Harvest time prediction for batch processes

Spooner, Max Peter; Kold, David; Kulahci, Murat

Published in:
Computers & Chemical Engineering

Link to article, DOI:
10.1016/j.compchemeng.2018.05.019

Publication date:
2018

Document Version
Peer reviewed version

Citation (APA):
Highlights

- Many batch processes contain variability in the durations (harvest times) of batches
- A case study is presented of a bacteria fermentation batch process that displays harvest time variability
- A method is presented for predicting the harvest time at an early stage in the process using dynamic time warping and lasso regression
- Warping information from dynamic time warping is used to update the harvest time predictions online during the process
Harvest time prediction for batch processes

Max Spooner\textsuperscript{a,}\textsuperscript{*}, David Kold\textsuperscript{b}, Murat Kulahci\textsuperscript{a,c}

May 23, 2018

\textsuperscript{a}DTU Compute, Technical University of Denmark, Kgs. Lyngby, Denmark
\textsuperscript{b}Chr. Hansen A/S, Hvidovre, Denmark
\textsuperscript{c}Department of Business Administration, Technology and Social Sciences, Luleå University of Technology, Luleå, Sweden

\textsuperscript{*}Corresponding Author. Address: DTU Compute, Asmussens Alle 322, 2800 Kgs. Lyngby, Denmark. E-mail: mpsp@dtu.dk (M. Spooner)

Keywords: Batch Process, Prediction, Dynamic Time Warping, Partial Least Squares, Lasso Regression

Abstract

Batch processes usually exhibit variation in the time at which individual batches are stopped (referred to as the harvest time). Harvest time is based on the occurrence of some criterion and there may be great uncertainty as to when this criterion will be satisfied. This uncertainty increases the difficulty of scheduling downstream operations and results in fewer completed batches per day. A real case study is presented of a bacteria fermentation process. We consider the problem of predicting the harvest time of a batch in advance to reduce variation and improving batch quality. Lasso regression is used to obtain an interpretable model for predicting the harvest time at an early stage in the batch. A novel method for updating the harvest time predictions as a batch progresses is presented, based on information obtained from online alignment using dynamic time warping.
1 Introduction

Batch processes are characterised by a beginning, when the raw materials are loaded into a reactor vessel, a finite period of transformation or growth, and an end when the finished product is harvested from the reactor. The time at which to harvest the batch is often defined based on some features in the process which from experience ensure the desired product specifications. There is often batch to batch variation in the time at which the harvest criterion occurs, and so batches have different durations. This is especially the case in bio-based industrial processes where the harvest time is dependent on the activity of living organisms. In order to ensure the batch is harvested at the optimum point in time, it must be monitored closely by a technician who must react quickly when the harvest criterion is reached. In this work we consider the problem of predicting the harvest time at an early stage in the process. Obtaining good harvest time predictions is of value for two reasons. Firstly, such predictions provide a guide for the technicians on when to focus on the process and when it is safe to work on other tasks. Secondly, the predictions facilitate scheduling of downstream processes such as packaging.

The general problem of predicting some response variable based on data measured during a batch process has been investigated extensively. The predominant approach, Multi-way Partial Least Squares (MPLS), was pioneered by [1] where the method was used to make predictions of 5 quality variables using batch process data. This approach has been applied and adapted by several authors including [2], [3] and [4]. Besides MPLS, a wide range of machine learning methods have been applied to the problem of predicting end-of-batch quality using online process data including neural networks [5], support vector regression [6] and lasso regression [7].

There is limited research on batch process prediction where the response variable is harvest time of the batch rather than end-of-batch quality. In [8] MPLS is used to predict the end time of a batch based on only the first two hours of process data. Others apply more ad-hoc methods to detect the optimal fermentation time of a batch process [9].

In this paper we present a real case study of predicting the harvest time of a batch bacteria fermentation process for the bioscience company Chr. Hansen A/S using a novel statistical approach, and apply the methods to datasets from two different bacteria fermentations. A need for more accurate harvest time prediction was identified by the company. The process is characterised by two phases which influenced our approach for harvest time
prediction. In brief, the proposed method consists of predicting the harvest time at the end of the first phase using a lasso regression model, and then updating the predictions from this model online during the second process phase using the time-information provided by online dynamic time warping (DTW). An overview of the steps involved in the method is shown in Fig. 1. Each step is explained in detail in Section 2.

