Mach Learn
DOI 10.1007/s10994-014-5450-3

Optimizing regression models for data streams with missing values

Indrė Žliobaitė · Jaakko Hollmén

Received: 4 January 2014 / Accepted: 18 May 2014

Abstract Automated data acquisition systems, such as wireless sensor networks, surveillance systems, or any system that records data in operating logs, are becoming increasingly common, and provide opportunities for making decision on data in real or nearly real time. In these systems, data is generated continuously resulting in a stream of data, and predictive models need to be built and updated online with the incoming data. In addition, the predictive models need to be able to output predictions continuously, and without delays. Automated data acquisition systems are prone to occasional failures. As a result, missing values may often occur. Nevertheless, predictions need to be made continuously. Hence, predictive models need to have mechanisms for dealing with missing data in such a way that the loss in accuracy due to occasionally missing values would be minimal. In this paper, we theoretically analyze effects of missing values to the accuracy of linear predictive models. We derive the optimal least squares solution that minimizes the expected mean squared error given an expected rate of missing values. Based on this theoretically optimal solution we propose a recursive algorithm for producing and updating linear regression online, without accessing historical data. Our experimental evaluation on eight benchmark datasets and a case study in environmental monitoring with streaming data validate the theoretical results and confirm the effectiveness of the proposed strategy.

Keywords Data streams · Missing data · Linear models · Online regression · Regularized recursive regression

Editor: Joao Gama.

I. Žliobaitė (✉) · J. Hollmén
Department of Information and Computer Science, Aalto University, Aalto,
PO Box 15400, 00076 Espoo, Finland
e-mail: indre.zliobaite@aalto.fi; zliobaite@gmail.com

J. Hollmén
e-mail: jaakko.hollmen@aalto.fi

I. Žliobaitė · J. Hollmén
Helsinki Institute for Information Technology (HIIT), Helsinki, Finland

Published online: 04 July 2014

1 Introduction

The amount of automated data acquisition systems installed in the urban and natural environments is rapidly increasing. It is predicted that sensor data collected from satellites, mobile devices, outdoor and indoor cameras will become the largest information trove for our society in the coming years (Brobst 2010). Predictive models using such streaming data as an input are widely applied in production quality control, air pollution monitoring, detecting traffic jams or severe road conditions, route recognition, road navigation, cargo tracking and many more (Gama and Gaber 2007). In the last decade a lot of predictive modeling approaches for streaming data have been developed (Gama et al. 2014), however, the majority assume a standard operational setting where input data is available clean and preprocessed, ready for prediction.

In real automated data acquisition systems, missing values often occur. Physical measurement sensors are exposed to various risks due to, for instance, severe environmental conditions, physical damage or wear and tear. Moreover, often data acquisition systems rely on limited power sources (batteries), are installed in remote or hardly accessible locations, or are unaccessible during operation runtimes. Under such circumstances it is very common to have time intervals when some data values are missing; however, at the same time a predictive model needs to deliver predictions continuously and without delays.

As an example, consider a remote environment monitoring system, which measures meteorological variables, such as temperature, humidity, and air pressure. Suppose the goal is to monitor and predict the level of solar radiation, that cannot be directly measured. It has been observed in a case study (Zliobaite and Hollmen 2013) that at least some readings may be missing 25 % of the times; however, solar radiation estimation need to be provided all the time.

This paper investigates how to deal with missing values in predictive modeling with streaming data. Discarding the incoming observations that contain missing values is not an option, since there will be no predictions for these periods of time, when some of the input values are missing. We could maintain a set of predictive models built on feature subspaces of the input data, and for each combination of missing values apply a model that does not require this data, e.g. Polikar et al. (2010). However, such an approach is practical when small number of missing values is expected and models do not need to be updated online. If we consider models for all possible combinations of missing values, we will end up with keeping and updating online 2^r predictive models, where r is the number of input features, which is impractical and quickly becomes computationally infeasible.

Hence, we need to consider missing value imputation. Missing value imputation has been widely studied for static data scenarios (Allison 2001; Little and Rubin 2002). Two main approaches to missing value imputation can be distinguished: single imputation and model-based imputation. The simplest single imputation approach is to replace missing values by a constant value, for instance, a sample mean. The approach is computationally simple, however, it ignores relationship between variables in the data and therefore weakens variance and covariance estimates. Regression imputation takes into account the relationship between variables, by building regression models for missing features that take the non-missing features as inputs. However, it requires 2^r imputation models (for each combination of missing values) and is impractical and often computationally infeasible for streaming data applications.

Model-based imputation includes probably the most popular and the most powerful approaches, but carry even higher computational costs. Two main approaches are: maximum likelihood imputation and multiple imputation. Maximum likelihood imputation chooses as estimates those values that, if true, maximize the probability of observing the non-missing data, estimation requires iterative computations. Multiple imputation averages across multiple imputed data sets, using randomized imputation method. Model-based imputation methods are powerful for static datasets where the goal is to reconstruct the data as accurately as possible, but they have a big computational overhead, and are impractical or even infeasible for streaming data, where missing values need to be estimated in real time and the imputation models need to be continuously updated, considering that the missing value rates may vary over time.

Therefore, the only realistically feasible approach for data steams is single unconditional imputation, such as sample mean imputation. While it ignores relations between the input variables, on the positive side, we are not required to reconstruct the input data accurately, the goal is to make accurate predictions. We will leverage this by optimizing predictive models in such a way that the deficiencies of the imputation approach are taken into account and compensated for, and the expected prediction accuracy is maximized.

Our study contributes a theoretically supported least squares optimization criteria for building regression models robust to missing values, and a corresponding algorithm for building and updating robust regression online in a recursive manner without the need to store historical data in memory. Theoretical findings, experimental validation on eight benchmark datasets and a case study in environmental monitoring domain demonstrate the effectiveness of the proposed approach.

This paper builds upon our conference paper (Zliobaite and Hollmen 2013). While the conference paper introduced the problem of missing data in predictive modeling with data streams, the present paper considers a broader problem setting and presents new solutions. New contributions with respect to the conference version are as follows. Generalization of the problem setting from a fixed number of missing values, to a probabilistic setting, where each feature may have missing values with a different probability. Moreover, the probabilities of missing values may vary over time. New theoretical analysis based on this extended problems setting is presented. The conference paper presented only intuition driven guidelines on how to build robust predictive models, this paper presents a theoretically optimal least squares solution. Finally, the conference paper considered only pre-trained and fixed predictive models and the main focus was on diagnosing how robust the model is to missing values. Now predictive models can be updated over time, and we present a theoretically supported recursive algorithm for doing that.

The paper is organized as follows. Section 2 presents background and problem setting. In Sect. 3 we theoretically analyze prediction accuracy of linear regression models with missing data. Section 4 develops a theoretically optimal least squares solution for data streams with missing data, and presents an online algorithm for building and updating robust regression models. Experimental evaluation is reported in Sect. 5. Section 6 discusses related work, and Sect. 7 concludes the study.

2 Background and problem setting

Suppose we have r data sources generating multidimensional streaming data vectors $\mathbf{x} \in \Re^r$ (e.g. weather observation sensors). Our task is to predict the target variable $y \in \Re^1$ (e.g. solar radiation) using these data vectors as inputs. Data arrives in a sequence over time, predictions need to be made in real or near real time.

There may be time periods when some values in the input data are missing. We assume that for each data source i the probability of missing data p_i does not depend on other data sources, neither it depends on the value of the target variable y, i.e. data is missing at random. The probabilities of missing data p_i may vary over time.

The goal is to build a predictive model, that may be continuously updated online if needed, such that the prediction error is minimized, considering that input data may contain missing values. In this study, we consider linear regression models for prediction. We analyze linear regression models; which are among the simplest regression models, but at the same time linear models can be very powerful and flexible, especially when used with non-linear input features. Also, any non-linear process can be thought of being approximated by linear models. Linear models are easy to interpret. Last but not least, for linear models we can provide theoretical guarantees of the missing value handling solutions.

2.1 Linear regression

Linear regression models (Hastie et al. 2001) assume that the relationship between r input variables $x_1, x_2, \ldots, x_r$ and the target variable y is linear. The model takes the form

$$y = b_0 + b_1x_1 + b_2x_2 + \ldots + b_rx_r + \epsilon = \mathbf{x}\beta + \epsilon, \tag{1}$$

where ϵ is the error variable, the input vector $\mathbf{x} = (1, x_1, \ldots, x_r)$ is augmented by 1 for taking into account the additive bias, and the vector $\beta = (b_0, b_1, \ldots, b_r)^T$ contains the parameters of the linear model (regression coefficients).

There are different ways to estimate the regression parameters (Golub and Van Loan 1996; Hastie et al. 2001). Ordinary least squares (OLS) is a simple and probably the most common estimator. It minimizes the sum of squared residuals giving the following solution

$$\hat{\beta}_{OLS} = \arg\min_{\beta}\left((\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)\right) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \tag{2}$$

where $\mathbf{X}_{n\times r}$ is a sample data matrix containing n records from r sensors, and $\mathbf{y}_{n\times 1}$ is a vector of the corresponding n target values. Having estimated a regression model $\hat{\beta}$ the predictions for on new data $\mathbf{x}_{new}$ can be made as $o = \mathbf{x}_{new}\hat{\beta}$.

2.2 Prediction error

For measuring the prediction error we use the mean squared error (MSE), which is a popular measure due to its convenient analytical properties. For a test dataset it is computed as

$$MSE = \frac{1}{n}\sum_{l=1}^{n}(o^{(l)} - y^{(l)})^2 = \frac{1}{n}(\mathbf{o} - \mathbf{y})^T(\mathbf{o} - \mathbf{y}), \tag{3}$$

where y is the true target value, o is a prediction and n is the number of observations in the testing set.

The expected MSE can be analyzed as follows:

$$\begin{aligned} E[MSE] &= E\left[\frac{1}{n}\sum_{l=1}^{n}\left(o^{[l]} - y^{[l]}\right)^2\right] = E[(o - y)^2] \\ &= E[o^2 - 2oy + y^2] = E[o^2] - 2E[oy] + E[y^2] \\ &= Var[o] - 2Cov[o, y] + Var[y] + (\bar{o} - \bar{y})^2. \end{aligned} \tag{4}$$

Here $\bar{o}$ and $\bar{y}$ denote the means of the prediction and the true target variable respectively. The last equation follows from $Var[z] = E[z^2] - (E[z])^2$ and $Cov[x, z] = E[xz] - E[x]E[z]$.

