Online optimization for variable selection in data streams

Christoforos Anagnostopoulos\textsuperscript{1, 2} and Dimitris Tasoulis\textsuperscript{2} and David J. Hand\textsuperscript{2, 3} and Niall M. Adams\textsuperscript{3}

Abstract. Variable selection for regression is a classical statistical problem, motivated by concerns that too many covariates invite overfitting. Existing approaches notably include a class of convex optimisation techniques, such as the Lasso algorithm. Such techniques are invariably reliant on assumptions that are unrealistic in streaming contexts, namely that the data is available off-line and the correlation structure is static. In this paper, we relax both these constraints, proposing for the first time an online implementation of the Lasso algorithm with exponential forgetting. We also optimise the model dimension and the speed of forgetting in an online manner, resulting in a fully automatic scheme. In simulations our scheme improves on recursive least squares in dynamic environments, while also featuring model discovery and changepoint detection capabilities.

1 Introduction

Regression models with many covariates can potentially capture more information about the response, but are also prone to overfit. Variable selection techniques have been used as a systematic way of negotiating this trade-off [11, 14] in the static, offline case. The need for such methods to carry over in the streaming data context has been recognised in the literature [6] but not yet systematically addressed.

Variable selection can lead to improvements in both parsimony and predictive performance [11]. Early methods would proceed in a greedy manner, iteratively adding maximally predictive covariates into the model, but suffered from local optima and lack of robustness [4]. More recent developments, such as the Lasso [15], have largely overcome these shortcomings via the use of convex, sparsity-inducing penalties. The weight of the penalty controls the model dimension and its optimal value can be estimated using general purpose model scoring techniques, such as the Akaike Information Criterion (AIC) [1], or more complex, tailored methods [17]. Several efficient algorithms have been proposed for the optimisation of Lasso-type problems (e.g., [15, 10, 8]). However, an online implementation had been lacking, so that only greedy heuristics have so far been employed in the streaming data case (e.g., [16]).

Another primary concern in the streaming context is adaptivity to drifting stream dynamics. An effective way to enforce adaptivity is to make past data decreasingly relevant to the estimation procedure via the use of forgetting factors [13]. In the case of linear regression, these forgetting procedures may be employed online and complemented by a gradient descent procedure that converges to the optimal rate of forgetting for fixed drift parameters. This complemented by a gradient descent procedure that converges to the

In case the true regression coefficients are time-varying, we can track them by iteratively scaling down the importance of past datapoints by a factor of $\lambda \leq 1$, where $\lambda = 1$ yields the static, OLS solution:

\[ Q_{t+1} = \sum_{i=1}^{t+1} \lambda^{t-i+1} X_i^T X_i = \lambda Q_t + X^T_{t+1} X_{t+1} \]
Since the update of the sample covariance is still of the same form, \( \beta_\text{LASSO}^t = P_t Q_t^{-1} \) may still be updated recursively via (2).

Setting the value of \( \lambda \) is analogous to choosing an optimal window size and can be done using standard techniques, such as cross-validation. However, the optimal value of \( \lambda \) may often be time-varying: for instance, if the drift features abrupt changes followed by intervals of no change, the value of \( \lambda \) should adapt accordingly, taking small values only at times of abrupt change. In [13], this is addressed via an adaptive scheme that at each timepoint moves \( \lambda \) in the direction that minimises RSS the most at the current timepoint:

\[
\lambda_{t+1} = \lambda_t + c_3 \text{sign}(\partial \text{RSS}/\partial \lambda_t)
\]

for some small constant \( c_3 \). The formula for the gradient is derived in [13, p.734] and is also \( O(p^2) \) so that, overall, RLS-AF is fully online.

### 2.3 The Lasso Estimator

The Lasso algorithm for variable selection [15] penalises the RSS by the sum of the absolute values of the learnt regression coefficients, i.e., the L1-norm of the regression vector:

\[
\beta^1 \leftarrow \text{argmin}_\beta \left\{ \sum_{i=1}^n (y_i - X_i \beta)^2 + \gamma \sum_{j=1}^p |\beta_j| \right\}
\]

It is a well-known fact that L1-norm penalties naturally favor sparse solutions [10], while retaining the convexity of the objective function. As a result, \( \beta^1 \) can be computed by a line search algorithm: for each \( j = 1, \ldots, p \), we minimise the penalized RSS with respect to \( \beta^{1,1}_j \) and iterate until convergence. The minimisation step is closed form:

\[
\beta^{1,1}_j \leftarrow \begin{cases} 
\text{sign}(\beta^{1,1}_j - S_j)(|\beta^{1,1}_j - S_j| - \gamma), & \text{if } |\beta^{1,1}_j - S_j| \geq \gamma \\
0, & \text{otherwise}
\end{cases}
\]

where \( S_j \) is the gradient of the penalised RSS with respect to \( \beta^{1,1}_j \), holding all other coefficients constant, and is given by:

\[
S_j = \partial \text{RSS}/\partial \beta^{1,1}_j = -0.5 \sum_{i=1}^n X_{ij}(y_i - X_i(\beta^{1,1})^T)
\]

This algorithm was first proposed as the shooting algorithm in [10].