**Fitting model**

- For each batch extract phase 1 data
- Perform offline DTW. Unfold aligned variables (including warped time)
- Fit lasso model and obtain phase 1 predictions \( \hat{y}_{\text{phase 1}} \)
- Do online DTW during phase 2 of each batch
- Calculate mean time to reference at end of each batch, \( \Delta_y \)
- Use OLS regression to estimate the weights \( \alpha \) and \( \beta \) of Equation 8

**Applying model**

- Phase 1 complete?
  - no
  - yes
- Phase 2 in progress?
  - no
  - yes
- Perform offline DTW. Unfold aligned variables (including warped time)
- Apply lasso model to obtain phase 1 prediction \( \hat{y}_{\text{phase 1}} \)
- Perform online DTW using alignment variable up to current time point
- Calculate current value of mean time to reference batch, \( \Delta_k \) and predict harvest time using Equation 8

**Figure 1: Summary of model fitting and application**

The proposed method provides a valuable early warning of the expected harvest time to enable scheduling of downstream tasks. As a benchmark, we compare the phase 1 lasso model to a MPLS model, which is the more usual approach to dealing with the high dimensionality of the unfolded phase 1 data. We advocate the use of lasso regression for two main reasons. Firstly,
lasso regression results in a far simpler model than MPLS, as it automatically selects only a subset of variables for which to give non-zero model coefficients. The lasso model is therefore easier to interpret than the PLS model, which is a big advantage for implementation in an industrial setting. Secondly, the test error, estimated using nested cross validation [10], is found to be smaller for the lasso model than for the MPLS model. Therefore, in this case study, the lasso model is expected to provide more accurate harvest time predictions than the MPLS model when used on new batches.

In summary, the novel contributions of this work consist firstly, in the presentation of real data from an existing batch process and a real problem to be solved: that of predicting the harvest time. Secondly, we demonstrate the advantages of lasso regression as a variable selection method, in contrast to the more predominant latent structure method MPLS that retains all variables in the model regardless of their relevance. Thirdly, we present a novel method that updates the harvest time predictions using dynamic time warping. This method exploits directly the ability of DTW to indicate time information regarding a batch, whereas previous applications for batch processes has mostly used dynamic time warping as a preprocessing step.

2 Methods

2.1 Process

The fermentation process used by Chr. Hansen to produce bacteria cultures consists of the following steps

1. A small volume of concentrated bacteria cells is added to a fermenter vessel which has been pre-filled with growth medium
2. The bacteria cells grow and multiply, thereby producing acid which lowers the pH inside the fermenter
3. When pH reaches a predefined set point, the pH level is automatically controlled at the set point level by adding a base to the fermenter
4. Based on expert judgement the batch is stopped and the contents of the fermenter are transferred to downstream processing
Figure 2: The raw data for both products
Data for two different bacteria fermentations was obtained which we refer to as product 1 and product 2. For both products, 6 variables are measured during the fermentation: pH, Base Flow (rate of base addition), Base Quantity (total amount of base added), Temperature, Level and Pressure. However, our interest was limited to the first three of these variables because they are most closely linked to the biological process. The temperature variable was not used because it was tightly controlled and maintained at a constant level. The product 1 and product 2 data consisted of 44 and 43 batches respectively and were all taken from normal operating conditions. The data is shown in Figure 2 where the two phases of the process can be distinctly seen. In the first phase, pH is not controlled and is decreasing until it reaches the set point. In the second phase, base is added in order to maintain the pH at the set point. The observed changes in pH and Base Flow are closely linked to the state of the process and reflect rates of bacteria growth and metabolism. There is variation between batches in the time taken to reach different stages, as well as in the magnitudes of the variables, due to differences in raw materials and inoculation material.

The criterion to start the harvest is product specific and is based on a combination of process parameters. The harvest process should be initiated manually when the criterion is met, but may be delayed as it relies on human judgement and taking additional factors into account, such as whether equipment downstream is available.

Due to the above mentioned variations and equipment limitations the harvest is done inconsistently as reflected in Figure 2. Therefore, the aim of this work was to predict in advance the time at which the harvest criterion will be met. This will reduce the need for human evaluation thereby enabling less process variation and better planning in regards to utilization of the downstream equipment.

Batches which do not attain the harvest criterion cannot be used for model building, as for such batches the response variable, correct harvest time, is missing. Therefore, due to the batch to batch variation, the harvest criterion for each product was redefined to features attained by all batches so that it would be possible to assess the performance. The harvest criterion for product 1 was defined as the moment when Base Flow falls to 95 % of its maximum value. For product 2 the harvest criterion was defined to be when Base Quantity reaches 0.72 (in scaled units). These criteria are very similar to those used in practice by the company, but have the advantage of being attained by all the batches. Of course, an alternative approach would
be to use the harvest criteria actually in use, and use some missing data imputation method to fill in the correct harvest time values for batches which were harvested too soon. However, then assessment of model performance would depend on how the missing data are imputed. Therefore, we use the former approach so that there is no doubt that the harvest time to be predicted corresponds to the same criterion in all batches.