Let the prediction be $o = \mathbf{x}\beta$. Then the variance of this prediction is

$$Var[o] = Var\left[\sum_{i=i}^{r} b_i x_i\right] = \sum_{i=1}^{r}\sum_{j=i}^{r} b_i b_j Cov[x_i, x_j] = \beta^T \mathbf{C}\beta, \tag{5}$$

where $\mathbf{C} = \frac{1}{n}(\mathbf{X}^T\mathbf{X} - \bar{\mathbf{x}}^T\bar{\mathbf{x}})$ is the covariance matrix of the input data, and $\bar{\mathbf{x}}$ denotes the mean of the input data.

The covariance between the prediction and the target is

$$Cov[o, y] = Cov[\mathbf{x}\beta, y] = \sum_{i=1}^{r} b_i Cov[x_i, y] = \beta^T \mathbf{z}, \tag{6}$$

where $\mathbf{z} = \frac{1}{n}(\mathbf{X}^T\mathbf{y} - \bar{\mathbf{x}}\bar{y})$ is the relation vector of the input data and the target variable. Our decomposition can be assimilated with the bias-variance decomposition, since we express the error in terms of the bias component as well as terms accounting for variance of the prediction.

2.3 Handling missing values

Detecting sensor failures is beyond the scope of this work. To keep the focus we assume that it is known when values are missing and there is no need to detect that. We assume that the data collection system can signal sensor failures automatically. If this is not the case one can set up a simple rule based detector, such as: if the value is constant for a period of time declare sensor failure.

We consider the following strategy for handling missing values. Predictions need to be made for every observation, therefore, we cannot ignore observations with missing values. We consider replacing the missing values by the sample mean, as they arrive. As discussed in the introduction, this is a computationally light method, suitable for streaming data settings.

However, for training and updating the predictive model itself, we can use case deletion. Since data streams produce large amounts of data, even after deleting, say, 25 % of the observations, there is still plenty of data left for training the model. We consider training the model only on complete observations, that contain no missing values. This alleviates the problem of weakening the variance and the covariance estimates (Allison 2001), when using mean imputation.

Biased estimates would not pose a problem for prediction, if the probabilities of missing data stay fixed over time. In such a case the variances and the covariances in the testing data would be biased in the same way as in the training data. However, as we will see in the theoretical analysis and experiments, if the probabilities of missing values vary, biased estimations largely distort the predictions. To correct for such discrepancies, we introduce a special optimization criteria for linear regression, that enables using computationally light mean imputation in data stream applications.

3 Theoretical analysis of prediction errors with missing data

We start from analyzing, what happens to the prediction error of a linear model when some input values are missing. The following proposition gives the expected error.

Proposition 1 *Suppose data sources produce missing values independently from each other. Let $\mathbf{p} = (p_1, p_2, \ldots, p_r)^T$ be the probabilities of missing values for each data source. Considering missing values, the expected error of a linear model is*

$$E[MSE_{\mathbf{p}}] = \beta^T\mathbf{HD}\beta + \beta^T\mathbf{H}(\mathbf{C} - \mathbf{D})\mathbf{H}\beta - 2\beta^T\mathbf{Hz} + Var_0[y] + (\bar{\mathbf{x}}\beta - \bar{y})^2.$$

where β is a vector of the regression coefficients, $\mathbf{C}$ is the covariance matrix of the input data, $\mathbf{z}$ is the relation vector of the input data and the target variable, $\mathbf{D} = diag(\mathbf{C})$, $\mathbf{H} = \mathbf{I} - diag(\mathbf{p})$, $\mathbf{I}_{r\times r}$ is the identity matrix.

Proof can be found in the Appendix.

Let us first analyze boundary cases. If no data is missing, $\mathbf{p} = (0, 0, \ldots, 0)^T$, then $\mathbf{H} = \mathbf{I}$. The error resorts to the expression in Eq. 4, hence, $E[MSE_{\mathbf{p}}] = MSE_0$.

If all the input values are missing, that is $\mathbf{p} = (1, 1, \ldots, 1)^T$, then $\mathbf{H} = \mathbf{0}_{r\times r}$ and $E[MSE_{\mathbf{p}}] = Var_0[y] + (\bar{\mathbf{x}}\beta - \bar{y})^2$. If we make naive predictions $o = \bar{y}$, the prediction error will be equal to the variance of the target $E[MSE_{\mathbf{p}}] = Var_0[y]$.

In order to analyze the effects of missing values further, we investigate a more convenient case, where the input data and the target are assumed to be standardized (zero mean, unit variance). There is no loss of generality, since any data can be easily standardized by subtracting the sample mean and dividing by the sample variance.

Proposition 2 *If the input data and the target variable is standardized (zero mean, unit variance), suppose data sources produce missing values independently from each other. Let $\mathbf{p} = (p_1, p_2, \ldots, p_r)^T$ be the probabilities of missing values for each data source. Considering missing values, the expected error of a linear model is*

$$E[MSE^{\star}_{\mathbf{p}}] = \beta^T \mathbf{H}\beta + \beta^T \mathbf{H}(\mathbf{C} - \mathbf{I})\mathbf{H}\beta - 2\beta^T \mathbf{H}\mathbf{z} + 1.$$

The proof follows directly from Proposition 1, and can be found in the Appendix.

From Proposition 2 we can find an interesting characteristic of the error. If the input data is independent, then the covariance matrix is diagonal, in this case $\mathbf{C} = \mathbf{I}$, and the second term in Proposition 2 disappears, the error becomes $\beta^T \mathbf{H}\beta - 2\beta^T \mathbf{H}\mathbf{z} + 1$.
We can see that if the input data is independent, the error linearly depends on the probabilities of missing values.

For further analysis and clarity of exposition, let us make a simplifying assumption that all data sources have equal probability of producing missing values, i.e. $p_1 = p_2 = \ldots = p$.

Proposition 3 *If the input data and the target variable is standardized (zero mean, unit variance), and data sources produce missing values independently with equal prior probability p, the expected error of a linear model is*

$$E[MSE^{\star}_{p}] = (1 - p)E[MSE_0] + p - p(1 - p)\beta^T (\mathbf{C} - \mathbf{I})\beta.$$

Proof can be found in the Appendix.

From Proposition 3 we can see that if input data is not correlated ($\mathbf{C} = \mathbf{I}$), then the error increases linearly with the probability of missing values, $E[MSE^{\star}_{p}] = (1-p)E[MSE_0] + p$. However, if $\mathbf{C} \neq \mathbf{I}$ the speed of increase in error depends on the term $\beta^T (\mathbf{C} - \mathbf{I})\beta$. The next proposition analyzes its behavior.

Proposition 4 *Given an $r \times r$ covariance matrix $\mathbf{C}$ and a vector $\beta \in \mathbb{R}^r$ with at least one non-zero element, the term $\beta^T (\mathbf{C} - \mathbf{I})\beta$ is bounded by*

$$-\beta^T \beta \leq \beta^T (\mathbf{C} - \mathbf{I})\beta \leq (r - 1)\beta^T \beta.$$

Proof can be found in the Appendix.

From Proposition 4 we see that the term $\beta^T (\mathbf{C} - \mathbf{I})\beta$ can be positive or negative depending on the regression model (β). That means, that it is possible to find such β that would make the term $\beta^T (\mathbf{C} - \mathbf{I})\beta$ positive, and, as a result, make the increase in error (Proposition 3)

Table 1 Summary of analyzed MSE cases.

Notation	Proposition	Missing values	Equal probabilities of missing	Standardized inputs and target
MSE_0	Eq. 4	No	–	–
$MSE_{\mathbf{p}}$	Prop. 1	Yes	No	No
$MSE^{\star}_{\mathbf{p}}$	Prop. 2	Yes	No	Yes
$MSE^{\star}_{p}$	Prop. 3	Yes	Yes	Yes

even slower than linear, which happens if the input data is independent or no dependence is captured by the predictive model. In other words, the proposition suggests that it is possible to utilize the redundancy in input data to slow down the increase in error, that happens due to missing values. In the next section we will investigate, how to achieve that.

Table 1 summarizes the cases that were analyzed. Now, as we have theoretical expressions for the expected errors with missing values, we can proceed with developing predictive models, that would minimize those errors.

4 Building regression models robust to missing data

We will develop a robust linear regression model for data streams in two stages. First we will find an optimal solution in the static learning scenario assuming fixed probabilities of missing values **p**. Next we will develop online learning algorithms utilizing that optimal solution in streaming data scenario with evolving probabilities of missing data.

4.1 Optimization criteria for robust regression

For convenience and interpretability, and without loss of generality we continue assuming standardized data. Our goal is to optimize a linear regression model in such a way that the expected error with missing data, given in Proposition 2, is minimized. The following proposition gives the solution.

Proposition 5 *If the input data and the target variable is standardized (zero mean, unit variance), and data sources produce missing values independently with prior probabilities* $\mathbf{p} = (p_1, p_2, \ldots, p_r)^T$. *The expected prediction error is minimized with regression model*

$$\hat{\beta}_{ROB} = \left(\mathbf{X}^T\mathbf{X}(\mathbf{I} - diag(\mathbf{p})) + diag(\mathbf{p})n\right)^{-1}\mathbf{X}^T\mathbf{y},$$

where n is the training sample size.

Proof is given in the Appendix.

The next proposition provides a special case, where the prior probabilities of missing values are equal for all the data sources.

Proposition 6 *If the input data and the target variable is standardized (zero mean, unit variance), and data sources produce missing values independently with equal prior probabilities p. The expected prediction error is minimized with regression model*

$$\hat{\beta}_{ROB} = \left((1-p)\mathbf{X}^T\mathbf{X} + pn\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y},$$

where n is the training sample size.

Proof can be found in the Appendix.

This solution down-weights the off-diagonal elements of the covariance matrix. Observe, that it is closely related to the Ridge regression (RR) (Hoerl and Kennard 1970; Hastie et al. 2001). If the input data is correlated, regularization is often used for estimating the regression parameters. The ridge regression (RR) regularizes the regression coefficients by imposing a penalty on their magnitude. The RR solution is

$$\hat{\beta}_{RR} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y},$$

where $\lambda > 0$ controls the amount of shrinkage: the larger the value of λ, the greater the amount of shrinkage.

