SPIRE
Sustainable Process Industry through Resource and Energy Efficiency

DISIRE

Grant Agreement no.	**636834**
Project Acronym	**DISIRE**
Project full title	**Integrated Process Control Based on Distributed In-Situ Sensors into Raw Materials and Energy**
Dissemination level	PU (after the embargo date)
Embargo date	5 May 2017
Date of delivery	5/6/2016
Deliverable number	D2.1
Deliverable name	Modeling and process control architectures for data-driven MPC
AL / Task related	T2.1
Authors	D. Herceg [IMTL], P. Sopasakis [IMTL], G. Georgoulas [LTU], M. Castaño [LTU], D. Piga [IMTL], R. Lindkvist [ABB], M. Tallfors [ABB], O. Aronsson [ABB], A. Arias [DCI], C. Herce [CIRCE], A. Godek (KGHM), A. Rozek (KGHM), R. Zimroz (WUT), R. Krol (WUT), W. Kawalec (WUT), L. Jurdziak (WUT), A. Wylomanska (WUT), J. Obuchowski (CUP), P. Stefaniak (CUP), M. Polak (CUP), P. Kruczek (CUP), J. Sokolowski (CUP), N. Saraf [ODYS], D. Bernardini [ODYS].
Contributors	IMTL, LTU, MEFOS, CIRCE, DOW, ABB, KGHM, CUP, WUT/PWR, ODYS
Keywords	Process Modelling, Process control, Streaming algorithms, Model predictive control, Stochastic control, Scenario-based analysis and control, Online identification
Abstract	The main objective of this document is to provide detailed technical specifications and guidelines regarding modelling, online system identification and propose control schemes. The presented work revolves around the industrial processes of DISIRE and is done in collaboration with WPs 5, 7 and 8. We propose and apply state-of-art streaming identification and data-driven control methodologies and always incorporate uncertainty in our analysis to operate processes in a fashion that is resilient to uncertainty.

Acronym	*Meaning*
ADMM	Alternating Directions Method of Multipliers
APC	Advanced Process Control
ASGD	Averaged Stochastic Gradient Descent
AUC	Area Under Curve
CA	Cellular Automata
CCS	Control Configuration Selection
COT	Coil Outer Temperature
CS	Compressed (or Compressive) Sensing
CSV	Comma Separated Values (File Format)
DB	Database
FBE	Forward-Backward Envelope
FBN	Forward-Backward Newton Algorithm
FBS	Forward-Backward Splitting
FIFO	First-In First-Out
FISTA	Fast Iterative Soft-Thresholding Algorithm
ISTA	Iterative Soft-Thresholding Algorithm
kNN	k Nearest Neighbours
KPI	Key Performance Indicator
LASSO	Least Absolute Shrinkage and Selection Operator
LDH	Load, Haul and Dump
LPG	Liquid Petroleum Gas
LPV	Linear Parameter-Varying (systems)
LSE	Least Squares Estimation
MPC	Model Predictive Control
MSVM	Multicategory Support Vector Machine
NLPGCF	Naphtha/LGP Cracking Furnace
N4SID	Nonlinear 4SID
PAT	Process Analytical Technologies
PCA	Principal Components Analysis
PER	Propylene Ethylene Ratio
PWA	Piecewise Affine
PWARX	Piecewise Affine Auto-Regressive with eXogenous inputs
RCS	Recursive Compressed (or Compressive) Sensing
RFID	Radio Frequency identification
RLP	Robust Linear Programming
ROC	Receiver Operating Characteristic
RPSN	Regularised Piecewise Smooth Newton
SCADA	Supervisory Control And Data Acquisition
SMOTE	Synthetic Minority Oversampling TEchnique
SMPC	Stochastic Model Predictive Control
SPH	Smooth Particle Hydrodynamics
4SID	Subspace State-Space System Identification
SVD	Singular Value Decomposition
SVM	Support Vector Machine
WBF	Walking Beam Furnace

Contents

Chapter 1

Introduction

1.1 Objectives

This report aims at delivering a sound theoretical and applied framework for the modelling and control of industrial processes and, in general, data-intensive processes, i.e., processes which produce a lot of data.

The main objective of this work is to define a modelling architecture and structured workflows which will serve as guidelines in future industrial applications. At the same time, we will provide algorithmic tools for data-driven modelling and control and online recalibration of the controllers using process analytics techniques. Our modelling approach is entirely data-driven and, to this end, we follow two approaches: black-box modelling, where we assume no structural knowledge of the underlying physical process and grey-box modelling where we commence from first principles to specify a structure for our model and then we use data to identify its parameters.