2.2 Phase 1 Models

The clear division of the process into two phases suggested a goal of predicting the harvest time at the end of the first phase. A prediction at this point is early enough to be useful for scheduling purposes, whilst enough data has been accumulated to make predictions realistic. In [11] approaches to statistical monitoring of multiphase and multistage batch processes are discussed and the idea of monitoring at different levels (phases/stages/entire batches) is presented. We suppose that data is available up to the end of the first phase, and use this data to train a model for predicting the harvest time. The model may then be used during future batches upon completion of the first phase in order to predict when the entire batch will be ready to harvest.

Each data set consists of $I$ batches, $J$ variables measured for $K_i$ observations where $K_i$ varies depending on the duration of phase 1 for the $i^{th}$ batch. The response variable is the $I \times 1$ vector of harvest times for the product.

2.2.1 Alignment

The methods we wish to apply require that each batch have the same number of observations. It is also preferable that events during the batch are synchronised [12, 13]. We applied dynamic time warping (DTW) to align the data. This dynamic programming algorithm, originally developed for speech recognition [14], has been used widely for alignment of batch process data [12, 15, 16] as well as in almost every other data analytic field concerned with time series data [17]. With DTW, a reference batch is selected and the other batches are aligned to this reference batch. The aligned batches all have the same number of observations as the reference batch, $K_{ref}$. In addition to the aligned variable trajectories, a warping function is obtained which represents the local batch time of each batch relative to the reference batch, and this warping function should be treated as an additional variable of the process.

As phase 1 is completed when the harvest time predictions are to be made,
we use the offline version of DTW in which end-point constraints enforce the first and last observation of each query batch to be aligned to the first and last observation respectively of the reference batch.

During phase 1, pH is the only variable closely related to the underlying process and is therefore used as the alignment variable. The reference batch is selected as the batch with duration closest to median duration. In order to avoid pathological warpings a local constraint as defined by [14] is used. These local constraints are represented by the parameter $P$ with $P = 0$ corresponding to no constraint on the warping path, and larger values restricting the warping path closer to the diagonal. The value of $P$ can be selected according to the methods in [13].

After alignment, each phase 1 data set is a three way data cube of $I$ batches by $J$ variables (now including the time warping function variable) by $K_{ref}$ observations.

### 2.2.2 MPLS

Since [1], MPLS has become a standard method for making prediction models from batch process data. In MPLS, the three-way batch process data is first converted to two way data so that the classic PLS method [18] can be applied. This unfolding consists of placing $K_{ref}$ many $I \times J$ time slices side by side to create the matrix $X (I \times JK_{ref})$. The response variable to be predicted (the final harvest time) forms the vector $y (I \times 1)$. Each column of $X$ is mean centred and scaled to unit variance. $y$ is also mean centred. We opt not to scale $y$ so that the errors of the model will be in the original units. PLS then deconstructs both $X$ and $y$ into scores and loadings in such a way that the covariance between the scores is maximised within the following outer relationship:

$$X = TP' + E \quad \text{and} \quad y = UQ' + F'$$  \hspace{1cm} (1)

$y$ is predicted by the scores of $X$ through the inner relationship

$$y = TBQ' + F$$  \hspace{1cm} (2)

where the algorithm minimises $||F||$. The different elements of the PLS decomposition are summarised in Table 1. The number of components (latent variables) in the PLS model, $n$, may be selected using cross validation. Problems of collinearity and overfitting may be overcome by selecting a limited number of components, $n$ to retain in the model. Cross validation is
used to select the value of $n$ which results in the smallest cross validation mean squared error, then this value is used to train the model on the entire data set.