An alternative form of the ROB regression (Proposition 6) may be useful. If the covariance matrix $\mathbf{C}$ and the relation vector between the input data and the target variable $\mathbf{z}$ are known, the regression model resorts to

$$\hat{\beta}_{ROB} = ((1-p)\mathbf{C} + p\mathbf{I})^{-1}\,\mathbf{z}. \tag{7}$$

4.2 An illustrative example

Let us consider a toy example for illustrating the performance of the new robust regression. Suppose there are four data sources identical to each other, and the target variable is identical to all sources: $x_1 \sim \mathcal{N}(0, 1)$, $x_1 = x_2 = x_3 = x_4 = y$.

Consider four regression models, that would all give perfect predictions, i.e. $MSE_0 = 0$:

- **Independent**: $o = x_1$;
- **Shared**: $o = 0.25x_1 + 0.25x_2 + 0.25x_3 + 0.25x_4$;
- **Overfitted**: $o = 1.75x_1 - 1.25x_2 + 0.75x_3 - 0.25x_4$;
- **ROB regression**: different model for each value of p.

The Independent model is the simplest approach, which does not utilize any redundancy in the data, and takes only one data source as an input. The shared model takes equal contributions from each data source. In fact, this solution is equivalent to PCA regression, that would build a model on the data rotated towards the first principle component. Here variance and redundancy in all the data sources is utilized. The overfitted model also utilizes redundancy, but intuitively we can see that the weights at each data source are unnecessarily high. Finally, the new ROB regression (Proposition 6) produces a different regression model taking the probability of missing values p as an input. For example, if $p = 0.5$, the model would be $o = 0.4x_1 + 0.4x_2 + 0.4x_3 + 0.4x_4$. This model is theoretically optimal for this fixed probability of missing data ($p = 0.5$). The model originates from Proposition 6 after substituting the parameters of the toy data $\mathbf{X}^T\mathbf{X} = n\mathbf{1}_{4\times4}$, and $\mathbf{X}^T\mathbf{y} = n\mathbf{1}_{4\times1}$, where $\mathbf{1}$ is a matrix of ones. Note, that n cancels out.

Figure 1 compares the performance of the four models over all spectrum of p, where $p = 0$ means that no data is missing and $p = 1$ means that all input data is missing. A dashed line indicates a naive prediction that would not use any input data, and always predict a constant value $o = 0$. The left plot depicts theoretical errors, and the right plot empirical errors tested over 10,000 samples. We see that the performance nearly identical, which, as well, validates the theoretical results.

The Overfitted model performs worse than a naive prediction would perform, which can be explained by the theoretical expression for error in Proposition 3. The performance is so bad because of the magnitude of the term $-\beta^T(\mathbf{C}-\mathbf{I})\beta = 4.25$. For the Independent model $\beta^T(\mathbf{C}-\mathbf{I})\beta = 0$ and hence the increase in error is linear with the increase in probability

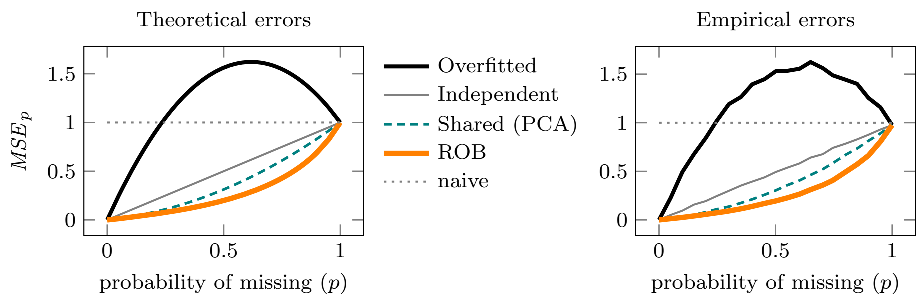


Fig. 1 Prediction error as a function of prior probability of missing values (p) with four regression models.

of missing values p. The Shared model has $-\beta^T(\mathbf{C}-\mathbf{I})\beta = -0.75$, which makes the increase in error lower than than linear, and we see that in the figure. Finally, the ROB model adjusts the regression coefficients based on expectations about p. For example, if $p = 0.5$, the ROB model would have $-\beta^T(\mathbf{C}-\mathbf{I})\beta = -1.92$, but higher initial error $MSE_0 = 0.4$. Nevertheless, the theoretical error (in Proposition 3) of ROB would be lower than that of the Shared model: $MSE_{0.5}(ROB) = (1-0.5)0.4 + 0.5 - 0.5(1-0.5)1.92 = 0.22$, and $MSE_{0.5}(Shared) = (1-0.5)0 + 0.5 - 0.5(1-0.5)0.75 = 0.31$. The example illustrates how ROB achieves optimal solution by dynamically balancing minimization of the initial error and robustness to missing data.

4.3 Online learning

Now as we have a theoretically optimal solution for handling missing data in batch learning, we can proceed to constructing an online algorithm for streaming data settings. The main restriction for online algorithms is no access to the historical data. All the parameter estimations need to be done recursively and only data summaries of a fixed size (that does not depend on the amount of data observed) can be stored.

We will construct online algorithms for data streams with missing data in two stages. First, we consider the probabilities of missing values to be fixed, the goal is to train a model incrementally with the incoming data that would be equivalent to the batch model trained on all that data. In the final stage we will consider evolving probabilities of missing values, and possibly evolving relation between the input data and the target (known as concept drift (Gama et al. 2014)).

Online learning version for the OLS regression (2) is a known result (Scharf 1990). This algorithm has no adaptation mechanisms, it is meant for learning over stationary data that arrives in a stream. Algorithm 1 gives the pseudocode for the solution.

Algorithm 1: Online OLS regression (Scharf 1990)

```
Data: previous model $\beta_{t-1}$, new observation $\mathbf{x}_t$, $y_t$,
previous covariance estimate $\mathbf{S}_{t-1}^{-1}$
Result: updated model $\beta_t$ and covariance estimate $\mathbf{S}_t^{-1}$
1 $\mathbf{S}_t^{-1} = \mathbf{S}_{t-1}^{-1} - \frac{\mathbf{S}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^T\mathbf{S}_{t-1}^{-1}}{1+\mathbf{x}_t^T\mathbf{S}_{t-1}^{-1}\mathbf{x}_t}$;
2 $\beta_t = \beta_{t-1} + \mathbf{S}_t^{-1}\mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1})$;
```

Proposition 7 *The iterative OLS solution (provided in Algorithm 1) is equivalent to the OLS solution.*

The proof, following Jordan (1998), is provided in the Appendix.

We propose an online version for ROB regression, that was introduced in Proposition 5. This version is meant for stationary data streams, and the prior probabilities of missing values $diag(\mathbf{p})$ need to be input as a parameter. The algorithmic solution is given in Algorithm 2.

Algorithm 2: Online ROB regression

```
Data: previous model $\beta_{t-1}$, new observation $\mathbf{x}_t$, $y_t$,
prior probabilities of missing values $\mathbf{p} = (p_1, p_2, \ldots, p_r)^T$,
previous estimate of the regularized covariance $\mathbf{S}_{t-1}$
Result: updated model $\beta_t$ and covariance estimate $\mathbf{S}_t$
1 if no missing values in $\mathbf{x}_t$ then
2     $\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T - diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I})$;
3     $\beta_t = \beta_{t-1} + \mathbf{S}_t^{-1}\mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1}) - \mathbf{S}_t^{-1}diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I})\beta_{t-1}$;
4 end
```

Proposition 8 *The online ROB regression (Algorithm 2) is equivalent to the batch ROB regression (Proposition 5).*

Proof can be found in the Appendix.

Recall that in Proposition 5 prior standardization of the input data and the target variable is assumed. That means, that Algorithm 2 as well operates under the assumption of standardized data. Online standardization can be easily achieved following the following procedure summarized in Algorithm 3.

Algorithm 3: Online standardization

```
Data: new observation $x_t$,
previous mean and standard deviation estimates $m_{t-1}$, $s_{t-1}$
Result: standardized data $x_t^\star$, updated estimates $m_t$, $s_t$
1 standardize data $x_t^\star = (x_t - m_{t-1})/s_{t-1}$;
2 update mean estimate $m_t = \eta x_t + (1-\eta)m_{t-1}$, where $\eta = \frac{1}{t}$;
3 update standard deviation estimate $s_t = \sqrt{\eta(x_t - m_t)^2 + (1-\eta)s_{t-1}^2}$;
```

In this procedure there is no forgetting, all the observations are taken into account with equal weights. The absolute weights of individual observations in the mean and standard deviation estimates are decreasing over time to keep the contributions equal. This is achieved by having η as a function of t. However, in the data streams scenario sometimes it is desired to give more emphasis to more recent observations. That can be achieved by replacing η in Algorithm 3 by a fixed constant value, for example $\eta = 0.001$. Then the mean and standard deviation estimates will be equivalent to taking the exponential moving average.

4.4 Full algorithmic solution for streaming data (ROBstream)

Online ROB regression in Algorithm 2 recursively learns a model that is optimal for fixed probabilities of missing values $\mathbf{p}$. Finally, we propose a full algorithm for predictive modeling over streaming data with missing values, where the probabilities of values to be missing may

Table 2 Storage memory requirements, r is the number of data sources.

	Estimate	Size	Memory
$\mathbf{C}$	Covariance matrix	$r \times r$	r^2
$\mathbf{z}$	Input-target relation	$r \times 1$	r
$\mathbf{p}$	Probabilities of missing	$r \times 1$	r
	Mean and SD of inputs	$2 \times r \times 1$	$2r$
	Mean and SD of target	$2 \times 1 \times 1$	2
Total memory			$r^2 + 4r + 2$

change over time, as well as the relation between the input variables and the target variable potentially may evolve over time. The algorithm has adaptation mechanisms, which allow to gradually forget old data, and focus on the new data distribution.

