

# Simulating dynamic covariance structures for testing the adaptive behavior of variable selection algorithms.

Christoforos Anagnostopoulos  
Maths Department, Imperial College London,  
South Kensington, London W7 2AZ, UK

Niall Adams  
Maths Department, Imperial College London,  
South Kensington, London W7 2AZ, UK

## Abstract

*Variable selection for regression is a classical statistical problem, motivated by concerns that too large a number of covariates may bring about overfitting and unnecessarily high measurement costs. Novel difficulties arise in streaming contexts, where the correlation structure of the process may be drifting, in which case it must be constantly tracked so that selections may be revised accordingly. A particularly interesting phenomenon is that non-selected covariates become missing variables, inducing bias on subsequent decisions. This raises an intricate exploration-exploitation tradeoff, which we explore in simulations, revealing its dependence on properties of the covariance tracking algorithm and the choice of variable selection scheme. We also take the opportunity to tackle the difficult problem of simulating dynamic correlation structures.*

## 1. Introduction

Variable selection is a ubiquitous problem in statistical modelling, motivated by concerns that too large a number of covariates may bring about overfitting and unnecessarily large measurement costs [7]. Although existing algorithms are tailored to the static case, it was shown in [1] that the dynamic case may be accommodated via the combined use of covariance tracking algorithms and fast static selection algorithms, such as the Lasso [13], giving rise to schemes that revise their selections to adapt to the drift.

Covariance tracking is a notoriously difficult problem. Existing approaches that explicitly model covariance drift include GARCH, Stochastic Volatility and Switching State-Space Models (see [10] for a review), but invariably scale poorly with the number of covariates in online contexts. An alternative approach, which we take here, is not to have a generative model of covariance drift, but rather assume the structure to be locally constant, dealing with change by ‘forgetting’ past data, i.e., via a sliding window or the use of forgetting factors [15]. Also, in the absence of an explicit

model, we are free to explore the problem of generating drifting covariance sequences, in simulations, separately to the theoretical subtleties of learning its parameters.

Variable selection is also a difficult problem that has received little to no attention in the context of drifting correlation structures. In [1], Adaptive Variable Selection for Regression (AVS-R) was discussed with emphasis on retaining high predictive ability using a small proportion of the available covariates. Interestingly, in such a scenario, only variables that are selected are observed, whereas non-selected covariates become missing variables, which induces bias upon subsequent decisions. This complication, known in the multi-arm bandit literature as ‘opaque’ versus ‘transparent’ decision-making [14], introduces a novel type of exploration-exploitation dilemma with links to Reinforcement Learning [12], with immediate relevance to several application areas, such as medical monitoring for adaptive trials [8], adaptive query processing in battery-powered sensor networks [5, 9] and other types of real-time decision support.

In this article, we attempt to further quantify and measure this exploration-exploitation tradeoff, extending the range of selection algorithms considered in [1]. A prerequisite to such an analysis was to provide a versatile, stable engine to simulate high-dimensional dynamic correlation structures, which we took great care to design. We first describe a framework for defining and testing AVS-R schemes, then proceeding to describe our simulation engine in Section 3. We finally report our results, with analysis, in Section 4.

## 2. AVS-R Framework

We consider datastreams of the form:

$$(y_1, X_1), \dots, (y_{t-1}, X_{t-1}), (y_t, X_t), \dots$$

where  $X_t$  is a  $p$ -dimensional vector of regressors and  $y_t$  a univariate response. In AVS-R, our aim is, at each timepoint  $t$ , to select which  $q < p$  regressors we should observe next

so as to be able to predict our response variable with minimal error. We encode this selection by  $\gamma_{t+1}$  and the partial observation that results from this selection by  $X_{t+1}^{(\gamma_{t+1})}$ , which we then use to form our prediction of the response at time  $t + 1$ ,  $\hat{y}_{t+1}$ . For the purposes of this study, we represent the stream dynamics by a sequence of estimates  $(\hat{\mu}_t, \hat{\Sigma}_t)$  of the means and covariance of the joint distribution of  $(y_t, X_t)$ , defining AVS-R schemes in terms of the following three *modules*:

Tracking (or Estimation):  $(\hat{\mu}^{(t)}, \hat{\Sigma}^{(t)}) = g(H_{[1,t-1]}^\gamma)$

Selection:  $\gamma_t = S(\hat{\mu}^{(t)}, \hat{\Sigma}^{(t)})$

Regression:  $\hat{y}_t = f(\hat{\mu}^{(t)}, \hat{\Sigma}^{(t)}, X_t^{(\gamma_t)})$

where  $H_{[i,j]}^\gamma = (y_i, X_i^{(\gamma_i)}), \dots, (y_{j-1}, X_{j-1}^{(\gamma_{j-1})}), (y_j, X_j^{(\gamma_j)})$  denotes an *opaque* history of partial observations.

## 2.1 Tracking the Mean and Covariance

As we mentioned in the introduction, in the current work we do not model covariance drift explicitly, but rather deal with it by discarding past data: at time  $t$ , we restrict our attention to the  $l$  most recent datapoints, which we assume to be i.i.d. samples from a multivariate Gaussian. This approach allows us fast, stable learning, which is essential in a first attempt to investigate a novel problem. We can then deal with opacity by approximating the maximum likelihood estimates for  $\mu^{(t)}$  and  $\Sigma^{(t)}$  via the EM algorithm [4]. This algorithm employs an iterative update step, starting at some user-specified initial estimate of the parameters,  $\theta(0)$  and each time choosing  $\theta(i+1)$  to maximise the expected log-likelihood of the observed data, where the expectation is taken over the missing data with respect to the current estimate  $\theta(i)$ . To ensure faster convergence and make use of previous computations, we initialise the EM algorithm at each timepoint using the previous window's estimates.

Note that the size  $l$  of our 'sliding window' approach is an important quantity: too large a window challenges our 'locally constant correlations' assumption as well as our mean tracking algorithm, whereas too small a window invites overfitting. Here, we control this parameter offline, with a view to fitting it to the data in future work.

## 2.2 Selection and Regression

It is an easy result that the sample covariance between a response  $y$  and a vector of covariates  $X$  is a sufficient statistic to generate the Ordinary Least Squares (OLS) estimate of the regression coefficients of  $y$  regressed on  $X$ :

$$\beta^{\text{OLS}} = \Sigma_{yX} \Sigma_{XX}^{-1}$$

Similarly, we may regress  $y$  on any subset  $\gamma$  of the covariates, by considering the respective submatrices,  $\Sigma_{yX^{(\gamma)}}$  and

$\Sigma_{X^{(\gamma)}X^{(\gamma)}}$ , to obtain  $\beta^{\text{OLS}}(\gamma)$ . In fact, it turns out also that a wide range of variable selection algorithms for linear regression can be computed solely on the basis of the sample mean and covariance. This observation underlies our choice of representation and framework, as it allows the deployment of existing selection techniques in spite of the presence of covariance drift and missing data.

It is customary in the literature to break down variable selection in terms of, on the one hand, learning an optimal subset of covariates for each dimensionality (i.e., number of covariates) and, on the other, selecting the optimal dimensionality [2]. This can be particularly useful in contexts where the dimensionality is controlled externally, e.g., by communication bandwidth constraints or measurement costs. In the current work, we fix the dimensionality offline, with a view to learning it from the data in future work.

Learning the optimal  $q$ -subset of covariates is still a combinatorially hard problem. In [1], the authors focus their attention to the *Lasso* [13] algorithm, an approach that escapes the combinatorial search implicit in most selection algorithms by penalising the absolute values of the regression coefficients (convex) rather than the number of non-zero entries (non-convex). Here we extend the range of selection algorithms tested, overall considering the following:

- *Best Subsets*: look for the subset  $\gamma$  that best predicts the response in the OLS sense, i.e., with minimal residual variance, as computed from the current estimate  $\hat{\Sigma}$ :

$$\gamma^{\text{Best}} \leftarrow \underset{\arg\min}{\gamma} \left\{ \hat{\Sigma}_{yy} - \hat{\Sigma}_{yX^{(\gamma)}} \hat{\Sigma}_{X^{(\gamma)}X^{(\gamma)}}^{-1} \hat{\Sigma}_{X^{(\gamma)}y} \right\}$$