Table 1: PLS and Lasso Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$(I \times JK_{\text{ref}})$ Unfolded process data</td>
</tr>
<tr>
<td>$y$</td>
<td>$(I \times 1)$ Response variable</td>
</tr>
<tr>
<td>$T$</td>
<td>$(I \times n)$ PLS scores of $X$</td>
</tr>
<tr>
<td>$U$</td>
<td>$(I \times n)$ PLS scores of $Y$</td>
</tr>
<tr>
<td>$P$</td>
<td>$(JK_{\text{ref}} \times n)$ PLS loadings of $X$</td>
</tr>
<tr>
<td>$Q$</td>
<td>$(1 \times n)$ PLS loadings of $y$</td>
</tr>
<tr>
<td>$E$</td>
<td>$(I \times JK_{\text{ref}})$ PLS errors of $X$</td>
</tr>
<tr>
<td>$F^*$</td>
<td>$(I \times 1)$ PLS errors of $y$ in outer relation</td>
</tr>
<tr>
<td>$F$</td>
<td>$(I \times 1)$ PLS errors of $y$ in inner relation</td>
</tr>
<tr>
<td>$B$</td>
<td>$n \times n$ PLS regression coefficients relating scores of $X$ to scores of $y$</td>
</tr>
<tr>
<td>$n$</td>
<td>Number of latent variables retained in PLS model</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$(JK_{\text{ref}} \times 1)$ Lasso coefficients</td>
</tr>
<tr>
<td>$e$</td>
<td>$(I \times 1)$ Lasso errors</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Lasso sparsity parameter</td>
</tr>
</tbody>
</table>

2.2.3 Multi-way Lasso

Another approach to dealing with the large numbers of correlated variables in batch process data, is to use a form of regularised regression such as lasso regression. Lasso regression has not been as widely applied to batch process data as PLS, but [19] developed a lasso framework for fault diagnosis, and [7] adapted lasso regression for multiphase batch processes. Lasso regression was developed by [20] and the method finds a linear model

$$y = X\beta + e$$  \hspace{1cm} (3)

where the coefficients $\beta$ are calculated based on the minimisation of

$$\min_{\beta} \frac{1}{I} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$  \hspace{1cm} (4)
where $||·||_1$ is the $L_1$-norm and $||·||_2$ the $L_2$-norm. $\lambda$ is a tuning parameter determining the overall influence of the $L_1$ penalty on the solution to the minimisation. The $L_1$-norm penalty has the effect of shrinking coefficients in $\beta$ to zero, and the greater the value of $\lambda$, the more sparse the coefficient vector will be.

$$\hat{\beta} = \arg\min_\beta \frac{1}{l} ||y - X\beta||_2^2 + \lambda ||\beta||_1$$

Again, as $X$ is a matrix, the three-way batch process data must be unfolded as described in the previous section. The unfolded matrix is centred and scaled. Cross validation is used to select the value of $\lambda$ which results in the smallest mean squared error, and the selected $\lambda$ value is used to train the model on the entire data set. Due to the regularisation, only a small subset of variables at specific times will be included in the model and so the model will be easy to interpret and apply.

2.2.4 Validation

In order to assess how a model will perform on future batches, as well as for comparison of different models, it is vital to obtain a reasonable estimate of the prediction error. When data is plentiful, the prediction error can be estimated by setting aside a test set for the final model. However, in this case we have data sets of only 44 and 43 batches. If, say 8 batches are set aside as a test set, then first of all the model will suffer due to being trained on a significantly smaller training data set, and second of all the prediction error will be highly dependent on which batches are chosen for the test set, resulting in a poor estimate of the prediction error. In addition the choice of test set may arbitrarily favour one modelling approach over another. Another approach often seen, is to use the cross validation error from the parameter selection as a prediction error estimate. However, this estimate is biased, as the error is directly minimised for selecting the parameter [10]. To correctly account for overfitting from model selection, a separate cross validation should be performed to estimate prediction error [21]. For each fold, the entire model selection procedure is performed (including an inner cross validation for selecting parameters). In this way, the prediction error estimate accounts for any overfitting due to parameter selection.
2.3 Phase 2 Models

After applying the methods in the preceding section, a model is obtained that predicts the final harvest time of a batch using process data from the first phase of a batch. In this section we present a method for updating the predictions during the second phase of the process. The classical approach [1, 15] to making predictions online as a batch progresses is as follows:

1. Offline model fitting: The model is fitted using offline batch process data from historical batches. A global alignment is performed to synchronise the process data and obtain same length batches. The desired model is fitted.

2. Online application of model: As a new batch progresses, first a partial alignment is performed between the data so far obtained for the ongoing batch and the complete reference batch. This alignment estimates the corresponding point in the reference batch currently reached by the ongoing batch and synchronises the existing ongoing batch data to this leading portion of the reference batch. Next, the future variable trajectories of the ongoing batch are predicted in order to obtain the necessary input data to use the model. The model is then applied to the combined actual and predicted ongoing batch data to predict the response variable.

In the case we are considering the response variable is harvest time and so would be explicitly contained in the synchronised set of complete offline batches. Therefore, fitting a model to predict the harvest time based on this data would not be useful. We take a different approach by focusing on the online alignment with dynamic time warping and fitting a model directly to the online warping information.

