Efron et al. (2004), which computes the Lasso solution as new features of $\boldsymbol{B}$ enter and leave as the penalty parameter decreases. The RecLasso algorithm greatly reduces computational requirements by keeping track of changes in the set of nonzero elements of $\boldsymbol{B}$. In practice, when the solution is sparse and the nonzero elements of $\boldsymbol{B}$ do not change much as a new observation is added, RecLasso is much more efficient than coordinate descent.

Let $Z \in \mathbb{R}^{t\times(p+ks)}$ be the matrix whose i^{th} row is equal to $\boldsymbol{z}_i$, and $y = (y_1, \ldots, y_t)^\top$. Denote the indices of nonzero elements of $\boldsymbol{B}$ as the *active set*. For simplicity of notation, let $\boldsymbol{B}_1 \subset \boldsymbol{B}$ denote the active features. We partition Z so that the columns of Z_1 correspond to the active set. Define v_1 be such that $v_{1i} = \text{sign}(\boldsymbol{B}_{1i})$ for all i in the active set. From the optimality condition of the Lasso in (2), if the active set and its signs are known, we can construct

$$\boldsymbol{B}_1 = (Z_1^\top Z_1)^{-1}(Z_1^\top y - \lambda v_1).$$

Suppose we have $\boldsymbol{B}^{(t)}$ as the solution to the Lasso problem at time t. Now, with a new observation $(\boldsymbol{z}_{t+1}, y_{t+1}) \in \mathbb{R}^{p+ks} \times \mathbb{R}$

and the corresponding penalty λ_{t+1}, we can compute $\boldsymbol{B}^{(t+1)}$ through the following augmented problem:

$$\boldsymbol{B}(\gamma, \lambda) = \arg\min_{\boldsymbol{B}} \frac{1}{2} \left\| \begin{pmatrix} y \\ \gamma y_{t+1} \end{pmatrix} - \begin{pmatrix} Z \\ \gamma \boldsymbol{z}_{t+1} \end{pmatrix} \boldsymbol{B} \right\|_2^2 + \lambda \|\boldsymbol{B}\|_1$$

From the augmented problem, we can express $\boldsymbol{B}^{(t)} = \boldsymbol{B}(0, \lambda_t)$ and $\boldsymbol{B}^{(t+1)} = \boldsymbol{B}(1, \lambda_{t+1})$. The RecLasso algorithm consists of two steps:

Step 1 Without considering the new observation (i.e., keep $\gamma = 0$), vary the penalty from λ_t to λ_{t+1}. Update $\boldsymbol{B}_1$ at *transition points* where the active set changes (i.e. a new feature is added or an existing one is removed).

Step 2

- Initialize the active set as the indices of the non-zero coefficients of $\boldsymbol{B}(0, \lambda_{t+1})$ and let

$$v = \text{sign}(\boldsymbol{B}(0, \lambda_{t+1})), \qquad \tilde{Z} = \begin{pmatrix} Z \\ \boldsymbol{z}_{t+1} \end{pmatrix}, \tilde{y} = \begin{pmatrix} y \\ y_{t+1} \end{pmatrix}.$$

- Initialize $\tilde{\boldsymbol{B}}_1 = (\tilde{Z}_1^\top \tilde{Z}_1)^{-1}(\tilde{Z}_1^\top \tilde{y} - \lambda_{t+1} v_1)$ and $\gamma = 0$.

- Keep computing the next transition point γ until $\gamma > 1$. At each transition point, update $v_1, \tilde{Z}_1$ according to the updated active set; update

$$\tilde{\boldsymbol{B}}_1 = (\tilde{Z}_1^\top \tilde{Z}_1)^{-1}(\tilde{Z}_1^\top \tilde{y} - \lambda_{t+1} v_1)$$

.

- Values of the active set of $\boldsymbol{B}^{(t+1)}$ are given by $\tilde{\boldsymbol{B}}_1$.

Note that since at each transition point, at most one feature can leave or enter the active set, we don't need to explicitly compute a matrix inverse when updating $\tilde{\boldsymbol{B}}_1$. We instead utilize the Sherman-Morrison formula Sherman & Morrison (1950) to perform rank one updates to $(\tilde{Z}_1^\top \tilde{Z}_1)^{-1}$.

3.3 Adaptive Regularization Procedures

As mentioned in Section 2.3, rolling cross validation, the conventional penalty parameter selection is based on minimizing the MSFE averaged over the training period. As the first step of RecLasso algorithm involves varying the penalty parameter over time, it is very amenable to an adaptive updating scheme. Our proposed Adaptive Regularization methods address two drawbacks to rolling cross validation. First, rolling cross validation is very computationally intensive. Second, a fixed grid of penalty values cannot accurately account for nonstationarity.

