ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Optimal Training of the Online Newton Algorithm based on Conformal CUSUM

Thomas Grava and Frédéric Vrins

LIDAM/LFIN, UCLouvain, Belgium

Abstract. We apply conformal CUSUM to identify the training period of the Online Newton Algorithm in a continuous learning setup. Numerical simulations demonstrate the ability of our approach, called Lean-ONS, to efficiently detect change-points and identify a relevant training set, which improves prediction performance compared to standard alternatives in the literature.

1 Introduction

Regression consists of fitting a model $\hat{g}$ in order to minimize the error between the observed outputs y_i and the predictions $\hat{y}_i := \hat{g}(X_i)$, $i = 1, 2, \ldots$ exploiting pairs (y_i, X_i) in a training set. A common example consists in estimating the relationship between house prices and some characteristics of the good (surface, region, floor, etc) as well as prevailing macro-economic variables (interest rate, inflation, etc). Often, $\hat{g}(\cdot)$ is estimated once and for all on a fixed training set, $\mathcal{S}(0)$. In an online setup, though, one usually considers a model $\hat{g}_t(\cdot)$ trained on a set including recent observations, such as $\mathcal{S}(t) := \{(y_i, X_i),\ i \in \mathcal{T}(t)\}$ where the training period $\mathcal{T}(t) = \mathcal{T}(t-1) \cup \{t\}$ is *progressively enlarged*. Another common alternative is to disregard old information, e.g., to rely on a rolling window of fixed size, $\mathcal{T}(t) = \{t-m+1, t-m+2, \ldots, t\}$. This is relevant in the presence of *concept drift*, when the conditional probability distribution changes over time, i.e., $P_t(y|X) \neq P_{t+1}(y|X)$. In the housing example, this may happen because of changes in fiscal policy, wars or pandemics. The above approaches are pretty naive and raise the following question: Which period $\mathcal{T}(t)$ should one use for building the model $\hat{g}_t$ prevailing at time t?

An intuitive methodology would proceed in two steps: (i) identify when a change-point τ happens, which can be done by relying on classical detectors such as CUSUM, generalized likelihood or sequential methods, and (ii) restart the learning process from a model $\hat{g}$ which is reset to a reference state $\hat{g}_0$ at τ (i.e., considering an empty information set: $\mathcal{T}(\tau) = \varnothing$, $\mathcal{T}(\tau+1) = \{\tau+1\}$, etc). Yet, this seems inefficient because, by resetting the model once a drift is detected, it is likely to take time before it can deliver decent performance post-drift.

To circumvent this drawback, we introduce a more efficient approach which resets the model at τ not to a reference state, but exploiting the latest data which could not be considered as "statistically different" from that available at τ. This can be achieved in three steps: (i) as above, raise an alarm at the earliest time τ that a change is detected, (ii) when an alarm is raised, go backward in

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

time to identify the most likely time $\hat{\tau}$ at which the change-point actually took place, and finally (iii) update the learning period as $\mathcal{T}(t) = \{\hat{\tau}+1, \ldots, t\}$, i.e., re-launch as of τ the learning process based on the information set $\mathcal{T}(\tau)$.[1] This can be made sequential in a walk-forward setting: Noting $\tau_{i(t)}$ the latest alarm time raised by time t, the relevant training period is $\mathcal{T}(t) = \{\hat{\tau}_{i(t)}+1, \ldots, t\}$. In other words, we reset the on-line learning process of model $\hat{g}$ exploiting the largest piece of relevant data, and we do so only when a new alarm is raised.

To identify τ and $\hat{\tau}$, we rely on the efficient non-parametric change-point detection approach based on conformal CUSUM (CCUSUM) introduced in [[11]]. We illustrate its performance when combined with an algorithm $\hat{g}$ in the continuous learning literature. We opt for the Online Newton Step (ONS) because it is among the most popular algorithms in the field. In addition, it is a standard approach to analyze the performance of change-point approaches in a continuous learning context, thereby providing relevant benchmarks to compare with. To the best of our knowledge, it is the first time that CCUSUM is used to optimize re-train decisions in a continuous learning setup. The relevance of our approach is confirmed by numerical results in a controlled environment (simulations).