Notably, the RHS of (7) expands to a formula that involves only the sample covariances, \( P \) and \( Q \). We are hence free to employ instead the forgetful versions, \( P_t \), \( Q_t \), obtained from RLS-AF. This allows us to hybridise the two algorithms, so that at each tick we perform one iteration of the shooting algorithm, initialised at the previous tick’s estimate of \( \beta^1 \), retaining an \( O(p^2) \) complexity per tick.

Note also that this algorithm assumes that the sample covariances have been standardised, so that rather than \( P_t \) and \( Q_t \), we have to work with their standardised versions, taking care to properly reweight \( \beta^1 \). This has no effect on computational complexity.

### 2.4 Learning the Dimensionality Online

There remains the issue of setting \( \gamma \), the shrinkage parameter featured in (5), which effectively controls the dimension of the learnt model. To do so we employ a standard model scoring criterion, the Akaike Information Criterion [1]. This penalises the RSS of a regression estimate \( \hat{\beta} \) by the number \( q \) of its non-zero coefficients:

\[
AIC(\hat{\beta}) = n \log \text{RSS} + 2q - n = n \log \left( R - 2P\hat{\beta}^T + \hat{\beta}Q\hat{\beta}^T \right) + 2q - n
\]

where we have rewritten the RSS in terms of the sample covariances \( P \) and \( Q \), as well as the sample variance \( R = y^T y \). Normally, \( n \) is equal to the sample size, but since we will be employing our forgetful versions \( P_t \), \( Q_t \), \( R_t \) the effective sample size at time \( t \) depends on previous forgetting factors:

\[
n(t) = \lambda_t + \lambda_{t-1} + \cdots + \sum_{i=1}^t \prod_{m=1}^i \lambda_m = \lambda_t(1 + n(t - 1))
\]

In our case, the candidate regression estimates which we wish to compare are Lasso solutions, \( \beta^1(\gamma) \), for various values of \( \gamma \). We hence wish to determine the value of \( \gamma \) that produces the Lasso solution with minimum AIC. Ideally, we should perform an exhaustive search over \( \gamma \), but at present we have no way of optimising this efficiently enough for an online scheme. Moreover, the relationship between \( \gamma \) and \( q \) is not closed form [8], so that gradient descent methods do not readily apply. Instead, as a first approximation, we perform a simple numerical minimisation at each tick, computing \( AIC(\beta^1(\gamma)) \) for \( \gamma \) equal to each of \( \gamma_t - c_2 \), \( \gamma_t + c_2 \), where \( c_2 \) is a small constant and \( \gamma_t \) our current estimate. We then use the value with the smallest AIC as our setting of \( \gamma_t+1 \).

Hybridising RLS-AF with the shooting algorithm and the numerical minimisation of AIC yields our algorithm, RLSASSO-AF, for online variable selection. Its complexity is dominated by RLS-AF and it is fully automatic, except for its dependence on initialised values and gradient descent steps (\( c_2 \) and \( c_3 \)).
3 Experimental Results

3.1 Description of the Simulation Engine

We test our method against simulated datasets \((y_t, X_t)\)\(_{t=1}^n\), letting
\[(y_t, X_t) \sim N(0, \Sigma_t)\]
where \(\Sigma_1, \ldots, \Sigma_n, \ldots\) is a sequence of dynamically changing \((p + 1) \times (p + 1)\) covariance matrices. The joint Gaussianity assumption is convenient in that it guarantees the required linear relationship between the response and the covariates, while also producing realistic variability in the covariates. We assume zero mean vectors for simplicity.

To generate the dynamic sequence of \(\Sigma_t\)‘s, we employed the simple and easy to control method described in [2]. We divide our timescale into regular intervals by introducing ‘changepoints’, \(t_1, \ldots, t_i, \ldots\). The distance between changepoints determines the ‘speed of the drift’. We then randomly generate symmetric, positive definite matrices \(Q_1, Q_2, \ldots\) to lie at each changepoint, setting \(\Sigma_t = Q_t\). In between, for \(t \in (t_i, t_{i+1})\), we set:
\[\Sigma_t = \mu Q_t + (1 - \mu)Q_{i+1}, \quad \text{with } \mu = \frac{t_{i+1} - t}{t_{i+1} - t_i}\]  
(9)

This turns the difficult problem of generating smoothly changing covariance matrices to the easy problem of generating arbitrary covariance matrices. Moreover, it allows us to control the true sparsity of the regression relationship: for each changepoint, we randomly generate the \(p \times p\) covariance of the covariates and, separately, a \(p \times 1\) vector of regression coefficients of a specified sparsity. We can then use standard formulas to obtain a joint covariance \(Q_t\).