- *Max Abs Reg*: select the covariates whose OLS regression coefficients are maximal in absolute value.
- *Max Abs Cov*: select the covariates whose estimated covariance with  $y$  is maximal in absolute value.
- *Random*: select uniformly at random.
- *Elastic Net*: use the algorithm of [6] to learn a regression vector  $\beta$  of dimensionality  $q$  (i.e., with  $p - q$  zero entries) that minimises:

$$(\beta - \hat{\beta}^{\text{OLS}})' \Sigma_{XX} (\beta - \hat{\beta}^{\text{OLS}}) + \lambda \sum_{j=1}^p |\beta_j| + \nu \sum_{j=1}^p \beta_j^2$$

Note that  $\nu = 0$  yields the Lasso.

The hyperparameter  $\lambda$  in the Elastic Net is uniquely determined by the choice of dimensionality. In contrast, the additional shrinkage parameter  $\nu$  is free and is here controlled offline. We have amended the Matlab implementation of [11] for the Elastic Net selection to take as input a covariance matrix, rather than a dataset.

## 2.3 Stickiness and Adaptive Behavior

The effect of previous selections on current selections, mediated via our choice of initialisation, can under certain circumstances introduce a feedback loop: uncorrelated predictors cease being observed; they are then assumed to remain uncorrelated and are hence not revisited. This is because the EM algorithm will only update any covariance entries,  $\Sigma_{i,j}$ , for which variables  $x_i, x_j$  have been jointly observed at least once in the training dataset (in our case, the window), leaving the rest equal to their initialised values. Our scheme can therefore be expected to be conservatively ‘sticky’: once it locates a good subset, it will be reluctant to move away from it, unless its performance deteriorates.

Although it is clear that stickiness can lead to sub-optimal behavior, it is less clear how to move towards more efficient schemes, because the interplay between selection, estimation and reward is very different in our problem than in the classical Markov Decision setting of Reinforcement Learning [14]: to name but a few of the differences, we have exponentially many ‘arms’ (= subsets), they are not independent, stationarity is violated and, perhaps most importantly, our reward (= prediction error) is affected by our decisions *only* via the quality of estimation, not via direct intervention. As a result, the performance of even the most classical algorithms has to be re-assessed via simulations in this context. We make a first step in this direction in this paper by testing alongside our AVS-R schemes their  $\epsilon$ -greedy variants, whereby each scheme performs its chosen selection at a frequency of  $1 - \epsilon$ , while also selecting randomly at a frequency of  $\epsilon$ . The selection at the exploration step is random for *all* covariates in the current work, although it is our intention to explore in future work variants with mixed exploration steps (e.g., only replacing the  $d < q$  ‘weakest’ covariates with random selections).

On a related strand, it is also important to realise that the degree of stickiness can be safely expected to vary with the selection scheme. For instance, random selections lie at the extreme of exploratory behavior and a statically fixed selection lies at the other extreme. It is much harder however to rate non-trivial selection schemes with confidence. Our simulations provide a first glimpse into this.

## 3. Simulating Dynamic Covariance Structures

In accordance with the previous discussion, we need to design a simulation engine whereby each data instance is sampled from a multivariate,  $m$ -dimensional Gaussian, for  $m = p + 1$ :

$$(y_i, X_i) \sim N(\mu^{(i)}, \Sigma^{(i)})$$

The objective then is to define some simple stochastic process on a certain chosen representation of the  $\mu^{(t)}$ ’s and

the  $\Sigma^{(t)}$ ’s. Whereas it is easy to define various kinds of mean movement, a random walk being the canonical choice, it is much harder to construct a well-behaved sequence of changing covariances. Besides retaining positive definiteness and invertibility (both in theory and practice), a whole range of statistically important quantities derivable from the  $\Sigma^{(t)}$  must also behave reasonably, including the entries of the covariance matrix  $\Sigma_{ij}^{(t)}$  themselves; the correlations  $\rho_{ij}^{(t)} = \Sigma_{ij}^{(t)} / \sqrt{\Sigma_{ii}^{(t)} \Sigma_{jj}^{(t)}}$ ; the partial correlations  $-1/k_{ij}^{(t)}$  (where  $K^{(t)} = (\Sigma^{(t)})^{-1}$ ); the regression coefficients of the response to the covariates,  $\beta_{y|X}^{(t)}$ ; as well as the residual variance of the response when regressed on the covariates,  $\sigma_{y|X}^{(t)}$ . Notably, there are several non-linear dependencies among these variables, so that linear movement in any chosen parameterisation of  $\Sigma^{(t)}$  is bound to non-linearly affect several other quantities of interest, often causing their values to explode, collapse or become overly volatile. Consequently, theoretical arguments that readily guarantee a bounded movement in one of these quantities must be backed up by extensive simulations and fine-tuning if we are to control all of them simultaneously.