2.3.1 Online Alignment

The procedure for implementing DTW alignment online was presented in [15]. The problem consists of aligning a partially complete batch to a complete reference batch. This entails that the fixed endpoint constraint in the standard DTW algorithm (the constraint that the endpoint of the query trajectory has to be aligned to the endpoint of the reference trajectory) must be abandoned. Instead, an "open-ended" alignment must be performed between the current
section of ongoing batch and the complete reference batch. The open-ended DTW algorithm synchronises the query to that leading portion of the reference which minimises the accumulated distance between them. The resulting alignment estimates which point has been reached in the reference batch based on events so far in the query batch. For each new observation that arrives, a new partial alignment is performed to re-align the ongoing batch to the reference batch. This means, that past alignments can be completely revised as new information comes in. For example, at $t_{batch} = 15$ minutes into the ongoing batch, online alignment may identify the corresponding reference time then obtained as $t_{reference} = 18$ (in which case the ongoing batch may be said to be "running fast" relative to the reference batch, having completed 18 minutes of the reference trajectory in only 15 minutes). However, when the next observation is measured at say $t_{batch} = 16$, it may occur that the new alignment identifies the current reference time as $t_{reference} = 11$, revising the preceding alignment. This instability in online alignment reflects the obvious difficulty of not knowing the future behaviour of the ongoing batch. Each online alignment is a "best guess" based on information so far. To try to limit the instability of online alignments, we incorporated the following adjustments to the basic algorithm.

1. **Alignment Variable** In the second phase of the process, Base Flow is the variable which most closely reflects the biological state. However, DTW tends to align according to magnitude of trajectories rather than shape, especially in the online version where there is no endpoint constraint. There is substantial variation between batches in the magnitudes of the Base Flow curves. Figure 3 shows an example where the ongoing batch has generally smaller Base Flow than the reference. When Base Flow is used as the alignment variable, it aligns according to the magnitude of Base Flow regardless of the smaller shape properties of the curves which are known to be more informative of the biological state (e.g. the small sinusoidal wave pattern that often occurs around two thirds of the way up the curves). To overcome this magnitude problem, Derivative Dynamic Time Warping (DDTW) was used [22]. In this version of DTW, the derivative of the trajectory is used as the alignment variable resulting in a more shape based rather than magnitude based alignment. To calculate the derivative of Base Flow, we first applied an exponentially weighted moving average to Base Flow (with smoothing factor $\alpha = 0.5$) to reduce noise. Letting $x_i$
denote the $i^{th}$ sample of this smoothed Base Flow, then the derivative was estimated by:

$$\frac{d}{dt}(x_i) = \frac{1}{2} [x_i - x_{i-1} + \frac{1}{2}(x_i - x_{i-2})]$$  \hspace{1cm} (6)

2. **Local Constraint** In the offline alignment of the phase 1 data a local constraint with $P = 1$ was found to be appropriate. Therefore, this same local constraint was applied for the online alignment during phase 2.

3. **Global Constraint** In the case of open ended alignment where the current end point of the ongoing batch is free to be matched to any point in the reference, it is advisable to implement a global constraint in addition to the local constraint. In DTW, a global constraint limits the region in which the warping function may be found, i.e., it specifies a limited range of reference times which may be matched to each ongoing batch time. Following [15] we calculate the global constraint upper and lower bounds from the empirical range of the warping functions resulting from an offline alignment of the phase 2 data for all batches (see Figure 4). In the offline alignments, the warping functions are guided by the future knowledge of the endpoint constraint and are therefore more accurate. Using the resulting global constraints for online alignment of a new batch thereby incorporates knowledge of how past batches evolved in order to improve the online alignment.

Applying the above modifications to the online DTW algorithm results in greater stability with regards to identification of the current reference time reached during the ongoing batch as illustrated in Figure 5.

An alternative approach to dealing with the instability of online DTW was devised by [23] who presented a "relaxed greedy DTW" algorithm which essentially only updates the warping function within some window around the current batch time, and fixes in place the warping function found before this window. This approach leads to far less variability in the online warping function improving the false alarm rate in their monitoring case study. In our case we prioritise finding the best alignment to the reference at each time point, and therefore do not apply the relaxed greedy DTW method which does not allow a bad alignment to be revised substantially in response to new data. We only wish to limit these revisions within reasonable bounds.
Figure 3: Online (open-ended) alignment using regular DTW (left) and using Derivative DTW (right) where the ongoing query batch is known up to sample 112. The future path of the query batch is shown in grey.