The paper is organized as follows. The literature on change-point detection and model fitting is briefly reviewed in Section 2. In Section 3, we detail our approach and apply it to the ONS algorithm. Section 4 provides the numerical results and Section 5 concludes.

2 Literature review

Adaptive online learning under concept drift is usually organized into drift detection and drift adaptation [9, 5]. Detection methods range from parametric sequential tests such as CUSUM [[10]] and Bayesian online change-point detection [1], or offline segmenters like PELT [8], to non-parametric windowing such as ADWIN [2] and conformal test-martingales including CCUSUM [11].

On the other hand, adaptation strategies include hard resets with full retraining, sliding or enlarging windows, and "soft" updates using forgetting factors or partial parameter resets, e.g., in Recursive Least Squares (RLS) with exponential forgetting [[6]]. These mechanisms are often combined with incremental learners such as Online Gradient Descent (OGD) [[12]], AdaGrad [[4]], Passive–Aggressive (PA) regression [[3]] and second-order methods such as Online Newton Step (ONS) [[7]].

In this paper we study how CCUSUM can optimally control retraining of ONS. Our experiments compare Lean-ONS with ADWIN-ONS (ONS with ADWIN resets), Standard ONS (no detector), AFF-ONS (an ONS variant with an adaptive forgetting factor on its curvature matrix, in the spirit of RLS-style forgetting), and the first-order baselines OGD, AdaGrad, PA-Regression and RLS, as well as Oracle ONS which is the standard ONS reset at the actual concept drift times.

[1] When model fitting is expensive, one may decide to not retrain the model every time a new observation is available, leading to the training period $\mathcal{T}(t) = \{\hat{\tau}+1, \ldots, \tau\}$.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

3 Retraining ONS using CCUSUM triggers

CCUSUM is a non-parametric version of CUSUM inspired from conformal prediction theory. It works as follows.

First, non-conformity scores (NCS) measuring the performance of a model $\hat{g}$ are collected. In this paper, we use as NCS the prediction mean-squared error (MSE) $\alpha_i := (y_i - \hat{y}_i)^2$.[2]. An increase in NCS reveals a deterioration of the model's performance, which may be the consequence of a concept drift.

Second, the NCSs are transformed into conformal p-values using

$$p_i \;:=\; \frac{1}{i}\left(\sum_{j=1}^{i} \mathbb{1}_{\{\alpha_j>\alpha_i\}} \;+ U_i \sum_{j=1}^{i} \mathbb{1}_{\{\alpha_j=\alpha_i\}}\right), \quad U_i \overset{i.i.d.}{\sim} \text{Uniform}[0,1] \tag{1}$$

where $\mathbb{1}_{\{.\}}$ is the indicator function.

Third, the conformal p-values p_i are mapped by a betting function f (typically f can be the power martingale, associated with $f(p) = \epsilon p^{\epsilon-1}$, $0 < \epsilon < 1$) to yield $f_i := f(p_i)$. Finally, we define the process S as $S_n = \prod_{i=0}^{n} f_i$, $S_0 := 1$.

Under the hypothesis H_0 that the observations (y_i, X_i) are exchangeable, the conformal p-values are uniformly distributed, in which case S is a martingale. Hence, deviations from H_0 can be detected thanks to Ville's inequality applied to S. However, S is sensitive to initial calibration and may lose power on late drifts. To address this problem, we rely instead on the CUSUM statistic γ:

$$\gamma_n \;=\; \max_j R_{j,n} \quad \text{where} \quad R_{i,n} = \frac{S_n}{S_{i-1}}. \tag{2}$$

The corresponding alarm and change times are then given by

$$\tau \;=\; \min\{n : \gamma_n \geq c\} \quad \text{and} \quad \widehat{\tau} \;=\; \arg\max_{1\leq j\leq \tau} R_{j,\tau} \;-\; 1. \tag{3}$$

More details about this procedure can be found in [11].

As explained in the introduction, this approach can be made dynamic by keeping the set $\mathcal{T}(t)$ unchanged until a new alarm is raised, in which case $\mathcal{T}(t)$ is updated and the procedure is restarted.