DISIRE involves two industrial processes which are relevant for our purposes: the walking beam furnace of MEFOS and the naphtha/LPG cracking furnace of DOW Chemical. Historical data have been obtained from both processes and have been properly curated. The existence of a dynamic simulator as a fast-running surrogate of the real process is essential for a reliable testing.

Great attention has been given to alternative and novel identification methodologies using linear and non-linear model structures. There are two schools in system identification – either for linear or nonlinear models – the traditional batch approach, where data are collected, pre-processed and used to build a model offline (the procedure may be repeated sporadically to detect changes in the model) and the emerging recursive approach, where data are merely stored in a buffer, a model is trained and as new data from the process arrive, the model is continuously updated. This report pursues both approaches and provides a comparative study.

The main objectives of this report can be summarized as follows:

1. To provide a precise technical problem statement for the above control problems
2. To derive dynamical models based on black and grey-box modelling techniques
3. To estimate optimal static maps using PAT data
4. To measure, visualise and characterise the modelling uncertainty
5. To define performance indicators based on the outcomes of WP1 and in collaboration with MEFOS and DoW Chemical. The performance indicators we will introduce will be

technical, economic and related to safety and the environment

6. To design MPC schemes based on stochastic process models to account for the uncertainty in modelling
7. To introduce new algorithmic and methodological techniques for control and identification and demonstrate their features

1.2 Contributions

1.2.1 Beyond the state of the art

In this report we present results which are beyong the state of the art and by doing so we try to offer a unique value proposition for the DISIRE industrial partners, but also a pan-European value proposition. We showcase that the use of data which can be easily collected from a process, can lead to the definition of a closed loop system which has the capacity to attenuate structural changes in the system dynamics (using adaptive identification techniques), changes in the probability distribution of the underlying uncertainty (using a scenario-based representation of the uncertainty) and control the system effectively and proactively (using stochastic model predictive control).

Moreover, we propose a unified framework for online system identification, control and process-level optimisation of

WP2 has produced the following dissemination material:

Journal articles

1. A. Bemporad, V. Breschi and D. Piga, “Identification of Piecewise Affine Models via Recursive Multiple Least Squares and Multicategory Discrimination,” submitted to Automatica on July, 1, 2015
2. S. Formentin, D. Piga, R. Toth and S. Savaresi, “Direct design of LPV controllers from data,” Paper provisionally accepted to Automatica, 2015
3. A. Ebadat, G. Bottegal, D. Varagnolo, B. Wahlberg, K.H. Johansson, “Regularized Deconvolution- based Approaches for Estimating Room Occupancies,” IEEE Transactions on Automation Science and Engineering (provisionally accepted) See http://goo.gl/5yVsWS
4. D. Varagnolo, F. Zanella, A. Cenedese, G. Pillonetto, L. Schenato, “Newton-Raphson Consensus for Distributed Convex Optimization,” IEEE Transactions on Automatic Control (accepted for publication) See http://goo.gl/c48usd

Refereed proceedings

1. P. Sopasakis, P. Patrinos and N. Freris, “Compressed sensing based on streaming data”, EUSIPCO: European signal processing conference, 2016, Accepted.
2. D. Varagnolo, G. Pillonetto, L. Schenato. “Auto-tuning procedures for distributed nonparametric regression algorithms,” European Control Conference, 2015 - see http://goo.gl/d0hnlR.
3. R. Carli, G. Notarstefano, L. Schenato, D. Varagnolo, “Distributed quadratic programming

under Asynchronous and Lossy Communications via Newton-Raphson Consensus," European Control Conference, 2015, See http://goo.gl/BBkuZx.
4. A. Ebadat, G. Bottegal, D. Varagnolo, B. Wahlberg, H. Hjalmarsson and K.H. Johansson. "Blind identification strategies for room occupancy estimation," European Control Conference, 2015 - See http://goo.gl/ePyD3Y
5. R. Lucchese and D. Varagnolo, "Networks cardinality estimation using order statistics," American Control Conference, 2015 - See http://goo.gl/ki2Sq5

Other A web page was created at dysco.imtlucca.it/rcs to summarise and present in a high-level manner our novel methodology for recursive compressed sensing.