Hence, we need to be able to adjust the optimization criteria depending on $\mathbf{p}$. We can do that if we have a pure (not regularized) up to date estimate of the covariance matrix $\mathbf{C}$ and an up to date estimate of the relation between the input data and the target $\mathbf{z}$. Since we continue assuming standardized data, the estimates can be expressed as $\mathbf{C}_t = \frac{1}{t}\mathbf{X}_t^T\mathbf{X}_t$ and $\mathbf{z}_t = \frac{1}{t}\mathbf{X}_t^T\mathbf{y}_t$. Then, from Proposition 5, the regression model at time t can be expressed as

$$\beta_t = (\mathbf{C}_t - diag(\mathbf{p}_t)(\mathbf{C}_t - \mathbf{I}))^{-1}\,\mathbf{z}_t. \tag{8}$$

Hence, the algorithm would need to keep and incrementally update five estimates, summarized in Table 2 together with storage memory requirements. The total required storage is quadratic to the number of input features.

Algorithm 4 presents the algorithm. Online standardization is not included, it can be done following the procedure presented in Algorithm 3.

Algorithm 4: ROB regression for data streams

```
Data: stream of observations $\mathbf{x}$,
stream of true target values $y$,
forgetting factors $\alpha \in (0,1)$, $\gamma \in (0,1)$
Result: stream of predictions $o$
1  Initialization: $\mathbf{C} \leftarrow \mathbf{I}$, $\mathbf{z} \leftarrow (1,\dots,1)^T$, $\mathbf{p} \leftarrow (0,\dots,0)^T$, $\beta \leftarrow (0,\dots,0)^T$;
2  for $t \leftarrow 1$ to $\infty$ do
3      new observation $\mathbf{x}$ arrives, predict $o = \mathbf{x}\beta$ ;
4      true target value $y$ arrives ;
5      if no missing values in $\mathbf{x}$ then
6          update covariance $\mathbf{C} \leftarrow \alpha\mathbf{x}^T\mathbf{x} + (1-\alpha)\mathbf{C}$;
7          update relation $\mathbf{z} \leftarrow \alpha\mathbf{x}^T y + (1-\alpha)\mathbf{z}$;
8          update missing value estimate $\mathbf{p} \leftarrow (1-\gamma)\mathbf{p}$;
9      end
10     else
11         record missing value indicator $\mathbf{m}$, where $m_i = 1$ if $i^{th}$ value in $\mathbf{x}$ is missing, and 0 otherwise ;
12         update missing value estimates $\mathbf{p} \leftarrow \gamma\mathbf{m} + (1-\gamma)\mathbf{p}$;
13     end
14     update model $\beta = (\mathbf{C} - diag(\mathbf{p})(\mathbf{C} - \mathbf{I}))^{-1}\,\mathbf{z}$;
15 end
```

The parameters α and γ are forgetting factors (weights) for the standard exponential moving average procedure, which is used as a forgetting mechanism. The weights need to be between 0 and 1, where 0 means no forgetting, close to 0 corresponds to very slow forgetting, and close to 1 corresponds to very fast forgetting, and very dynamic updates.

If the forgetting factors α and γ are fixed, then more recent observations are emphasized and old observations are gradually forgotten. In case one wishes only to include new information, but not forget any old information, then the following settings should be used: $\gamma = 1/t$, where t is the time count since the beginning, and $\alpha = 1/(j+1)$, where j is the number of covariance matrix updates so far.

The operation of matrix inversion in line 14 is the most expensive computationally. If computational resources are scarce, one may chose to update the model more rarely (i.e. execute line 14 say every 10^{th} time). The only requirement is that the missing value estimates are being updated continuously (lines 8 and 12).

The algorithm assumes immediate availability of the target value, the way it is typically assumed in the standard streaming data research settings. In case labels are arriving with a delay, one could consider fixing a maximum time period to wait for the label to arrive. If the label has arrived, treat it as prescribed in the algorithm. If the label has not arrived within the specified time, then just discard the instance without using it for model update.

5 Experimental analysis

In this section, we experimentally evaluate the proposed techniques on real benchmark datasets with simulated missing values, and a real world case study with real missing values. Firstly, we experimentally compare the proposed ROB regression to state-of-the art linear regression models on a set of benchmark datasets in the batch learning setting. Secondly, we analyze the performance of the online version of the ROB regression. Finally, we evaluate the proposed adaptive learning algorithm ROBstream on a case study in environmental monitoring domain.

5.1 Comparison to the state-of-the art linear regression models

Theoretically, ROB regression is the optimal solution minimizing the expected error. In this experiment we investigate how ROB regression performs in practice as compared to seven state of the art regressions.

5.1.1 Linear regression models for comparison

We compare the performance of ROB with seven regression models in Table 3. **ALL** builds a regression model on all r input variables. **SEL** selects k input variables that have the largest absolute correlation with the target variable (correlation is measured on the training data) and builds a regression model on those k variables. **PCA** rotates the input data towards k principal components and then builds a regression model on those k new attributes. **PLS** is a popular technique in chemometrics (Qin 1998). Input data is rotated to maximize the covariance between the inputs and the target. A regression model is built on k new attributes. For feature selection SEL, PCA and PLS models we set the number of components to be a half of original number of features: $k = r/2$.

ALL, SEL and PCA use the OLS optimization procedure (OLS) for parameter estimation. In addition, we test the same approaches but using the regularized RR, these models are denoted as rALL, rSEL and rPCA. PLS uses its own iterative optimization procedure, there is no regularization. The regularization parameter in the RR experiments is fixed to 100.

Table 3 Linear regression models for comparison: OLS - ordinary least squares, RR - Ridge regression.

		Inputs			
		all r	selected k	PCA k	PLS k
Optimization	OLS	ALL	SEL	PCA	PLS
	RR	rALL	rSEL	rPCA	

Table 4 Datasets r - number of input variables, n - number of observations, Xi - collinearity.

	r	n	Xi	Source	Domain
Catalyst	13	8 687	0.54	NiSIS[2]	Chemical production
Chemi	70	17 562	0.21	INFER[1]	Chemical production
Concrete	8	1 030	0.23	UCI[3]	Civil engineering
CPU	19	8 192	0.25	DELVE[4]	Computing
Gaussian	30	10 000	0.14	–	Synthetic
Protein	9	45 730	0.57	UCI[3]	Protein structure
Wine	10	4 897	0.18	UCI[3]	Chemical analysis
Year	90	515 345	0.11	UCI[3]	Music songs

	Input data	Target variable
Catalyst	Sensor readings	Catalyst activity
Chemi	Sensor readings	Concentration of product
Concrete	Measurement	Strength of concrete
CPU	Activity measures	Time in user mode
Gaussian	$\mathcal{N}(\mathbf{0}, s^T s)$, where $s \sim \mathcal{U}(-1, 1)$	$\sum_{i=1}^{30} x_i + u$, where $u \sim \mathcal{N}(0, 6)$
Protein	Physicochemical properties	Size of residue
Wine	Chemical measurements	Wine quality
Year	Timbre features	Year of a song

[1] Source: http://infer.eu/
[2] Source: http://www.nisis.risk-technologies.com/
[3] Source: http://archive.ics.uci.edu/ml/
[4] Source: http://www.cs.toronto.edu/~delve/

5.1.2 Datasets

We consider eight regression datasets collected from different sources. For a wide coverage we chose datasets from various domains and to be different in sizes. The characteristics are summarized in Table 4. Among other characteristics we report collinearity Ξ, which is measured as the mean absolute covariance between input variables, $\Xi = \frac{\sum(|\mathbf{C}|)-k}{k(k-1)}$. The input data is standardized, thus $\Xi \in [0, 1]$, where 0 means that data is independent, and 1 means that it is fully collinear.

5.1.3 Experimental protocol

The experimental protocol is as follows. We investigate how the prediction error depends on the probability of missing values. For easier interpretation of the results we consider the prior probabilities of missing data to be equal, i.e. $\mathbf{p} = (p, \ldots, p)^T$. The original datasets do not have missing values. We introduce missing values with a probability p for each input variable independently at random. We investigate all the range of probabilities p from 0 to 1.

Each dataset is split into training and testing at random (equal sizes). The regression models are trained on the training part and the reported errors and sample covariances are estimated on the testing part. The input data and the target variable is standardized, the mean and the variance for standardization is calculated on the training data. We repeat every experiment 100 times and report averaged results.

5.1.4 Results

The results of this experiment are presented in Fig. 2, which plots testing errors as a function of the probability of missing values. We can see that the proposed ROB regression clearly demonstrates the best performance on all the datasets. The advantage of ROB is particularly prominent when the probability of missing values is high (around 50 % or so). The lower the probability p, the more close the performance of ROB is to ALL, which is a theoretically supported result. From Proposition 6 we can see that if $p = 0$ then ROB regression becomes the same as ordinary regression optimized using the ordinary least squares, that is ALL.

We can also observe that on some datasets ALL and PLS perform particularly badly generating a huge peak in error near $p = 0.5$. This happens because of the term $\beta^T(\mathbf{C} - \mathbf{I})\beta$ in Proposition 1. If input data is highly collinear and β is not regularized in any way, then the term $-p(1-p)\beta^T(\mathbf{C} - \mathbf{I})\beta$ is likely to be positive and boost the error. Table 5 presents empirical $-\beta^T(\mathbf{C} - \mathbf{I})\beta$ for all the eight datasets, estimated on full datasets. For ROB we set $p = 0.5$, other models do not depend on p. We see that on Chemi, Gaussian and Catalyst datasets ALL indeed has much higher $-\beta^T(\mathbf{C} - \mathbf{I})\beta$ term that most of the other models. This corresponds to large deterioration in performance in Fig. 2 on these datasets.

5.2 What if estimation of the prior probabilities is incorrect?

The proposed ROB requires an accurate estimate of the missing value rate p. In this section we analyze how ROB performs, if the estimate of p is noisy.

5.2.1 Experimental protocol

The experimental protocol is the same as in the previous section. We introduce missing values in data at random with a fixed prior probability p. We compare the best performing off-the-shelf approach PCA regression, and ROB regression with a fixed estimate of the missing value rate $p^\star$ (ROB uncertain). For comparison, we also analyze the performance of ROB with an exact estimate of p (ROB ideal).

For each plot we suppose that the missing value rate is set to $p = p^\star$, and investigate what happens, if the actual p varies ± 10 %. We analyze the performance on the toy example presented in Sect. 4.2, and the Chemi dataset, which is a representative sensory dataset with underlying high dimensional prediction problem.

5.2.2 Results

Figure 3 plots the results. We see that within the vicinity of the assumed $p^\star$, indicated by a black dot, the performance of ROB fixed is very close to that of ROB exact, and clearly better that the best alternative (PCA regression).