There is a wide selection of parameterisations of the covariance matrix to consider (e.g., [3]). A particularly appealing one is the Cholesky factor decomposition, as perturbations of the off-diagonal entries of the Cholesky factor retain positive definiteness. In particular, we may try a random walk:

$$\forall i > j, A_{ij}^{(t+1)} = cA_{ij}^{(t)} + \epsilon$$

where  $\epsilon \sim N(0, \sigma^2)$  with  $c \approx 1$  and  $\sigma^2$  small. Variations on this theme would include enforcing a random walk on the off-diagonal entries of the Cholesky factor of  $(\Sigma^{(t)})^{-1}$  or perturbing in this manner the covariance of the covariates only,  $\Sigma_{XX}^{(t)}$  (or its inverse), while controlling explicitly the regression coefficients of the covariates with the response.

We considered all four choices, as well as an Inverse Wishart random walk, as described in [10], plotting all quantities of interest (e.g., Figure 3) over a timescale of about  $1000m$ , for various values of  $m$  and settings of the damping factors and standard deviations of the random walks. No satisfactory solution emerged: the correlations as a function of  $c$  and  $\sigma$  jumped from being overly explosive (producing badly scaled near-singular matrices) to being collapsive (whereby they all quickly became contained within a very small neighborhood around 0). Attempting to control the regression coefficients separately, by imposing a separate random walk on them, aggravated singularity problems. In response, we considered *centering* the random walk around a non-zero value:

$$\forall i > j, A_{ij}^{(t+1)} = A_{ij}^{(0)} + \eta^{(t+1)}$$

where  $\eta^{(t+1)} = c\eta^{(t)} + \epsilon$ ,  $\epsilon \sim N(0, \sigma^2)$ . This choice

demonstrated satisfactory volatility but produced the undesirable artefact of having larger entries further down the diagonal of  $\Sigma^{(t)}$  (since these entries were obtained via multiplying and adding together more terms of  $A^{(t)}$  than the entries in the upper left corner of  $\Sigma^{(t)}$ ). We corrected this to a reasonable extent (though not entirely) by scaling the variances of the random steps:

$$\eta^{(t+1)} = c\eta^{(t)} + \epsilon, \epsilon \sim N(0, e^{-(i+j)}\sigma^2)$$

Certain typical properties of the resulting scheme, which we call ‘Cholesky Random Walk’, are demonstrated in Figure 1. The behavior of the engine is controlled to a satisfactory level but still tends to produce ‘non-generic’ covariance matrices in some respects: the residual error stabilises around a very small value, the condition of the matrix oscillates, the regression coefficients jump and so on. We therefore also considered a much more simplistic and easier to control generation engine, which exploits the fact that a convex sum of two positive definite matrices is positive definite. In this scheme, we divide our timescale into regular intervals by introducing ‘changepoints’,  $t_1, \dots, t_i, \dots$ . The distance between changepoints determines the ‘speed of the drift’. We then fix a sequence of distinct, randomly generated covariance matrices  $Q_1, Q_2, \dots$  to lie at each changepoint, setting  $\Sigma^{(t_i)} = Q_i$ . In between, for  $t \in (t_i, t_j)$ , we set:

$$\Sigma^{(t)} = \lambda Q_i + (1 - \lambda)Q_j, \text{ with } \lambda = \frac{t_j - t}{t_j - t_i}$$

The resulting scheme is much more stable (Figure 2). In simulations, however, we preferred to use Cholesky Random Walk, as it seemed to challenge our AVS-R schemes more. We also kept  $\mu^{(t)} = 0$  constant so as to observe in isolation the difficulties of tracking covariance movement.