Garrigues & El Ghaoui (2008) describe an approach that adaptively updates the penalty parameter using each point in the training period as a test set. Their adaptive regularization process successively determines the amount of regularization in a data-driven manner at a substantially lower computational overhead than conventional methods. However, in practice we observed that the selected penalty parameter depends heavily on the starting value of the adaptive process, which consequently affects the performance of our estimator.

In order to better select the starting penalty value, we divide the training period into two parts. In the first half of the training period T_1 to T_2', we perform rolling cross validation on a fixed grid of penalty parameters. We use the optimal λ value from rolling cross validation as a starting value and adaptively update the penalty value over the second half of the training period T_2' to T_2. The procedure is illustrated in Figure 4.

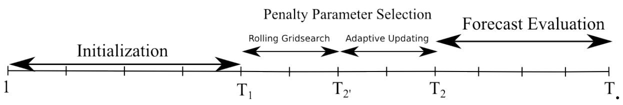

Figure 4: Illustration of Adaptive Regularization Procedure

We propose two adaptive updating procedures for the second half of the training period, which we outline in Section 3.3.1 and Section 3.3.2. Let $\lambda_{T_2'}$ be the optimal penalty selected from rolling cross validation in the first half of the training period. For $t \in [T_2', T_2 - 1]$, let λ_t be the penalty value at point t. We use the next observation, $t + 1$, as a "test set" and update λ_t in the direction that minimizes the prediction error $err(\lambda) = (y_{t+1} - \boldsymbol{z}_{t+1}\boldsymbol{B}(\lambda))^2$ where

$$\boldsymbol{B}(\lambda) = \arg\min_{\boldsymbol{B}} \frac{1}{2}\sum_{i=1}^{t}(y_i - \boldsymbol{z}_i\boldsymbol{B})^2 + \lambda\|\boldsymbol{B}\|_1.$$

The direction of the update and the step size can be determined using either Gradient Descent or Newton's Method.

3.3.1 Gradient Adaptive Regularization

In a similar manner to the scheme proposed in Garrigues & El Ghaoui (2008), we first consider using gradient descent to update λ. Following the previous notation, let $\boldsymbol{B}_1$ and Z_1 be the active features and the submatrix with columns corresponding to the active set at time t, respectively. Let v_1 denote the signs of the active set. For the next observation $(\boldsymbol{z}_{t+1}, y_{t+1})$, partition $\boldsymbol{z}_{t+1}$ so that $\boldsymbol{z}_{t+1,1}$ corresponds to the active set. The error function of the $(t + 1)^{st}$ observation is

$$\begin{aligned} err(\lambda) &= (\boldsymbol{z}_{t+1}^{\top}\boldsymbol{B}(\lambda) - y_{t+1})^2 \\ &= (\boldsymbol{z}_{t+1,1}^{\top}\boldsymbol{B}_1(\lambda) - y_{t+1})^2 \end{aligned}$$

where $\boldsymbol{B}_1(\lambda) = (Z_1^\top Z_1)^{-1}(Z_1^\top y - \lambda v_1)$. The subdifferential $\nabla err(\lambda) = \frac{\partial err(\lambda)}{\partial \log \lambda}$ is

$$\begin{aligned} &2\left(\boldsymbol{z}_{t+1,1}^\top \boldsymbol{B}_1(\lambda) - y_{t+1}\right) \cdot \left(-\boldsymbol{z}_{t+1,1}^\top (Z_1^\top Z_1)^{-1} \lambda v_1\right) \\ =& -2v_1 \lambda \boldsymbol{z}_{t+1,1}^\top (Z_1^\top Z_1)^{-1} (\boldsymbol{z}_{t+1,1}^\top \boldsymbol{B}_1(\lambda) - y_{t+1}) \end{aligned}$$

Following the update rule using Gradient Descent to minimize $err(\lambda)$ on a $\log$-scale of λ with the current $\lambda = \lambda_t$, we have

$$\begin{aligned} \log(\lambda_{t+1}) &= \log(\lambda_t) - \eta \frac{\partial err(\lambda_t)}{\log \lambda} \\ \Rightarrow \quad \lambda_{t+1} &= \lambda_t \times \exp\left\{-\eta \frac{\partial err(\lambda_t)}{\log \lambda}\right\}, \end{aligned}$$

where η, the learning rate controlling the stepsize is small; in our simulations and data applications in Section 4, we set it to 0.01. Note that we perform the update in the $\log$ domain to ensure that λ_t is always positive.