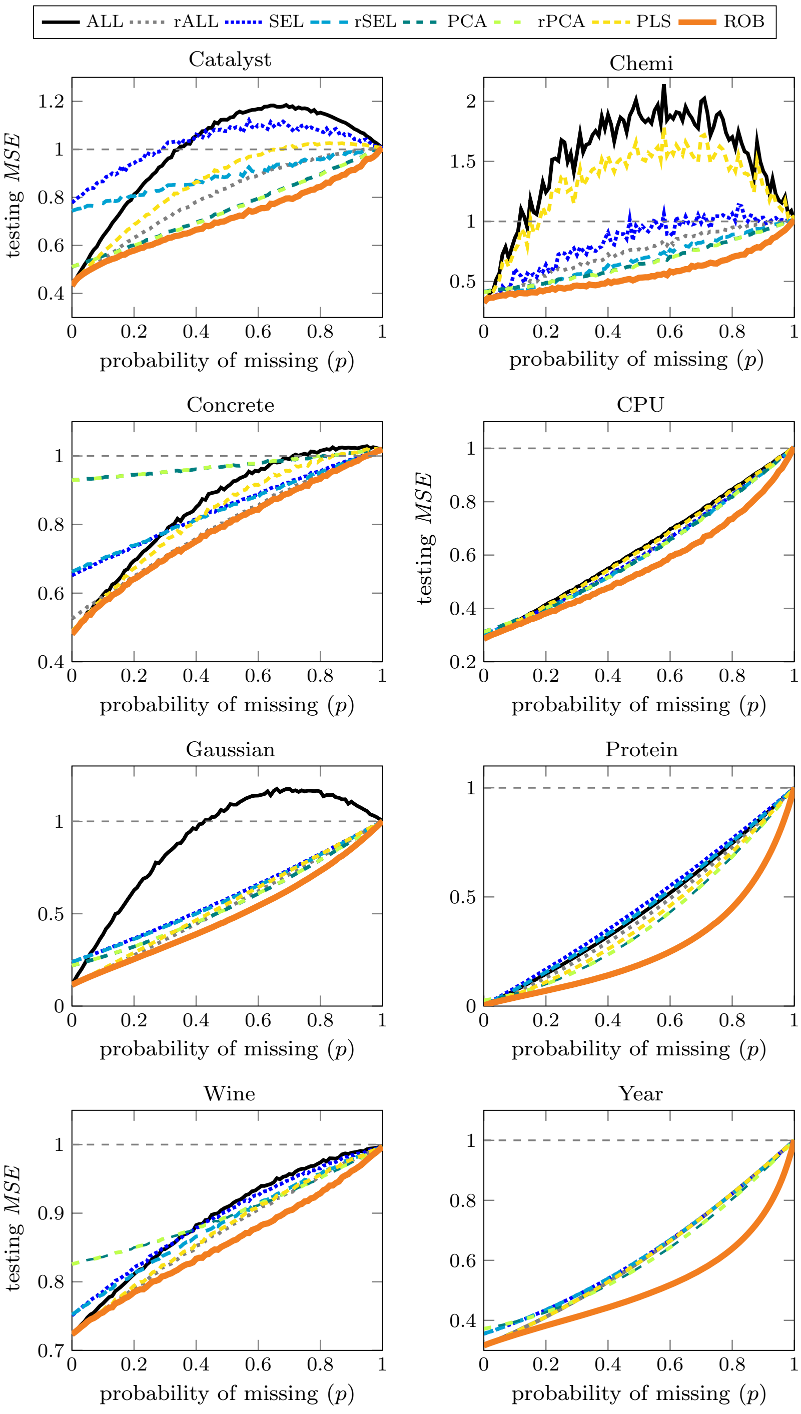


Fig. 2 Empirical *MSE* versus the probability of input data source failing averaged over 100 runs.

Table 5 Empirical $-\beta^T(\mathbf{C}-\mathbf{I})\beta$, the smaller the better

	ALL	rALL	SEL	rSEL	PCA	rPCA	PLS	ROB
Catalyst	1.6	0.8	0.5	0.1	0.0	−0.1	0.9	**–0.4**
Chemi	6.4	3.1	0.4	0.2	−0.2	−0.2	5.9	**–0.8**
Concrete	0.6	0.3	0.0	0.0	**–0.0**	**–0.0**	0.5	0.1
CPU	−0.1	−0.2	−0.2	−0.3	−0.3	−0.3	−0.1	**–0.8**
Gaussian	0.6	−0.1	−0.1	−0.2	−0.3	−0.3	−0.0	**–0.6**
Protein	−0.3	−0.4	−0.2	−0.2	−0.7	−0.7	−0.6	**–2.2**
Wine	0.2	0.1	0.1	0.1	**–0.1**	**–0.1**	0.1	**–0.1**
Year	−0.3	−0.3	−0.3	−0.3	−0.4	−0.4	−0.3	**–1.4**

The best result on each dataset is in bold

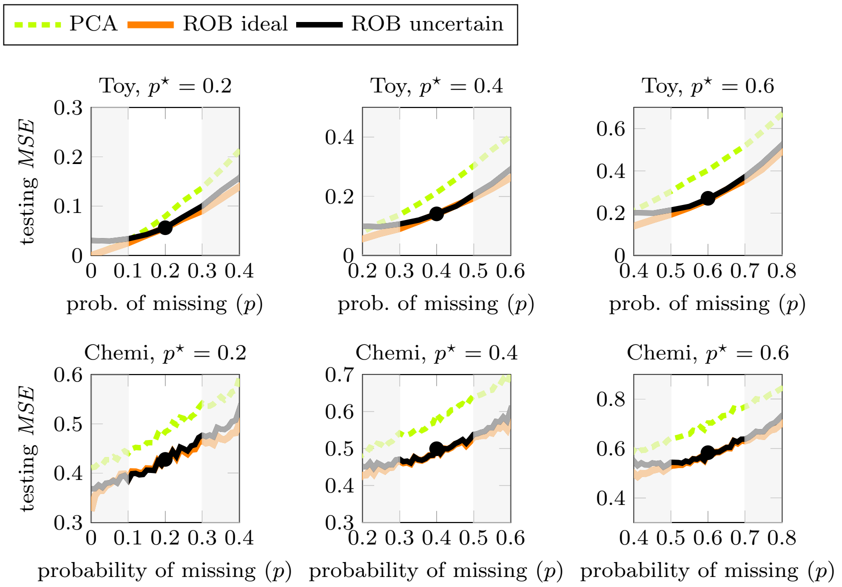


Fig. 3 Performance with noisy estimates of p. The white windows denote p within ±10 % of the assumed $p^\star$.

5.3 Different prior probabilities of missing values

The previous section experimentally investigated ROB regression assuming equal prior probabilities of missing values for all data sources. In this section we analyze the performance of the generic variant in Proposition 5, which assumes that each input data source can have a different prior probability of missing values, the probabilities are recorded in vector $\mathbf{p} = (p_1, p_2, \ldots, p_r)^T$.

5.3.1 Experimental protocol

We simulate missing values by randomly selecting a prior probability for missing values for each data source, $p_i \in [0, 1]$. We repeat the experiment 1,000 times, each time resampling the prior probabilities and then generating missing values according to those probabilities.

We test the same seven state-state-of-the-art models ALL, rALL, SEL, rSEL, PCA, rPCA, PLS. We compare the performance of ROB(p) and ROB$(\mathbf{p})$, where the former uses one fixed prior probability of missing value for all data sources, and the latter considers a different prior probability for each data source.

5.3.2 Results

Table 6 presents the results averaged over 1,000 runs. We see that ROB(p) with fixed prior probability outperforms all the competing methods on all the datasets, even though the assumption about fixed probability of missing is does not match the data. This happens, since the other methods (ALL, rALL, SEL, rSEL, PCA, rPCA and PLS) do not take missing data into consideration at all, this explains superior performance of ROB(p). The more generic version of ROB regression, ROB$(\mathbf{p})$, that takes into account different prior probabilities, outperforms ROB(p) and all the other regressions on all datasets. This is an expected and intuitive result, since ROB is the theoretically optimal solution, as we proved in Proposition 5.

5.4 Online ROB regression

Next we investigate the performance of online ROB regression (Algorithm 2). Our first goal is to analyze how the performance of online ROB regression, tailored to handling missing values, compares to the performance of the ordinary online regression (Algorithm 1). The second goal of this experiment is to investigate, how the performance of online ROB regression depends on the training sample size.

5.4.1 Experimental protocol

The experimental protocol is as follows. We split a dataset into two equal parts at random, and designate one part as a hold out testing set. In the testing set we introduce missing values at random with a fixed probability $p = 0.5$. The other half of data is used for sequential training, it contains no missing values. We present training observations one by one, the predictive model is updated with each observation recursively. After each update we test the performance on the holdout testing set.

We run this experiment on the eight datasets used in the previous experiments. To keep the focus we standardize the datasets offline before the experiment.