## 4. Simulated AVS-R Experiments

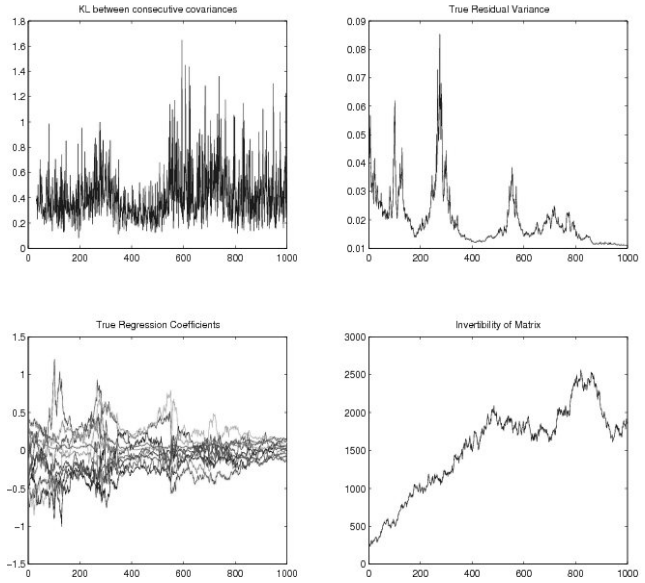
### 4.1 Measuring Performance

Performance in this context for a given dataset is naturally measured by prediction error per tick:

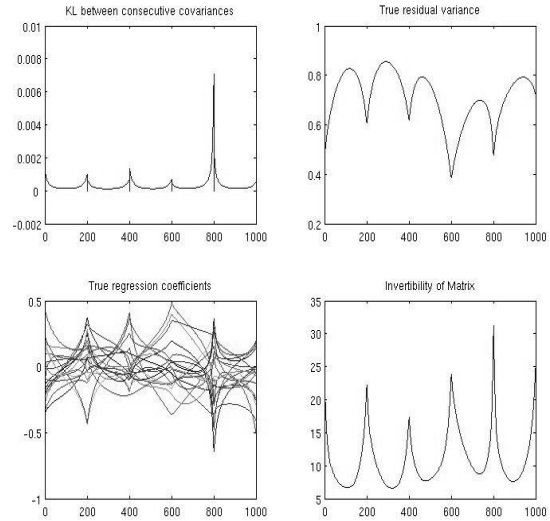
$$(\hat{y}_1 - y_1)^2, \dots, (\hat{y}_t - y_t)^2, \dots$$

(Recall that  $y_t$  is the response at time  $t$  and  $\hat{y}_t$  is our estimate of it on the basis of selected observations from the vector of regressors  $X_t$ ). It is however difficult to draw conclusions from this highly volatile sequence of numbers. To do so, we follow [1] in employing a combination of *averaging* and *benchmarking*.

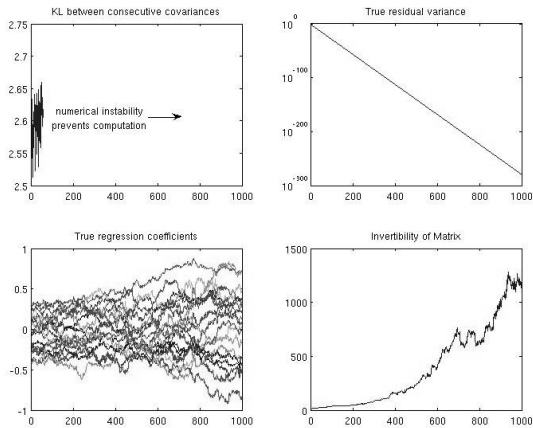
For each scheme, we form a *sample average*, over several datastreams generated from the same covariance process, as well as a *time average*. Moreover, to capture the additional variability of schemes that involve random choice,



**Figure 1. For 1000 covariances following a Cholesky random walk, we plot the Kullback-Leibler divergence between  $\Sigma^{(t+1)}$  and  $\Sigma^{(t)}$ ; the residual variance of  $y$  regressed on  $X$ ; the value of the respective regression coefficients; and finally the condition of  $\Sigma^{(t)}$ .**