3.3.2 Newton Adaptive Regularization

We additionally propose an update rule that uses the Newton's Method to minimize $err(\lambda)$ on the $\log$-scale of λ with the current $\lambda = \lambda_t$. Newton's Method takes curvature into account; its stepsize is inversely related to "steepness", which should make its update more adaptive than using a fixed step size. Newton's Method requires the computation of the Hessian matrix of $err(\lambda)$:

$$\begin{aligned} H_{err}(\lambda) =& \frac{\partial^2 err(\lambda)}{\partial(\log \lambda)^2} \\ =& -2v_1 \lambda \boldsymbol{z}_{t+1,1}^\top (Z_1^\top Z_1)^{-1} \left(\boldsymbol{z}_{t+1,1}^\top \boldsymbol{\Sigma}(\lambda) - y_{t+1}\right) \end{aligned}$$

where $\boldsymbol{\Sigma}(\lambda) = (Z_1^\top Z_1)^{-1}(Z_1^\top y - 2\lambda v_1)$. Following Newton's update rule, we have

$$\begin{aligned} \log \lambda_{t+1} &= \log \lambda_t - \frac{\nabla err(\lambda_t)}{H_{err}(\lambda_t)} \\ \Rightarrow \quad \lambda_{t+1} &= \lambda_t \times \exp\left\{\frac{\boldsymbol{z}_{t+1,1}^\top \boldsymbol{B}_1(\lambda_t) - y_{t+1}}{\boldsymbol{z}_{t+1,1}^\top \boldsymbol{\Sigma}(\lambda_t) - y_{t+1}}\right\} \end{aligned}$$

4 Results

In this section, we will compare the forecasting and computational performance of our adaptive procedures with conventional IC methods as well as the standard Lasso (with the penalty parameter selected by rolling cross validation). We present both a simulation example and a macroeconomic data application. All of our analysis was performed in R R Core Team (2015). Our RecLasso implementation was heavily influenced by the Python code made publicly available by Garrigues & El Ghaoui Garrigues & El Ghaoui (2008).

4.1 Computational Performance

Global convergence rates of coordinate descent have not been well established; theoretical results exist only under very restrictive specifications Saha & Tewari (2010). Similarly, the computational complexity of the adaptive procedure depends heavily on the size of the active set and on the number of transition points that occur as a new observation is added. Hence, it is outside the scope of this paper to perform a theoretical analysis of the computational complexity of the adaptive lasso procedure.

Instead, we present an empirical computational experiment. Using the `microbenchmark` package in R, we calculate the average computational time of updating the Lasso AR-X solution from time t to time t+1, in which $t = 60, p = 12, s = 12, k = 10$. At every iteration, there is at least one transition point as the Lasso solution is updated. We compare our adaptive updating procedure with the conventional approach that completely re-solves the Lasso problem via coordinate descent using the previous period's result as a "warm start."

The results are recorded in Table 1 and visualized in Figure 5. Note that despite utilizing a warm start, the adaptive updating procedure is orders of magnitude faster than completely refitting with coordinate descent.

Table 1: : Distribution of computational time (in microseconds) of 1000 iterations of Coordinate Descent using the previous period's solution as a "warm start" versus the Adaptive Lasso procedure.

Procedure	min	lower quartile	mean	median	upper quartile	max
Coordinate Descent	124938.59	131971.22	160468.06	136999.20	198328.85	265418.39
Adaptive Lasso	502.86	635.03	748.78	735.50	781.45	4086.62

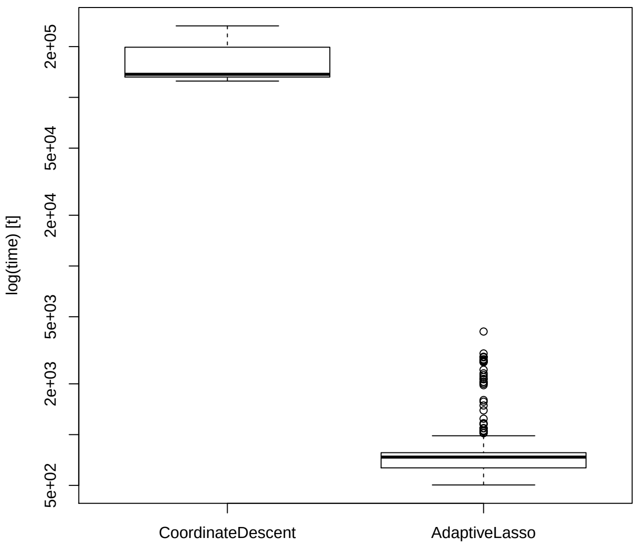

Figure 5: Boxplots comparing the computational performance of Coordinate Descent and the Adaptive Lasso Procedure.