Table 6 Testing errors with different prior probabilities of missing data

	ALL	rALL	SEL	rSEL	PCA	rPCA	PLS	ROB(p)	ROB$(\mathbf{p})$
Catalyst	1.12	0.83	1.06	0.89	0.75	0.74	0.91	0.70	**0.69**
Chemi	2.79	1.37	1.23	0.96	0.71	0.71	2.57	0.65	**0.63**
Concrete	0.91	0.81	0.85	0.84	0.95	0.95	0.87	0.80	**0.79**
CPU	0.62	0.60	0.59	0.59	0.59	0.59	0.62	0.54	**0.52**
Gaussian	1.17	0.51	0.56	0.56	0.51	0.52	0.53	0.45	**0.41**
Protein	0.42	0.39	0.46	0.43	0.30	0.30	0.34	0.16	**0.13**
Wine	0.91	0.88	0.90	0.89	0.90	0.90	0.88	0.86	**0.85**
Year	0.60	0.60	0.61	0.61	0.58	0.58	0.60	0.48	**0.47**

The best performance is in bold

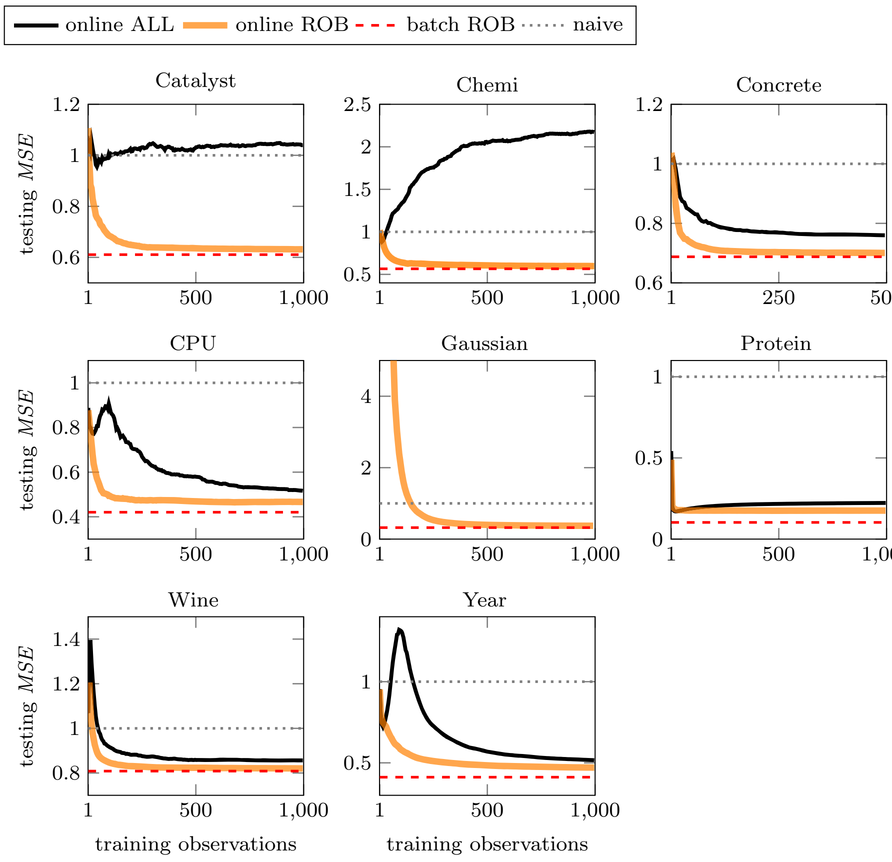
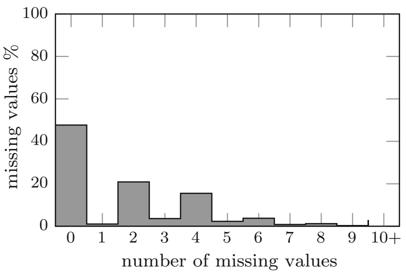


Fig. 4 Testing *MSE* versus training set size, averaged over 100 runs.

5.4.2 Results

Figure 4 plots the testing error as a function of training sample size seen. For clarity of exposition we plot only what happens in the first 1,000 training instances, since, as it can be seen from the plots, by reaching 1,000 instances learning stabilizes and later approach asymptotically the batch learning version. ALL is not visible in the Gaussian data plot, it performs by orders of magnitude worse than ROB. The plots demonstrate stable performance of ROB regression, the testing error consistently decreases as more training samples arrive, which should be expected, since more data allows to estimate model parameters more accurately. The ordinary regression (ALL) in all cases shows worse performance. Moreover, ALL sometimes even exhibits the opposite behavior, where testing error increases as more training observations arrive. This can be explained by the fact that the model is initialized by assuming that all the regression coefficients are equal to zero, which corresponds to the naive prediction by the target mean. Later on correlations between the input data are learned into the model and predictions become worse than naive, similarly to the Overfitted regression example that was presented in Fig. 1.

Fig. 5 Distribution of missing values in the case study data.



Overall, we can conclude from this experiment that the proposed online ROB consistently outperforms the ordinary online regression, and the predictive accuracy asymptotically approaches that of batch learning algorithm, as expected.

5.5 A case study in environmental monitoring

To validate the proposed algorithms ROBstream (Algorithm 4) we perform a case study in environmental monitoring with real missing values. The task is to predict the *level of solar radiation* from meteorological sensor data (such as temperature, precipitation, wind speed). We use a data stream recorded at SMEAR II station in Hyytiälä, Finland and delivered via web interface (Junninen et al. 2009).

5.5.1 Data and experimental protocol

We use data over a 5 years period (2007–2012), recorded every 30 min. from 39 meteorological sensors at one station. The data coming from the station has about 7 % of missing values. There is no single sensor that would provide non interrupted readings over those 5 years; for any sensor from 1 % up to 30 % values are missing. To simply capture seasonality we include in the input feature space a delayed target variable, which indicates the last known solar radiation (30 min ago). The distribution of missing values over the dataset is given in Fig. 5. The plot shows that about 50 % of the observations contain no missing values at all, the rest contain at least one missing value. The solar radiation (target variable) is available 99 % of the times, we eliminate from the experiment the instances having no target value.

The average probability of missing values over time is plotted in Fig. 6. We can see that the rate of missing values is evolving over time. A distinct increase is visible in the second half of the observation time. The evolution of probabilities motivates the need for adaptive handling of missing values, as proposed in this study.

Our final dataset contains 105 216 observations over 41 input variables, consisting of 39 raw sensor measurements, theoretical maximum radiation (calculated mathematically), and the last known value of the target variable. No prior standardization of data is used. Incoming data is standardized online following the procedure in Algorithm 3 with $\eta = 0.001$. In order to eliminate possible effects of autocorrelation in the data we work with the first order differences of the time series. The prediction accuracies are reported with respect to the original target variable.

For testing we use the test-then-train approach, that is common in evaluating data streams algorithms. Data is presented in the time order. When an observation arrives, first a prediction is made and recorded assuming that the true target value is unknown. Then the corresponding true target value arrives and the predictive model is updated with the latest observation.

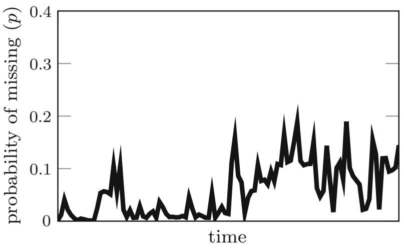


Fig. 6 Probability of producing missing values per data source over time.

Table 7 Average testing errors (MSE) on the case study data.

ALL	ALLimp	ROBstream	persistent	naive
2,954	2,450	**2, 408**	2,625	22,995

The best performance is in bold

We compare the performance of ROBstream (Algorithm 4) with the ordinary sequential regression (Algorithm 1) in two versions: ALL is incrementally trained on incoming observations that contain no missing values (the same way as ROBstream), and ALLimp is incrementally trained on all the observations, where missing values are replaced by the mean values.

For ROBstream the decay factor for covariance matrix estimates is fixed to $\alpha = 0.001$, and the decay for missing value rate estimates is fixed to $\gamma = 0.01$. The value of γ is larger than α, because we expect more rapid changes in the missing value situation (captured by γ), than changes in the data generating distribution (captured by α). We compare to a naive prediction, which always predicts the mean of the target variable, and a persistent prediction, which does not use any input data, but predicts the target value to be the same as the last observed target value.

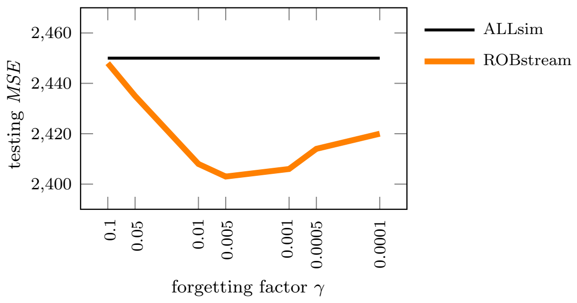
5.5.2 Results

Table 7 shows the testing errors MSE. We see that the intelligent predictors with incremental update (ALL, ALLimp and ROBstream) outperform the baseline by a large margin. However, the standard regression without missing value treatment performs worse than another baseline—persistent predictor. ALLimp that uses mean replacement and incrementally updates itself performs reasonably well, and ROBstream that reacts to changes in the missing value rates perform even better. The performance is consistent with our expectations.

In Fig. 7 we analyze the sensitivity of the performance to the choice of forgetting factor γ, when estimating missing value rates online. The performance of ROBstream is compared to the second best alternative ALLimp. We can see from the figure that the performance is not very sensitive to an exact choice of γ, and the performance remains competitive along all the spectrum of tested γ values.

In Fig. 8 we closer inspect prediction errors over time of ROBstream and the persistent predictor, which is a baseline for the performance. We see that the persistent predictor often makes much larger errors. The error curves have four distinct peaks. This is the daylight time in summer, hence the variance of the target variable is larger and there are more opportunities to make an error. We can see ROBstream shows consistently the best performance among the peer methods. The results support our theoretical findings.

Fig. 7 Sensitivity to parameter setting.



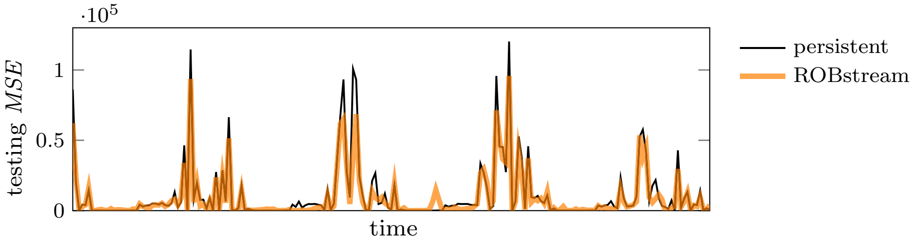


Fig. 8 A snapshot of predictions over time (case study data).

6 Related work

Our study is closely connected with handling missing data research, see e.g. Little (1992), Allison (2001), Alippi et al. (2012) and Ciampi et al. (2012). The main techniques are: imputation procedures where missing values are filled in and the resulting compete data is analyzed, reweighing procedures where instances with missing data are discarded or assigned low weights, and model-based procedures that define models for partially missing data. We investigate what happens *after* missing values are imputed during real-time operation using a very popular and practical mean value imputation. In our setting discarding streaming data is not suitable, since there would be continuous periods when we have no input data and thus no predictions. Model-based procedures could handle one-two missing sensors; however, when many sensors may fail, such a procedure is computationally impractical and likely infeasible, as we would need to keep an exponential number of models to account for all possible situations.