**Figure 2. As in Figure 1, using instead a Convex Walk with changepoints every 200 ticks.**



**Figure 3.** As in Figures 1 and 2, employing an Inverse Wishart Random walk.

we average over several different outcomes of such choices as well, as part of forming the sample average per tick. In addition to averaging, we also assess relative performance against benchmarks. This allows us to separate out several distinct sources of error: sample uncertainty, changing stream dynamics, cumulative selection bias, local minima in the EM procedure and so on. We hence contrast *opaque* estimation, which uses EM on the incomplete window  $H_{[t-l, t-1]}^\gamma$  to *transparent estimation*, which forms the maximum likelihood estimator using the complete window  $H_{[t-l, t-1]}$  and *oracle estimation*, which uses the true parameters  $(\mu^{(t)}, \Sigma^{(t)})$ . Finally, we normalise overall performances by dividing them with the total variance in the response, hence obtaining a measure of the *percentage of total variance explained* by each scheme. This is equivalent to benchmarking against *mean imputation* of the response, where no covariates are used at all.

## 4.2 Measuring Stickiness

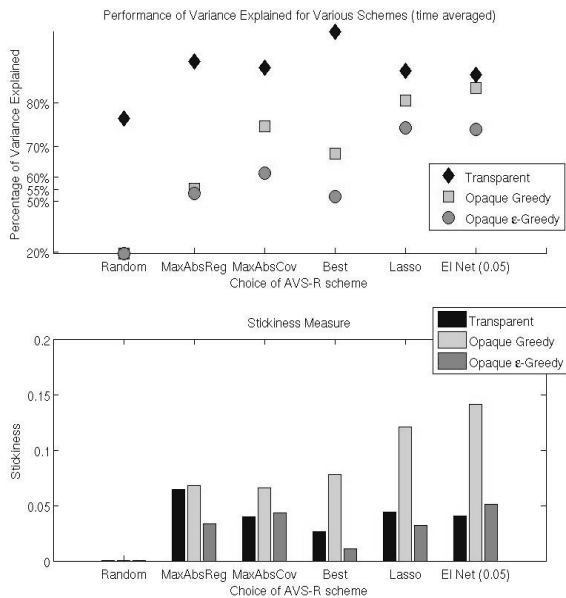
Stickiness is even harder to quantify than predictive error. In [1] the authors plot the frequencies at which each covariate was selected, to observe that under opaque learning their values are either close to 1 or to 0, whereas under transparent learning they attain mostly mid-range values. This indicates that, under opaque learning, schemes tend to focus on certain covariates, while disregarding others. However, whereas in [1] a qualitative, visual comparison is offered, in the current work we explicitly quantify this effect, by measuring the *variance of the observation frequencies* and comparing its values under opaque and transparent learning respectively. The latter comparison is essential, as otherwise it is impossible to tell whether lack of movement reflects

overexploitative behavior or local stagnancy of the covariance drift. We additionally normalise these quantities using random selections as benchmark. Our resulting measure is robust across different simulations and interpretable.

## 4.3 Results with Discussion

In Figure 4, the plot on the top shows the (normalised) performance of each of our selection schemes, under transparent, greedy opaque and  $\epsilon$ -greedy opaque learning respectively. For the underlying simulations, the dimensionality of the model was fixed at  $q = 7$  out of a total of 15 covariates. We kept the number of covariates low, so that we may compare convex selection schemes to the exponentially slow Best Subsets selection scheme. We may observe that a much higher predictive performance may be attained using appropriate selection schemes in comparison to random selections. Indeed, Elastic Net with  $\nu = 0.05$  achieves comparable performance under opaque than under transparent learning, a remarkable result since the former employs less than 50% of the data available to the latter. Equally, it is an important result that the Elastic Net performs better than Best Subsets selection in the Opaque context. This can be attributed to the latter’s sensitivity to poor estimation — indeed, in the transparent setting, where the estimation task is easier, Best Subsets outperforms all other techniques.