4.2 Simulation Results

We consider simulating from a stationary AR-X model with $k = 10, p = 13, s = 13$, (143 potential features) with series length $T = 100$. The simulation structure consists of 10 nonzero features selected at random.

We compare the forecasting performance of our adaptive regularization approaches with the standard rolling Lasso. We also compare against three conventional approaches: the conditional sample mean, and two IC based methods.

The conditional sample mean forecasts $\hat{y}_{t+1}$ using the mean of all observations up to time t. Under scenarios with evidence of nonstationarity, the sample mean will provide very poor forecasts. We additionally consider selecting the lag order of an AR-X according to minimization of AIC and BIC (note that maximal lag orders are restricted to ensure that $ks + p < T$). Table 2 reports the resulting MSFE averaged over 100 simulations.

Table 2: Out of sample MSFE of one-step ahead forecasts averaged over 100 simulations (relative to Rolling Lasso-ARX)

Model	Relative MSFE	Standard Error
Rolling Lasso-ARX	1.0000	0.00263
Gradient Descent Adaptive Lasso-ARX	0.9507	0.00232
Newton Adaptive Lasso-ARX	1.2387	0.00749
Sample Mean	3.3243	0.00856
AR-X with lag order selected by AIC	20.682	0.02049
AR-X with lag order selected by BIC	20.682	0.02049

As expected, the Lasso procedures outperform the IC based approaches. The Gradient Descent based Adaptive Lasso achieves superior performance to the Rolling Lasso, whereas Adaptive Lasso that uses Newton's Method performs slightly worse than the Rolling Lasso.

Note that AIC and BIC achieve the same MSFE. This is likely due to the the inability of IC methods to impose a sparsity penalty that can accommodate the true underlying structure, resulting in both criteria choosing the smallest possible lag orders at every timepoint.

4.3 Data Application

We additionally evaluate our procedures on a set of quarterly US macroeconomic indicators procured from Koop Koop (2011). These series range from Quarter 2 of 1957 to Quarter 4 of 2007. We forecast the three series depicted in Figure 3: the US Gross Domestic Product growth rate (GDP), a measure of economic activity, the Consumer Price Index (CPI), a measure of inflation, and the Federal Funds Rate (FFR), a measure of monetary policy. Procedures that can accurately forecast these three series are of substantial interest to both macroeconomists and policymakers.

We consider forecasting these three series using their own past lags as well as the past lags of 19 related macroeconomic series as exogenous variables (see the *medium* model of Koop (2011) for the full list of included series). Following Banbura et al. Banbura et al. (2009), we set $p = s = 13$, resulting in 260 potential features. Quarter 3 of 1977 to Quarter 3 of 1992 is used for penalty parameter selection, while Quarter 4 of 1992 to Quarter 4 of 2007 is used for forecast evaluation. Our results are summarized in Table 3.

Table 3: Out of sample MSFE of one-step ahead forecasts of three macroeconomic indicators (relative to Rolling Lasso AR-X)

Model/Series	FFR RMSFE	CPI RMSFE	GDP RMSFE
Rolling Lasso AR-X	1.0000	1.0000	1.0000
Gradient Descent Adaptive Lasso AR-X	0.7008	0.6932	0.7479
Newton Adaptive Lasso AR-X	1.0081	1.0017	1.0053
Sample Mean	11.236	1.2033	2.1087
AR-X with lag selected by BIC	9.409	7.942	20.083
AR-X with lag selected by AIC	9.409	7.942	20.083

We find that for every series, the Gradient Descent Adaptive Lasso AR-X achieves the best forecasting performance, substantially improving upon the Rolling Lasso AR-X as well as upon the more naive approaches. Newton's Adaptive Lasso AR-X achieves very similar forecasts to the Rolling baseline model.

4.4 Feature Importance

In general, it is not possible to perform statistical inference on features recovered from the Lasso. The inclusion of the L_1 penalty makes it very difficult to accurately calculate degrees of freedom Lockhart et al. (2014). However, since the Adaptive Lasso computes an entire regularization path, in which features enter the active set sequentially, by examining the ordering of the active set recovered by the Adaptive Lasso, we can some gain insight as to the relative importance of certain features.