Figure 4: Calculation of global constraints to use for online alignment (red) as the empirical range of the warping functions (black) from an offline alignment of the data.
Figure 5: For a single batch, all online warping functions are shown. Left: using DDTW without any local or global constraints. Right: Using DDTW with local constraint $P = 1$ and empirical global constraints. These measures encourage less revisions in the warping functions as evident by the lesser degree of branching in the right hand plot.
using the above safety measures, rather than remove the warping revisions completely.

2.3.2 Updating Predictions Online

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>Current time (observation number) of the ongoing batch</td>
</tr>
<tr>
<td>$f(k)$</td>
<td>Current reference time according to online DTW</td>
</tr>
<tr>
<td>$\Delta_k$</td>
<td>The time to reference batch ($= k - f(k)$)</td>
</tr>
<tr>
<td>$\overline{\Delta}_k$</td>
<td>Mean time to reference batch of all alignments up to time $k$</td>
</tr>
<tr>
<td>$\overline{\Delta}_y$</td>
<td>Mean time to reference batch of all alignments up to the end (harvest time) of the batch</td>
</tr>
<tr>
<td>$\hat{y}_{\text{phase1}}$</td>
<td>Predicted harvest time from lasso model at end of phase 1 of process</td>
</tr>
<tr>
<td>$\hat{y}_k$</td>
<td>Updated prediction at time $k$ of harvest time using DTW</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Weight of the phase 1 prediction in the online harvest time prediction</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Weight of the mean time to reference batch in the online harvest time prediction</td>
</tr>
</tbody>
</table>

In this section we present how to combine the online DTW information with the phase 1 predictions. The notation used is summarised in Table 2. Let $k$ denote the current observation number of the ongoing batch (i.e., the "real time"). Let $f(k)$ denote the corresponding current reference time determined from online DTW. Note that $f(k) \in \mathbb{R}$ and corresponds to the maximum reference time in the current online DTW warping function. Each online warping function must itself be monotonically increasing, but because the warping function is re-calculated for each new observation of the ongoing batch, $f$ which we call the "outer warping function", does not have to be monotonically increasing. Let $\Delta_k = k - f(k)$ denote the "time to reference
If $\Delta_k$ is negative, online DTW estimates that the batch is currently running faster than the reference batch, whilst positive $\Delta_k$ suggests that it is running slower. Next, we define the running average of time to reference batch as

$$\overline{\Delta}_k = \frac{1}{k} \sum_{l=1}^{k} \Delta_l \quad (7)$$

Hence, $\overline{\Delta}_k$ is a measure of how fast or slow the ongoing batch is relative to the reference batch on average up to time $k$. Finally, in order to combine the information provided by $\overline{\Delta}_k$ with the phase 1 predictions, we use the following linear model

$$\hat{y}_k = \alpha \hat{y}_{\text{phase 1}} + \beta \overline{\Delta}_k \quad (8)$$

Equation (8) defines the online harvest time prediction at time $k$ (denoted by $\hat{y}_k$) as the weighted sum of the prediction made at the end of phase 1 and the average time to reference so far. This means that if the ongoing batch progresses faster during the second phase, then $\overline{\Delta}_k$ will be negative and the phase 1 harvest time prediction will be reduced. To determine the precise amounts by which to weigh the phase 1 prediction and the online DTW information (denoted by the coefficients $\alpha$ and $\beta$ respectively) the following steps are taken:

1. For all existing batches, perform online alignments throughout phase 2 and calculate the value of $\overline{\Delta}_y$ which is the mean time to reference of all alignments up to the end (harvest time) of the batch.

2. Regress $y$ (the correct harvest time) against $\hat{y}_{\text{phase 1}}$ (phase 1 harvest time prediction) and $\overline{\Delta}_y$ to obtain the least squares estimates of the coefficients $\alpha$ and $\beta$.

To apply the model on a new batch, the following steps are taken:

1. Upon completion of the first phase of the batch, perform offline alignment and apply the phase 1 lasso model to obtain the phase 1 harvest time prediction $\hat{y}_{\text{phase 1}}$.