We now turn to dynamic behavior. We can observe in the top part of Figure 4 that  $\epsilon$ -greedy performs in all cases worse than greedy. This is a striking result, since it is customarily the case that  $\epsilon$ -greedy boosts performance, since it also takes into account information gain [12]. In our case, however, this no longer holds. A tentative explanation for this is the following: random selections force the EM algorithm to recompute several entries of the covariance matrix on the basis of a very small sample, which adversely affects the quality of the estimand: in other words, the value of information can be under certain conditions dramatically depreciated due to the difficulties of the estimation task. This readily suggests a direction for future work, namely to explore variants to  $\epsilon$ -greedy learning where the occasional exploration steps are mixed, or perhaps guided by a second variable selection algorithm, rather than random selection. It remains however a valid and important conclusion that stickiness seems to potentially also be of value in the sense of information gain, as it allows the scheme to focus in on a small part of covariate space, which it becomes capable of estimating well. This is also confirmed by the fact that the loss in performance when moving from opaque to transparent learning is very large for schemes that are too exploitative but becomes smaller as schemes become more sticky — this can be visually confirmed in Figure 4 by considering, in sequence, the relative loss in performance of random, best subsets, lasso and elastic net selections.



**Figure 4. A plot of the performance (top) and stickiness (bottom) of several selections schemes under transparent, greedy opaque and  $\epsilon$ -greedy opaque estimation.**

## 5 Conclusion

In [1], the authors establish a versatile framework for defining and testing schemes that perform Adaptive Variable Selection for Regression and outline the need for further study into the exploration-exploitation tradeoff implicit in opaque decision-making. We extend that work by simulating dynamically changing covariance matrices, which we then use to generate datastreams. We test a wide variety of selection schemes against this data, confirming the possibility of retaining high predictive accuracy with less data and investigating each scheme’s adaptivity to the covariance drift. We observe a strong correlation between robustness and stickiness. A striking result is that  $\epsilon$ -greedy learning is outperformed by greedy, which can be explained by the fact that stickiness in this context is also of information value, as it allows the scheme to focus in on a small part of the space that it becomes capable of estimating well. As future work, we intend to consider variations to  $\epsilon$ -greedy learning that do not affect as adversely the stability of the estimation module, as well as to explore alternative ideas from Reinforcement Learning, via simulations and theoretical analysis. Another important challenge will be to allow the dimensionality to be adaptively learnt online, which will put much extra strain on the already difficult task of quanti-

fying and optimising stickiness.

## 6 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.

## References

- [1] C. Anagnostopoulos, N. M. Adams, and D. J. Hand. Deciding what to observe next: adaptive variable selection for regression in multivariate data streams. *to appear in Proceedings of ACM Symposium on Applied Computing*, 2, 2008.
- [2] L. Breiman. The little bootstrap and other methods for dimensionality selection in regression: X-fixed prediction error. *Journal of the American Statistical Association*, 87(419):738–754, 1992.
- [3] T. Chiu, T. Leonard, and K. Tsui. The matrix-logarithmic covariance model. *Journal of the American Statistical Association*, 91, 1996.
- [4] A. Dempster, N. Laird, and D. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
- [5] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. *Proceedings of the 30th VLDB Conference*, 2004.
- [6] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2):407–499, 2004.
- [7] E. I. George. The variable selection problem. *Journal of the American Statistical Association*, 95(452):1304–1308, 2000.
- [8] L. Gunter, J. Zhu, and S. Murphy. Variable selection for optimal decision making. *Proceedings of the 11th Conference on Artificial Intelligence in Medicine*, 2007.
- [9] J. Hellerstein, M. Franklin, S. Chandrasekaran, A. Deshpande, K. Hildrum, S. Madden, V. Raman, and M. Shah. Adaptive Query Processing: Technology in Evolution. *IEEE Data Engineering Bulletin*, 23(2):7–18, 2000.
- [10] K. Platanioti. Estimation methods for univariate stochastic volatility models and multivariate stochastic volatility models. Technical report, Imperial College London, 2005.
- [11] K. Sjöstrand. Matlab implementation of LASSO, LARS, the elastic net and SPCA, 2005. Ver 2.0.
- [12] R. Sutton. *Reinforcement Learning*. Springer, 1992.
- [13] R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.
- [14] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. *Proceedings of the 16th European Conference on Machine Learning*, pages 437–448, 2005.
- [15] B.-K. Yi, N. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, and A. Biliris. Online data mining for co-evolving time sequences. pages 13–22, 2000.