Our approach, called *Lean-ONS*, efficiently combines the ONS learner with a reset strategy based on the CCUSUM statistic γ. Building on the online regression setup of the Introduction, we now specialize to a streaming linear predictor under possible drift: Each time t we observe (X_t, y_t), the learner predicts $\hat{y}_t = \hat{g}_{t-1}(X_t)$ where $\hat{g}_t(X) := X^\top \hat{w}_t$. The key point of attention here is on which training period should $\hat{w}_t$ be learned. As long as no alarm is triggered, the model learns continuously according to the standard ONS update rule

$$\hat{w}_t \;\leftarrow\; \hat{w}_{t-1} \;-\; \eta\,(A_t + \lambda I)^{-1}(\hat{y}_t - y_t)X_t \quad \text{and} \quad A_t = A_{t-1} + X_t X_t^\top. \tag{4}$$

[2] Alternative NCS approaches for concept drift have been discussed in recent literature, see [11]

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

However, as soon as CCUSUM raises an alarm ($t = \tau_{i(t)}$), the change time $\hat{\tau}_{i(t)}$ is estimated backward, the training period is updated to $\mathcal{T}(t) = \{\hat{\tau}_{i(t)}+1, \ldots, \tau_{i(t)}\}$, and $(\hat{w}_t, A_t)$ are reset to the ridge solutions exploiting relevant information [5]:

$$A_t \leftarrow \sum_{j \in \mathcal{T}(t)} X_j X_j^\top, \qquad \hat{w}_t \leftarrow (A_t + \lambda I)^{-1} \sum_{j \in \mathcal{T}(t)} X_j \, y_j \, . \tag{5}$$

The CCUSUM detector is then reset to avoid an immediate retrigger. Finally, the rule (4) is restarted from the new state (5), i.e., $\mathcal{T}(t)$ is progressively enlarged with observations posterior to $\hat{\tau}_{i(t)}+1$ until another alarm is raised. This method yields a series of NCS $\alpha_t = \left(y_t - \hat{y}_t\right)^2$ which can be plugged in (1).

The procedure exploiting the conformal p-values and the associated test martingale is provably valid (type-I error controlled by α) when the scores are exchangeable under the no-drift regime. In our online setup, however, the predictor $\hat{w}_t$ changes with t according to (4). As a consequence, the map $(X_t, y_t) \mapsto \alpha_t$ is time-varying and the NCS are non-exchangeable[3]

4 Numerical experiments

We simulate an online linear regression $y_t = X_t^\top w_t + \varepsilon_t$, $t = 1, \ldots, T$, with $T = 2000$, $d = 5$, i.i.d. covariates $X_t \sim \mathcal{N}(0, I_d)$, and noise $\varepsilon_t \sim \mathcal{N}(0, 0.5^2)$. There is a single abrupt concept drift at $\tau^\star = 1000$. Before the change-point, $w_t = w^{(1)} = (3, -2, 1, 4, -3)^\top$; after the concept drift, $w_t = w^{(2)} = (-4, 5, -2, -1, 1)^\top$.

At each step t, the learner observes (X_t, y_t), predicts $\hat{y}_t = X_t^\top \hat{w}_{t-1}$, and incurs squared loss $(y_t - \hat{y}_t)^2$. This loss defines the NCS α_t, from which we build conformal p-values and the associated test martingale as in Section 3. CCUSUM then generates (i) alarm times τ and (ii) estimated change locations $\widehat{\tau}$.

Hyperparameters (learning rate, regularisation, detector thresholds, window sizes) are tuned once by grid search on independent synthetic runs with the same drift pattern, minimizing post-drift MSE. For each configuration, we run 500 Monte Carlo simulations with different seeds and average the results. Performance is evaluated via the cumulative squared error (CumSE), post-drift MSE, short-horizon recovery error (RecMSE_H), and the Cumulative RMSE (CRMSE). Formally, we have $\text{CumSE} = \sum_{t=1}^{T} (y_t - X_t^\top \hat{w}_{t-1})^2$, $\text{PostDriftMSE} = \frac{1}{T-\tau^\star} \sum_{t=\tau^\star+1}^{T} (y_t - X_t^\top \hat{w}_{t-1})^2$, $\text{RecMSE}_H = \frac{1}{H} \sum_{t=\tau^\star+1}^{\tau^\star+H} (y_t - X_t^\top \hat{w}_{t-1})^2$, $\text{CRMSE} = \sqrt{\frac{1}{h} \sum_{t=\tau^\star+1}^{\tau^\star+h} (y_t - X_t^\top \hat{w}_{t-1})^2}$, with $T = 2000$, $\tau^\star = 1000$, $H = 100$.