2. For each new observation during phase 2, perform online DTW alignment and calculate the value of $\overline{\Delta}_k$ then apply Equation (8) to obtain the updated harvest time prediction.
It is relevant to note that a harvest time prediction could be obtained from the online DTW alignment directly as, for example,

\[ \hat{y}_k = y_{ref} + \Delta_k \]  

where the harvest time of the ongoing batch is estimated as the harvest time of the reference batch, \( y_{ref} \), adjusted for the mean time to reference so far. For example, if the reference batch was harvested at \( y_{ref} = 425 \) and an ongoing batch is currently 10 time units faster than the reference batch on average (\( \Delta_k = -10 \)), then we predict the ongoing batch should be harvested at time \( \hat{y}_k = 415 \). We investigated this approach, but observed that even though the DTW information on its own provided a general indication of the speed of the ongoing batch relative to the reference batch, it was not powerful enough to provide accurate enough harvest time prediction. This was the reason for using the combined approach of Equation (8), which makes use of the DTW information according to the weight \( \beta \). As will be seen in Section 3, in the case study we consider, the resulting values of \( \alpha \) and \( \beta \) mean that the online predictions are dominated by the phase 1 prediction, plus only a small adjustment for the current mean time to reference. In other cases, if it is found that the online predictions are dominated by the DTW information (large \( \beta \), small \( \alpha \)), it may be worth considering the simplified version of Equation 9.

3 Results and Discussion

As the batches are not harvested consistently at the same point in the process, the first step was to cut off the data at the two harvest time criteria we defined. For product 1, this the data was cut off at the point where Base Flow falls to 95% of its peak value (Figure 6) resulting in harvest times ranging from 395 to 461 time units. For product 2, the data was cut off according to the criterion that total Base Quantity reaches 0.72 in standardised units resulting in harvest times between 223 to 259 time units.

Next, each dataset was split into two according to the two phases of the process. The first phase was defined as the period from the start of the batch until the last observation with Base Flow equal to zero, with the first non-zero Base Flow sample onwards as the second phase.

DTW was used to align the phase 1 data as described in section 2.2.1, with the "dtw" package [24] in R. The alignment scores from the local constraint
Figure 6: Data for product 1 cut off at the 95% base max harvest criterion and product 2 data after cutting the trajectories at the defined harvest time of Base Quantity = 0.72 in scaled units
selection procedure of [13] are shown in Figure 7 from which the $P = 1$ local constraint was chosen for product 1. The same result was obtained for product 2. Figure 8 shows phase 1 pH before and after alignment along with the DTW warping functions. After aligning the phase 1 data, every product 1 batch was 245 samples long, and every product 2 batch was 96 samples long. Then, for each product the aligned phase 1 data was unfolded. Only the pH variable and the warping function were included in the unfolding as Base Flow/Quantity are zero during phase 1. Thus the unfolded $X$ matrix for product 1 was $I = 44$ batches by $J = 490$ ”variables” (pH at reference times 1 to 245 and local batch time at reference times 1 to 245). The unfolded $X$ matrix for product 2 was $I = 43$ batches by $J = 192$ ”variables”.

For each product, the unfolded matrix was auto-scaled by subtracting the mean from each column and dividing by the standard deviation. However, note that in subsequent cross validations for model fitting and RMSE estimation, auto-scaling was repeated using only the relevant training data. The
response variable, \( y \) was centred but not scaled, so that RMSE calculations would be in original time units.

Using this processed phase 1 data, a Lasso model was fitted for each product as well as a PLS model as a benchmark (using the "glmnet" [25] and "pls" [26] R packages respectively). Hyper-parameters were selected using ten fold cross validation as shown in Figure 9 with \( \lambda = 1.28 \) and \( n = 5 \) resulting in smallest RMSE for product 1 and similarly \( \lambda = 0.56 \) and \( n = 4 \) for product 2. Using these parameter values the models are then fitted to the whole data sets resulting in 5 non-zero coefficients for the product 1 Lasso model and 10 non-zero coefficients for the product 2 Lasso model as shown in Figure 10. It is immediately seen that the main predictor of harvest time for product 1 was the batch time value at reference time 245 (in other words the duration of phase 1 as determined by the DTW alignment). However, the lasso model also selected pH at 5 critical times throughout phase 1. In contrast, the PLS model retained 5 components where each component includes a 450 element \( X \) loading vector (Figure 10). For product 2, the Lasso model is dominated by batch time at reference times 91 and 95, which again corresponds to the overall duration of phase 1 for product 2. The PLS model for product 2 consisted of 4 components. From these models the training RMSE was calculated as shown in Figure 11.