Handling missing values in regression is reviewed in (Little 1992). The majority of research focuses on training regression models from data with partially missing values. In our setting discarding some *training* data with missing values is not a problem, since the volumes of data are typically large. The problem arises during real-time operation. We not only need to input missing values, but also make the regression models fault tolerant. Hence, our work solves a different problem and is not directly comparable with missing value imputation techniques.

Topic-wise our work relates to fault tolerant control that is widely studied in industrial and aerospace systems (Patton 1997; Zhang and Jiang 2008). The main focus is on detecting the actual fault, not operating with a fault present. In our setting there is no fault in the system, just sensors fails, our model needs to remain accurate.

Redundancy in engineering duplicates critical components of a system to increase reliability, see e.g. Frank (1990). A common computational approach is to use an average the redundant sensors to reduce the impact of possible missing values. In fact, this is the effect we are aiming to achieve by minimizing the expected MSE. The main difference from our setting is in availability of backup sensors, it is even possible to install duplicate sensors on demand. In our setting; however, the data is given as is and we aim at exploiting it.

Robust statistics aims at producing models that are robust to outliers, or other small departures from model assumptions, see e.g. Huber (1981). The main idea is to modify loss functions so that they do not increase so rapidly, to reduce the impact of outliers. In our setting there are no large deviations in the input data due to missing data, but in fact the opposite. In our setting the variance of a missing sensor goes to zero. Hence, robust statistics approaches target a different problem.

Robust linear regression (Huber 1973) belongs to the robust statistics domain. Robust regression differs from our approach technically and conceptually. Conceptually, robust statistics solutions are aimed at reducing the contribution of individual instances (e.g. outliers). Our solution, similarly to ridge regression, does not prioritize instances in any way, the same treatment is applied to all instances. Technically, robust statistics solutions typically modify the form of the loss function, while we add a penalty component related to the resulting regression model.

Our theoretical analysis of the MSE resembles bias-variance analysis [see e.g. Geman et al. (1992)] in the way we decompose MSE into components. Regarding the connection of the bias-variance decomposition to the RR solution, we well know that enforcing strong regularization is likely to decrease variance and to increase bias. Further investigation is left for future work.

Finally, the setting relates to concept drift (Gama et al. 2014) and transfer learning (Pan and Yang 2010) settings in a sense that the training and the testing data distributions are different. However, in our setting there is no model adaptation during real-time operation.

7 Summary and conclusions

Systems relying on predictive models should be robust with regard to missing data, due to transient failures in the sensors, for instance. We focused on linear models for predictions, and theoretically analyzed the criteria for linear regression to be robust to missing values. We derived an optimization criteria for linear regression with missing values. Based on this optimization criteria we developed an algorithm for learning predictive models from streaming data with missing values. Our experiments with benchmark data and a case study in environmental monitoring domain confirmed the theoretical results and demonstrated the effectiveness of the proposed strategy.

Our analysis relies on theoretical variance and covariance of the prediction. Potentially it could be extended to higher order regression models (e.g. quadratic), that would require much more involved theoretical analysis due to interaction terms. Alternatively, one could obtain non linear prediction models by using the same linear regression with non-linear input features.

Acknowledgments We would like to thank the INFER project for sharing the Chemi dataset. This work has been supported by Academy of Finland Grant 118653 (ALGODAN).

Appendix: Proofs

In this appendix we provide proofs of Propositions 1, 2, 3, 4 , 5, 6, 7 and 8.

Proof *(Proof of Proposition 1)* From Eq. (4) the error can be decomposed into $E[MSE_0] = Var_0[o] - 2Cov_0[o, y] + Var_0[y] + (\bar{o} - y)^2$. Here MSE_0 denotes the error when no data is missing. We will analyze how missing data affects each component.

We can decompose the variance from Eq. (5) into $Var_0[o] = \beta^T \mathbf{D}\beta = \beta^T \beta + \beta^T(\mathbf{C}-\mathbf{D})\beta$, where $\mathbf{D} = diag(\mathbf{C})$.

Consider the first part $\beta^T \mathbf{D}\beta = \sum_{i=1}^{r} b_i^2 Var[x_i]$, which describes the variance of input variables. If an i^{th} data source fails, its variance becomes zero, $Var[x_i] = 0$. The probability of the i^{th} source not failing is $1 - p_i$, hence with missing values considered is

$$Var_{\mathbf{p}}[o](\text{part I}) = \sum_{i=1}^{r}(1 - p_i)b_i^2 Var[x_i] = \beta^T \mathbf{DH}\beta, \tag{9}$$

where $\mathbf{H} = \mathbf{I} - diag(\mathbf{p})$.

Now consider $\beta^T(\mathbf{C} - \mathbf{D})\beta = 2\sum_{i=1}^{r-1}\sum_{j=i+1}^{r} b_i b_j Cov(x_i, x_j)$, which describes the covariance between input variables. If either source i or j fails, then the covariance becomes zero, $Cov(x_i, x_j) = 0$. The probability that neither source i nor j fails, and the covariance does not become zero, is $(1 - p_i)(1 - p_j)$. Taking missing values into account the term becomes

$$Var_{\mathbf{p}}[o](\text{part II}) = 2\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}(1 - p_i)(1 - p_j)b_i b_j Cov(x_i, x_j) = \beta^T \mathbf{H}(\mathbf{C} - \mathbf{D})\mathbf{H}\beta. \tag{10}$$

Next we consider the covariance between the prediction and the target, which From Eq. (6) can be expressed as $Cov_0[o, y] = \sum_{i=1}^{r} b_i Cov[x_i, y] = \beta^T \mathbf{z}$. If the i^{th} data source fails, then the data from that source becomes constant, $x_i = \bar{x}_i$ and independent from the target y. As a result, the term becomes zero, $(E[yx_i] - \bar{y}\bar{x}_i) = E[y]\bar{x}_i - \bar{y}\bar{x}_i = \bar{y}\bar{x}_i - \bar{y}\bar{x}_i = 0$. For any term in the sum, the probability of not becoming zero, i.e. that data source not failing, is $1 - p_i$. Therefore, taking missing values into consideration

$$Cov_{\mathbf{p}}[o, y] = \sum_{i=1}^{r}(1 - p_i)b_i Cov[x_i, y] = \beta^T \mathbf{Hz}. \tag{11}$$

Missing input data does not affect the true target, thus

$$Var[y] = Var_0[y]. \tag{12}$$

The last term is not affected by missing values as well, since if a data source fails, then the values are replaced by the mean $x_i = \bar{x}_i$, but in this term the mean is used anyway.

$$(\bar{o} - \bar{y})^2 = (\sum_{i=1}^{r} b_i\bar{x}_i - \bar{y})^2 = (\bar{\mathbf{x}}\beta - \bar{y})^2. \tag{13}$$

After inserting the results from Eqs. (9), (10), (11), (12) and (13) into the expression for error in Eq. (4) we get

$$E[MSE_{\mathbf{p}}] = \beta^T \mathbf{HC}\beta + \beta^T \mathbf{H}(\mathbf{C} - \mathbf{D})\mathbf{H}\beta - 2\beta^T \mathbf{Hz} + Var_0[y] + (\bar{\mathbf{x}}\beta - \bar{y})^2.$$

□

Proof *(Proof of Proposition 3)* Since $\mathbf{p} = (p, p, \ldots, p)^T$, we can simplify $\mathbf{H}$ as

$$\mathbf{H} = \mathbf{I} - diag(\mathbf{p}) = \mathbf{I} - p\mathbf{I} = (1 - p)\mathbf{I}.$$

Substituting this expression into the equation in Proposition 2 gives

$$\begin{aligned}E[MSE^\star_p] &= \beta^T(1-p)\mathbf{I}\beta + \beta^T(1-p)\mathbf{I}(\mathbf{C}-\mathbf{I})(1-p)\mathbf{I}\beta - 2\beta^T(1-p)\mathbf{I}\mathbf{z} + 1\\ &= p(1-p)\beta^T\beta + (1-p)^2\beta^T\mathbf{C}\beta - 2(1-p)\beta^T\mathbf{z} + 1\end{aligned}$$

□

From Eq. 4, if data is standardized the error is $E[MSE_0] = Var_0 - 2Cov_0 + 1 = \beta^T\mathbf{C}\beta - 2\beta^T\mathbf{z} + 1$. The error with missing values can be expressed as

$$\begin{aligned}E[MSE^\star_p] &= p(1-p)\beta^T\beta + (1-p)^2\beta^T\mathbf{C}\beta - 2(1-p)\beta^T\mathbf{z} + 1\\ &= (1-p)\beta^T\mathbf{C}\beta - 2(1-p)\beta^T\mathbf{z} + (1-p) + p - p(1-p)\beta^T(\mathbf{C}-\mathbf{I})\beta\\ &= (1-p)E[MSE_0] + p - p(1-p)\beta^T(\mathbf{C}-\mathbf{I})\beta\end{aligned}$$

Proof (Proof of Proposition 2) The proof follows directly from Proposition 1. Since the input data is standardized, variances are equal to one, hence $\mathbf{D} = \mathbf{I}$, and $Var[y] = 1$. Moreover, the mean of the target is zero $\bar{y} = 0$, the means of the input variables are also zero, $\bar{x}_i = 0$ for $i = 1, \dots, r$, and $b_0 = \bar{y} = 0$, therefore $\bar{\mathbf{x}}\beta = 0$. With the terms in place we get

$$E[MSE^\star_\mathbf{p}] = \beta^T\mathbf{H}\beta + \beta^T\mathbf{H}(\mathbf{C}-\mathbf{I})\mathbf{H}\beta - 2\beta^T\mathbf{H}\mathbf{z} + 1.$$

□

Proof (Proof of Proposition 4) The Rayleigh quotient of the covariance matrix is defined as $\frac{\beta^T\mathbf{C}\beta}{\beta^T\beta}$, for non-zero $\beta \in \mathbb{R}^r$ and is bounded by the maximum and the minimum eigenvalues of $\mathbf{C}$, $\ell_{min} \le \frac{\beta^T\Sigma\beta}{\beta^T\beta} \le \ell_{max}$, where ℓ are eigenvalues, and takes the extreme values when β is equal to the corresponding eigenvectors.