Table 1 reports the mean values of these metrics over 500 runs for the single change point (concept drift) scenario. Not surprisingly, the Oracle ONS achieves the lowest errors overall, due to its knowledge of the actual concept drift times $\tau^\star$.

[3]To circumvent this issue, we can keep a fixed calibration set of ONS residuals to form conformal p-values (via rank), which are then fed into CCUSUM; this yields an adjusted detector.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Table 1: Synthetic Concept-drift. Lower is better for all columns.

Algorithm	Marker	CumSE	PostDriftMSE	RecMSE_{H}
Standard ONS		74651.91	74.23	134.66
RLS		8384.24	7.97	66.62
ADWIN-ONS		8218.63	7.79	75.59
AdaGrad		5794.90	5.09	44.11
AFF-ONS		3021.07	2.58	22.55
PA-Regression		2272.62	1.47	9.25
OGD		2136.06	1.49	11.78
Lean-ONS		1623.15	1.20	9.67
Oracle ONS		1038.17	0.62	3.85

Figure 1 (left) summarises the overall performance through the distribution of the *CumSE* across 500 runs. Algorithms that fail to adapt (Standard ONS, RLS, ADWIN-ONS) exhibit very large total loss, while Lean-ONS is the best non-oracle method and lies close to the Oracle. Figure 1 (right) plots the average CRMSE trajectory over 500 runs. The initial spike at $h = 1$ reflects unavoidable detection lag, matching Standard ONS errors before the reset.

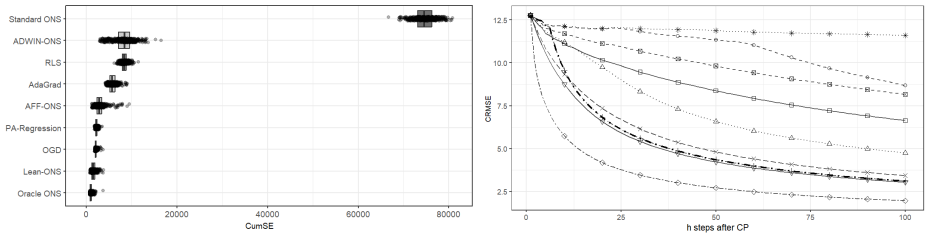

Fig. 1: Results over 500 runs. Left: CumSE distribution. Right: Averaged CRMSE for $h = 1, \ldots, 100$ steps post-drift. Lean-ONS is highlighted in bold.

Lean-ONS remains very close to the Oracle, extracting almost all achievable recovery while using a single data-driven retrain per concept drift. PA-Regression uses first-order passive–aggressive updates, that strongly correct large prediction errors on each new sample, so it reacts faster right after the change, hence its slightly smaller RecMSE_H. But it never forgets pre-drift data; its weights stay biased toward the old regime, hence its worse PostDriftMSE and CumSE.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

5 Conclusion

Lean-ONS approaches the Oracle using a single, data-driven retraining per change, outperforming standard ONS and baselines. An extended version covers diverse drifts (gradual, multiple, etc.) and real datasets, addresses exchangeability for Lean-ONS, and analyzes the impact of betting functions and NCS on efficiency.

References

[1] R. P. Adams and D. J. C. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[2] A. Bifet and R. Gavaldà. Learning from time-changing data with adaptive windowing. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*, pages 443–448, 2007.

[3] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[4] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[5] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and H. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46:44:1–44:37, 2014.

[6] S. Haykin. *Adaptive Filter Theory*. Prentice-Hall, Inc., USA, 3rd edition, 1996.

[7] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.

[8] R. Killick, P. Fearnhead, and I. A. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107:1590–1598, 2012.

[9] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31:2346–2363, 2019.

[10] E. S. Page. Continuous inspection schemes. *Biometrika*, 41:100–115, 1954.

[11] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*. Springer, 2nd edition, 2022.

[12] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 928–936, 2003.