Although there was only a single informative variable during the first phase of the process, pH, it is still the case that this variable is measured
Figure 9: Cross validation results for selecting $\lambda$ in the lasso model (left) and the number of components in the PLS model (right).

at many time points throughout the batch and the inclusion of warped time as a second variable brings the number of columns in the design matrix to 490 and 192 for products 1 and 2 respectively. Therefore, multivariate methods (lasso/PLS) are still appropriate. Using the knowledge gained from the lasso model, further refinement could be carried out to build an even simpler phase 1 regression model based on phase 1 duration and selected pH summary statistics. However, this would be a manual procedure, and we choose to focus on the automatic variable selection of the lasso method. Note that the PLS model alone gives no indication of the existence of a much simpler model structure for this data.
Figure 10: Coefficients of the lasso model and X loadings of the PLS model
Figure 11: Training and prediction RMSE for the two phase 1 models for products 1 and 2

To estimate the prediction RMSE, a separate ten fold cross validation is performed. For each excluded fold, an inner ten fold cross validation is used to select the hyper-parameter so the error arising from each model fitting procedure is realistically estimated. The resulting estimated RMSE is shown in Figure 11. It was of interest that for both product 1 and product 2, the Lasso model resulted in the smallest prediction RMSE despite consisting of vastly fewer parameters than the PLS model. In addition, the Lasso model is easier to interpret. For these reasons, we chose the Lasso models for making predictions of harvest time at the end of phase 1.

The phase 1 Lasso model provides a useful prediction of the harvest time upon completion of the first phase of the process. Next, we wished to update the predictions in real time during the second phase. Online alignments were calculated for the phase 2 data of each batch following the methods in Section 2.3.1. Then, for each batch \( f \) (the outer warping function), \( \Delta_k \) (time to reference) and \( \overline{\Delta}_k \) (running average time to reference) were obtained. For a single batch, \( \Delta_k \) and \( \overline{\Delta}_k \) are shown in Figure 12. Note that these quantities could be monitored directly by technicians to provide insight on the speed of an ongoing batch relative to the reference batch. In order to provide explicit harvest time predictions, the model described in Section 2.3.2 was fitted using the phase 1 Lasso predictions and the final values of \( \overline{\Delta}_k \) of all...
batches. This resulted in the coefficients given in Table 3. These coefficient values essentially mean that the online predictions calculated as the phase 1 prediction ($\alpha \approx 1$ for both products) with the addition of $\Delta_k$ down-weighted according to $\beta$. This is logical because it means the predicted harvest time is increased when $\Delta_k$ is positive and the ongoing batch is running slow on average during phase 2 compared to the reference batch.

Performance of the online model was evaluated based on the RMSE of the harvest time predictions across batches over time since the start of phase 2 (Figure 13). Prediction RMSE was estimated using ten fold cross validation. For each fold, the phase 1 lasso model was refitted as well as the online phase 2 model. The results show that both training and prediction RMSE decreases as phase 2 progresses (in both cases by over 0.2 in Product 1, and over 0.5 in product 2). Besides improving the prediction of harvest time, the process of online alignment is valuable for adding insight into the progress of the batch during phase 2.

Figure 12: The time to reference ($\Delta_k$) and running average time to reference ($\bar{\Delta}_k$) during phase 2 for a given batch.
Table 3: Phase 2 model coefficients

<table>
<thead>
<tr>
<th></th>
<th>( \alpha ) (p-value)</th>
<th>( \beta ) (p-value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product 1</td>
<td>1.003 (( \sim )0)</td>
<td>0.177 (0.016)</td>
</tr>
<tr>
<td>Product 2</td>
<td>1.007 (( \sim )0)</td>
<td>0.343 (0.012)</td>
</tr>
</tbody>
</table>

Figure 13: Online RMSE
4 Conclusions

We have presented a case study where a batch process exhibited considerable variation in its duration, and where batches were not harvested consistently according to some harvest criterion. There was a need for a method to predict the harvest time at an early stage in the process to facilitate resource allocation and ability to harvest consistently. We proposed using a Lasso regression model at the end of the first phase in the process and this model was shown to make useful predictions for two different products. We compared the Lasso model to a PLS model (the default approach in batch process prediction), and the Lasso model was shown to be more parsimonious and easier to interpret whilst performing as well as the PLS model. The Lasso model provides useful predictions of batch harvest time at an early stage in the process. Furthermore, we presented a novel method for updating the harvest time predictions during the second phase of the process by using information from online DTW alignment. This method was shown to reduce the RMSE of the predictions during the second phase. The proposed methods may be readily applied to other batch processes to anticipate harvest time and thereby contribute to improved process efficiency and quality.

5 Acknowledgements

The research is partially funded by BIOPRO (www.biopro.nu) which is financed by the European Regional Development Fund (ERDF), Region Zealand (Denmark) and BIOPRO partners. We thank Chr. Hansen A/S for providing access to data and expertise on the production process.

References