In our data application, this ordering can hold economic significance. For example, upon examining the active set when forecasting Gross Domestic Product and Consumer Price Index, we find that the most "important" exogenous features for both series include the Producer Price Index (PPI), which measures the average change in selling prices received by domestic producers for output. The PPI serves as a proxy for inflation, performing a very similar function to the CPI; in fact, in many European countries a causal relationship between the two indices has been established Akçay (2011). Hence, the PPI's inclusion as one of the most important features in forecasting CPI is justifiable economically. Similarly, Elliott & Timmermann (2013) find that including lagged values of PPI in forecasting equations can substantially aid in predicting GDP growth.

When forecasting the Federal Funds rate, the only feature included in the active set is the most recent lagged value of the Federal Funds Rate, providing evidence that the series closely follows a random walk. Hein & Spudeck (1988) tested this hypothesis and concluded that more complex models provided only slightly better forecasts than a random walk.

Potential future applications may involve performing an Adaptive Lasso procedure with the goal of feature recovery as opposed to forecasting.

5 Conclusion

Our initial results are encouraging, as we see that the Gradient Descent Adaptive Lasso AR-X substantially outperforms the conventional Rolling Lasso AR-X in both simulation and data applications, indicating that it can accurately forecast both stationary and nonstationary series. Additional analysis is required to determine the relatively poor performance of the Newton based Adaptive Lasso AR-X.

Our work still has considerable room for extensions. The procedures in this paper were constructed to optimize forecasting performance, but in other applications, such as empirically validating economic theories, feature recovery may be of primary interest. Under such a scenario, with an economically motivated loss function, we could potentially devise a more formal expert learning method that we can utilize to best approximate a model's most important features. Additional extensions include incorporating the online methodology for structured penalties, such as the Group Lasso Yuan & Lin (2006) or to a multivariate time series setting.

References

Akaike, Hirotugu. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

Akçay, Selçuk. The causal relationship between producer price index and consumer price index: Empirical evidence from selected european countries. *International Journal of Economics and Finance*, 3(6):p227, 2011.

Banbura, Marta, Giannone, Domenico, and Reichlin, Lucrezia. Large bayesian vector auto regressions. *Journal of Applied Econometrics*, 25(1):71–92, 2009.

Box, Jenkins and Jenkins, Gwilym M. Reinsel. time series analysis, forecasting and control, 1994.

Efron, Bradley, Hastie, Trevor, Johnstone, Iain, Tibshirani, Robert, et al. Least angle regression. *The Annals of statistics*, 32 (2):407–499, 2004.

Elliott, Graham and Timmermann, Allan. *Handbook of Economic Forecasting SET 2A-2B*. Elsevier, 2013.

Fama, Eugene F. The behavior of stock-market prices. *Journal of business*, pp. 34–105, 1965.

Friedman, Jerome, Hastie, Trevor, and Tibshirani, Rob. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

Garrigues, P. and El Ghaoui, L. An homotopy algorithm for the lasso with online observations. In *Proc. NIPS*, 2008.

Hein, Scott E and Spudeck, Raymond E. Forecasting the daily federal funds rate. *International Journal of Forecasting*, 4(4): 581–591, 1988.

Hsu, N. J., Hung, H. L., and Chang, Y. M. Subset selection for vector autoregressive processes using lasso. *Computational Statistics & Data Analysis*, 52(7):3645–3657, 2008.

Koop, Gary. Forecasting with medium and large bayesian vars. *Journal of Applied Econometrics*, 2011.

Litterman, Robert B. Techniques of forecasting using vector autoregressions. Working Papers 115, Federal Reserve Bank of Minneapolis, 1979. URL `http://ideas.repec.org/p/fip/fedmwp/115.html`.

Lockhart, Richard, Taylor, Jonathan, Tibshirani, Ryan J, and Tibshirani, Robert. A significance test for the lasso. *Annals of statistics*, 42(2):413, 2014.

Penm, Jack HW, Penm, Jammie H, and Terrell, RD. The recursive fitting of subset varx models. *Journal of Time Series Analysis*, 14(6):603–619, 1993.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015. URL `http://www.R-project.org/`.

Saha, Ankan and Tewari, Ambuj. On the finite time convergence of cyclic coordinate descent methods. *arXiv preprint arXiv:1005.2146*, 2010.

Schwarz, Gideon et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

Sherman, Jack and Morrison, Winifred J. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, pp. 124–127, 1950.

Shumway, Robert H and Stoffer, David S. *Time series analysis and its applications: with R examples*. Springer Science & Business Media, 2010.

Song, Song and Bickel, Peter. Large vector auto regressions. 2011. arXiv preprint arXiv:1106.3915.

Tibshirani, Robert. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Yuan, Ming and Lin, Yi. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.