We also considered abrupt change, setting \(\mu = 1\) in (9).

Note that all the underlying covariance matrices were chosen to be standardised, so as to retain our sense of scale despite the changes in the correlation structure. We have repeated our experiments with non-standardised covariances and have found that our qualitative findings persist, although lack of standardisation introduces appreciably more variability in the error sequence across all settings.

3.2 Results

We start by discussing performance in the static case to get some first insights into the problem. We measure performance by the mean squared residual error, \((y_t - X_t \beta)^2\), averaged over time as well as over several streams of varying dimensionality. The results are reported in Figure 1, where we compare RLS-AF with RLASSO-AF, as well as its offline version where \(\gamma\) is optimally set via exhaustive search. We immediately observe that online RLASSO-AF is outperformed by RLS-AF for all values of \(p\). However, this effect can be exclusively attributed to the difficulties of learning the shrinkage parameter online: when \(\gamma\) is optimally selected, our estimator outperforms RLS-AF by a margin that grows wider with \(p\).

These results are consistent with theory. In the static case, our scheme is guaranteed for any fixed setting of \(\gamma\) to converge to the respective Lasso estimator, which is known to be superior to the OLS estimate when \(\gamma\) is suitably set, in particular when the true model is sparse and \(p\) large [15]; we can outperform RLS-AF if we optimally select \(\gamma\). However, making this selection online is not expected to be easy. First, closed-form model scoring criteria, such as the AIC, do not always perform well in practice as they rest on several unrealistic assumptions [4], so that their suitability cannot be guaranteed. Second, our adaptive scheme only approximately minimises AIC. These shortcomings can be expected to dominate when there is little or no covariance drift, while least squares estimation enjoys ever increasing sample sizes and eventually ceases to overfit, making its performance hard to beat.

In contrast, in the dynamic case where the least squares fit is imperfect, \(\gamma\) need not be as fine-tuned for our scheme to reap the benefits of variable selection. In Figure 2, we demonstrate via a surface plot that the relative improvement in performance of RLASSO-AF over RLS-AF depends on the dimension and true sparsity ratio of the underlying stream. Results are averaged over several streams of length 1000, featuring changepoints every 100 points, for each of a range of settings for the stream dimension and sparsity. The plot suggests that RLASSO-AF outperforms RLS-AF more and more as \(p\) grows, when the covariance structure is changing, regardless of whether the true model is sparse or not. In fact, relative performance does not seem to be affected strongly by the sparsity of the true regression model: our adaptive learning of \(\gamma\) seems not to be sophisticated enough to sufficiently exploit sparsity in the true model, suggesting room for improvement.
We now move to describe the adaptive characteristics of our algorithm. This behavior is best illustrated against a single stream, but persists across a variety of settings, as explained later. Our chosen testbed is a stream of length \( n = 2000 \), featuring an abrupt change every 200 timepoints. The true regression relationships are 50% sparse: half the covariates are randomly selected at each changepoint to be set to exact zeros. The total number of covariates is 50.

Our results against this testbed are summarised in Figure 3. First, we confirm the findings of [13] in immediately noticing that the forgetting factor adapts brilliantly in most cases (Figure 3, Plot III): shortly after a changepoint occurs, \( \lambda \) begins to drop fast, allowing the algorithm to quickly forget data prior to the changepoint and re-estimate the sample covariances. An easy computation (that of equation (8)) shows that, when \( \lambda \) is at its lowest, the effective sample size drops to a mere 10 datapoints, an impressively short memory given that we are estimating a 50 \( \times \) 50 covariance matrix. After a short interval, \( \lambda \) increases again making available a longer memory, in recognition of the fact that any irrelevant data have by now been forgotten and the process is once again constant.

Second, we confirm that the Lasso predictions are more precise and less volatile than the RLS-AF predictions at and around the changepoints (shown as smaller peaks in Figure 3, IV), yielding an overall 155% boost in performance. Since this stream features abrupt changes, the covariance is static in between changepoints so that RLS-AF catches up with RLASSO-AF. Indeed, switching to smooth changes removes these static periods yielding further advantage to RLASSO-AF and raising the relative performance to 161% (plot not included for lack of space). Beyond predictive performance, we also gain appreciably in model discovery capabilities: in Figure 4 we can clearly observe that whenever a covariate suddenly becomes inactive, our algorithm will prune it fairly consistently shortly afterwards.