Since $\mathbf{C}$ is a covariance matrix, all eigenvalues are non-negative and their sum is equal to the sum of the trace. As the data is standardized the sum of eigenvalues is r, hence the maximum eigenvalue does not exceed r, $0 \le \ell_{min} \le \frac{\beta^T\Sigma\beta}{\beta^T\beta} \le \ell_{max} \le r$. Algebraic manipulations give the bound

$$-\beta^T\beta \le \beta^T(\mathbf{C}-\mathbf{I})\beta \le (r-1)\beta^T\beta.$$

□

Proof (Proof of Proposition 5) In order to minimize the expected error we take the derivative of the *MSE* expression in Proposition 2 with respect to β.

$$\frac{\partial E[MSE]}{\partial\beta} = 2\mathbf{H}\beta + 2\mathbf{H}(\mathbf{C}-\mathbf{I})\mathbf{H}\beta - 2\mathbf{H}\mathbf{z} = 0.$$

Since data is standardized, the covariance and relation between inputs and target simplify to $\mathbf{C} = \mathbf{X}^T\mathbf{X}/n$ and $\mathbf{z} = \mathbf{X}^T\mathbf{y}/n$. Recall that $\mathbf{H} = \mathbf{I} - diag(\mathbf{p})$. Solving for β gives

$$\begin{aligned}&\mathbf{I}\beta + (\mathbf{C}-\mathbf{I})\mathbf{H}\beta - \mathbf{z} = 0,\\ &(\mathbf{I} + \mathbf{C}\mathbf{H} - \mathbf{H})\beta = \mathbf{z},\\ &(\mathbf{I} + \mathbf{C}(\mathbf{I} - diag(\mathbf{p})) - \mathbf{I} + diag(\mathbf{p}))\beta = \mathbf{z},\\ &(\mathbf{X}^T\mathbf{X}(\mathbf{I} - diag(\mathbf{p}))/n + diag(\mathbf{p}))\beta = \mathbf{X}^T\mathbf{y}/n,\\ &\beta = \left(\mathbf{X}^T\mathbf{X}(\mathbf{I} - diag(\mathbf{p})) + diag(\mathbf{p})n\right)^{-1}\mathbf{X}^T\mathbf{y}.\end{aligned}$$

□

Proof (Proof of Proposition 6) The proof follows directly from Proposition 5, where $\mathbf{p} = (p, p, \ldots, p)^T$. We substitute $diag(\mathbf{p}) = p\mathbf{I}$ into the expression in Proposition 5. Standard algebraic manipulations give the result:

$$\hat{\beta}_{ROB} = \left(\mathbf{X}^T\mathbf{X}(\mathbf{I} - p\mathbf{I}) + pn\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} = \left((1-p)\mathbf{X}^T\mathbf{X} + pn\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}.$$

□

Proof (Proof of Proposition 7) Let $\mathbf{X}_t$ be a matrix of all the observations up to time t, $\mathbf{X}_t = (\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_t^T)$. Let $\mathbf{y}_t$ be a vector of all the true target values up to time t, $\mathbf{y}_t = (y_1, y_2, \ldots, y_t)^T$. The covariance matrix can be expressed as

$$\mathbf{S}_t = \mathbf{X}_t^T\mathbf{X}_t = \mathbf{X}_{t-1}^T\mathbf{X}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T = \mathbf{S}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T.$$

Likewise,

$$\mathbf{X}_t^T\mathbf{y}_t = \mathbf{X}_{t-1}^T\mathbf{y}_{t-1} + \mathbf{x}_t y_t.$$

An offline OLS regression model estimated from all the data up to time t is given by $\mathbf{S}_t\beta_t = \mathbf{X}_t^T\mathbf{y}_t$. Substituting the recursive expressions defined above gives

$$\begin{aligned}
\mathbf{S}_t\beta_t &= \mathbf{X}_{t-1}^T\mathbf{y}_{t-1} + \mathbf{x}_t y_t\\
\mathbf{S}_t\beta_t &= \mathbf{S}_{t-1}\beta_{t-1} + \mathbf{x}_t y_t\\
\mathbf{S}_t\beta_t &= (\mathbf{S}_t - \mathbf{x}_t\mathbf{x}_t^T)\beta_{t-1} + \mathbf{x}_t y_t\\
\mathbf{S}_t\beta_t &= \mathbf{S}_t\beta_{t-1} + \mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1})\\
\beta_t &= \beta_{t-1} + \mathbf{S}_t^{-1}\mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1}).
\end{aligned}$$

Following standard matrix algebra the inverse can be computed as

$$\mathbf{S}_t^{-1} = (\mathbf{S}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T)^{-1} = \mathbf{S}_{t-1}^{-1} - \frac{\mathbf{S}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^T\mathbf{S}_{t-1}^{-1}}{1 + \mathbf{x}_t^T\mathbf{S}_{t-1}^{-1}\mathbf{x}_t},$$

which is the update equation given in Algorithm 1. □

Proof (Proof of Proposition 8) Let $\mathbf{X}_t$ be a matrix of all the observations up to time t, $\mathbf{X}_t = (\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_t^T)$. Let $\mathbf{y}_t$ be a vector of all the true target values up to time t, $\mathbf{y}_t = (y_1, y_2, \ldots, y_t)^T$. A covariance estimate can be expressed as

$$\begin{aligned}
\mathbf{S}_t &= (\mathbf{I} - diag(\mathbf{p}))\mathbf{X}_t^T\mathbf{X}_t + diag(\mathbf{p})t\\
&= (\mathbf{I} - diag(\mathbf{p}))\mathbf{X}_{t-1}^T\mathbf{X}_{t-1} + (\mathbf{I} - diag(\mathbf{p}))\mathbf{x}_t\mathbf{x}_t^T + diag(\mathbf{p})(t-1) + diag(\mathbf{p})\\
&= \mathbf{S}_{t-1} + (\mathbf{I} - diag(\mathbf{p}))\mathbf{x}_t\mathbf{x}_t^T + diag(\mathbf{p}) = \mathbf{S}_{t-1} + \mathbf{x}_t\mathbf{x}_t^T - diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I}).
\end{aligned}$$

Likewise,

$$\mathbf{X}_t^T\mathbf{y}_t = \mathbf{X}_{t-1}^T\mathbf{y}_{t-1} + \mathbf{x}_t y_t.$$

An offline ROB regression model estimated from all the data up to time t is given by $\mathbf{S}_t\beta_t = \mathbf{X}_t^T\mathbf{y}_t$. Substituting the recursive expressions defined above gives

$$\begin{aligned}
\mathbf{S}_t\beta_t &= \mathbf{X}_{t-1}^T\mathbf{y}_{t-1} + \mathbf{x}_t y_t \\
\mathbf{S}_t\beta_t &= \mathbf{S}_{t-1}\beta_{t-1} + \mathbf{x}_t y_t \\
\mathbf{S}_t\beta_t &= \left(\mathbf{S}_t - \mathbf{x}_t\mathbf{x}_t^T + diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I})\right)\beta_{t-1} + \mathbf{x}_t y_t \\
\mathbf{S}_t\beta_t &= \mathbf{S}_t\beta_{t-1} + \mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1}) - diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I})\beta_{t-1} \\
\beta_t &= \beta_{t-1} + \mathbf{S}_t^{-1}\mathbf{x}_t(y_t - \mathbf{x}_t^T\beta_{t-1}) - \mathbf{S}_t^{-1}diag(\mathbf{p})(\mathbf{x}_t\mathbf{x}_t^T - \mathbf{I})\beta_{t-1}.
\end{aligned}$$

□

References

Alippi, C., Boracchi, G., & Roveri, M. (2012). On-line reconstruction of missing data in sensor/actuator networks by exploiting temporal and spatial redundancy. In *Proceedings of the 2012 International Joint Conference on Neural Networks, IJCNN* (pp. 1–8).

Allison, P. (2001). *Missing data*. Thousand Oaks: Sage.

Brobst, S. (2010). Sensor data is data analytics' future goldmine. www.zdnet.com.

Ciampi, A., Appice, A., Guccione, P., & Malerba, D. (2012). Integrating trend clusters for spatio-temporal interpolation of missing sensor data. In *Proceedings of the 11th International Conference on Web and Wireless Geographical Information Systems, W2GIS* (pp. 203–220).

Frank, P. (1990). Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results. *Automatica*, *26*(3), 459–474.

Gama, J., & Gaber, M. (Eds.). (2007). *Learning from data streams: Processing techniques in sensor networks*. Heidelberg: Springer.

Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys, 46*(4) (in press).

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, *4*(1), 1–58.

Golub, G., & Van Loan, C. (1996). *Matrix computations*. Baltimore: Johns Hopkins University Press.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning: Data mining, inference, and prediction*. New York: Springer.

Hoerl, A., & Kennard, R. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *42*(1), 55–67.

Huber, P. (1981). *Robust statistics*. New York: Wiley.

Huber, P. J. (1973). Robust regression: Asymptotics, conjectures and monte carlo. *Annals of Statistics*, *1*(5), 799–991.

Jordan, M. (1998). Notes on recursive least squares. University of California, Berkeley. www.cs.berkeley.edu/~jordan/courses/294-fall98/readings/rls.ps.

Junninen, H., Lauri, A., Keronen, P., Aalto, P., Hiltunen, V., Hari, P., et al. (2009). Smart-SMEAR: On-line data exploration and visualization tool for smear stations. *Boreal Environment Research*, *14*, 447–457.

Little, R. (1992). Regression with missing X's: A review. *Journal of the American Statistical Association*, *87*(420), 1227–1237.

Little, R., & Rubin, D. (2002). *Statistical analysis with missing data* (2nd ed.). New York: Wiley.

Pan, S., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Patton, R. (1997). Fault-tolerant control: The 1997 situation. In *Proceedings of the 3rd IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes* (pp. 1033–1055).

Polikar, R., DePasquale, J., Syed Mohammed, H., Brown, G., & Kuncheva, L. I. (2010). Learn++.mf: A random subspace approach for the missing feature problem. *Pattern Recognition*, *43*(11), 3817–3832.

Qin, S. J. (1998). Recursive PLS algorithms for adaptive data modeling. *Computers & Chemical Engineering*, *22*(4/5), 503–514.

Scharf, L. L. (1990). *Statistical signal processing—Detection, estimation and time series analysis*. New York: Addison-Wesley.

Zhang, Y., & Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control*, *32*(2), 229–252.

Zliobaite, I., & Hollmen, J. (2013). Fault tolerant regression for sensor data. In *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases, ECMLPKDD* (pp. 449–464).