Finally, we turn our attention to the adaptive behavior of the shrinkage parameter. As we indicated earlier, our adaptive routine, despite its shortcomings, manages to learn \( \gamma \) well enough for fast-paced environments, where there is enough to gain by variable selection so that fine-tuning is not necessary. This is already in some sense important, as it makes our algorithm practically applicable. However, the adaptive behavior is very interesting in its own right, as it reveals a complicated dependency of AIC to the shrinkage parameter, the residual error and the forgetting factor. To fully address this interplay lies beyond the scope of the current work and is our main research aim for the future. We take, however, the opportunity to introduce some key issues.

Perhaps surprisingly, our adaptive algorithm does not spot a single reasonable value for \( \gamma \). Instead, as depicted in Plot I of Figure 3, it seems to react to change in a consistent, repeatable manner, peaking shortly after changepoints. This can be attributed to the strong dependence of AIC to the residual error, as is suggested by plotting the cross-correlation between \( \gamma \) and the residual error (Figure 5), averaged over several samples from the same stream characteristics. This is evidence that, at times of crisis, our algorithm tends to prune more violently, in an attempt to contain the error. Still, as shown in Figure 3, Plot II, the actual number of variables pruned is on average much more closely than it does the residual error. Indeed, a cross-correlation plot averaged over several streams confirms this (Figure 6). This strong correlation can be explained by the dependence of AIC on both the effective sample size (a function of \( \lambda \)) and, of course, the quality of the Lasso estimates, which also increases with \( \lambda \).

Overall, a complicated picture is painted involving lagged feedback mechanisms. At times of stability, AIC tends to favor sparser models, since the increasing quality of the Lasso estimates offers ‘more for less’, in particular since the true, underlying regression model is sparse. At times of change, however, the reaction of the algorithm is more complex. The sudden increase in residual error causes \( \gamma \) to temporarily shoot up, shrinking and pruning away coefficients so as to contain the error. Shortly afterwards the forgetting factor drops in response and \( \gamma \) follows. This dependence of \( \gamma \) on \( \lambda \) is

![Figure 3](image_url)  
Figure 3. A plot of the learnt shrinkage parameter (Plot I), the learnt model dimension (Figure 3, Plot II), the learnt forgetting factor (Figure 3, Plot III) and the residual errors of both RLS-AF and RLASSO-AF against time. The underlying simulation utilises a stream featuring abrupt changes every 200 timepoints and 50% sparse true regression models.

![Figure 4](image_url)  
Figure 4. Plot of five randomly selected true regression coefficients stacked against their respective learnt Lasso estimates, against time. Black indicates an exact 0, white indicating values away from 0. The underlying simulation utilises a stream featuring abrupt changes every 200 timepoints and 50% sparse true regression models.
in fact the key relationship of interest and deserves further study. In particular, it could perhaps be exploited to improve our algorithm by way of a joint optimisation step for $\gamma$ and $\lambda$.

![Figure 5](image1.png)

Figure 5. Plot of the sample cross-correlation between the number of variables pruned at each timepoint and the value of the forgetting factor.

![Figure 6](image2.png)

Figure 6. Plot of the sample cross-correlation between the shrinkage parameter and the value of the forgetting factor.

To conclude, we remark that our algorithm is somewhat sensitive to the size of the gradient steps, $c_\delta$, and $c_\gamma$, although to a much smaller extent than to the underlying quantities optimised, $\lambda$ and $\gamma$.

4 Conclusion

Variable selection can lead to improvements in predictive performance in the static case, but has so far not been sufficiently exploited in the streaming context, for lack of online implementations of standard selection algorithms. We propose an online optimisation scheme that implements the Lasso algorithm, converging to the optimal solution in the static case. We hybridise it with recursive least squares adaptive forgetting to allow it to track changing covariance structures. We also estimate the optimal dimensionality via an algorithmic framework that implements the Lasso algorithm, converging to the optimal subset selection, so that we may investigate in a principled manner the interplay between forgetting, choice of focus and change. Our results open up an exciting research direction that raises novel questions about the nature of learning in dynamic environments.

ACKNOWLEDGEMENTS

This work was undertaken as part of the ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Systems) project and is jointly funded by a BAE Systems and the EPSRC (Engineering and Physical Research Council) strategic partnership, under EPSRC grant EP/C548 051/1. The work of David Hand was partially supported by a Royal Society Wolfson Research Merit Award.

REFERENCES