1.2.2 Contributions of partners

WP2 has actively involved many partners of the DISIRE consortium who have contributed either to the theoretical efforts, and/or the statement of a sensible (modelling or control or other) problem formulation, and/or the development of algorithms, software and tools to address these problems. The contributions of the various partners are summarised in Table 6.1. It can be seen that this report is the outcome of the concerted efforts of most partners in DISIRE.

1.3 Structure

This document starts with a theoretical part (Chapter 2), where we report our theoretical contributions. These will then serve as the machinery with which we address the control applications of the project, namely the walking beam furnace of MEFOS (Chapter 4) and the naphtha/LPG cracking process of DOW Chemical (Chapter 5).

In Chapter 6 we present two modelling approaches, namely using FIFO models and cellular automata models, for the dynamics of material flow through ore bunkers. We extend the discussion towards how these theoretical tools can be verified and used in an industrial environment and we formulated and stated a control problem.

Chapter 6 provides full technical specifications for KGHM's Lubin mine and details on the development of dynamical models for the material flow through bunkers (using cellular automata) and the dynamics of the transportation of ore using conveyor belts. The report is concluded in Chapter 7 where we summarise the main contributions of this report and discuss how these can be used to improve the industrial case studies of DISIRE and from this perspective we provide a plan of future work.

Partner	*Contribution*
IMTL	IMTL was the WP leader and organised the exchange of know-how among industrial partners (MEFOS, DOW Chemical and KGHM), LTU which is a university and research institute and ABB, G-Stat and ODYS which are technological developers. IMTL derived models for the walking beam furnace, the most successful of which are reported here, analysed their predictive ability using the approach described in Section 4.2.1 and proposed the stochastic MPC formulation in Chapter 3. Further, IMTL organised weekly Skype meetings to promote the exchange of knowledge and experiences among the involved partners and finally compiled this report in LaTeX.
LTU	LTU participated actively in the development and analysis of uncertainty of nonlinear models such as (LS)SVM/SVR and studied the control configuration problem both from a theoretical perspective and for the case study of MEFOS (the walking beam furnace). Developed models for the combustion quality in terms of predicted averate Oxygen concentration.
MEFOS	MEFOS provided the data that were used to produce the results in Chapter 4 and assisted the development of those models enabling the understanding of crucial details about the process. The model structure and the grey-box approach were based on information provided by MEFOS.
DOW Chemical	DOW Chemical provided the data that were used to produce the results which are reported in Chapter 5. They also clarified the objectives of the control problem as a follow-up from deliverable reports D1.X. Reviewed Chapter 5
CIRCE	Confirmed the process specifications for the naphtha/LPG cracking furnace in Chapter 5. Both CIRCE and DOW Chemical assisted WP2 in compiling Table 5.2.
G-Stat	G-Stat preprocessed the data of DOW Chemical which were used in Chapter 5 by combining multiple CSV files with raw measurements provided by DCI.
ABB	Using the theory of cellular automata, constructued detailed models for the mass flow inside the ore bunkers. Simulated the dynamic behaviour of the ore bunkers using these models. Stated the control problem and its objectives and performance indicators in Chapter 6.
PWR	Developed and analysed the ore mass flow models in Section 6.3.5 using statistical modelling tools.
KGHM	Provided the detailed technical specification in Chapter 6 and collected the pertinent data.
ODYS	Reviewed Chapters 2, 3 and 6. Filtered the preprocessed DOW Chemical data provided by G-Stat for application in model identification (Section 5.1.2). Constructed models using custom identification algorithm in order to describe cracking furnace operation with simulation results indicating quality of predictions (Section 5.2).

Table 1.1: Contributions of DISIRE partners to this report.

Chapter 2

Novel methodologies in modelling and control

2.1 From data to models – a streaming approach

Time-invariance is often an essential assumption which effectuates great simplifications and allows for process models to be derived once offline following the so-called *batch* approach. This is though a double-edged sword: deriving a model from data is a an one-shot task, but the resulting model will be insensitive to systematic changes which may occur because of the natural wear of the involved mechanical parts and sensors. With the traditional batch approach, data which are gathered in real time contribute nothing to the original model's accuracy and predictive ability. What is more, the batch approach to model training is often far from trivial especially when the available data are of very large size (sometimes up to thousands of GB). But, there is an alternative approach which is promising for data-intensive processes and can be used to recalibrate the obtained process model in real time — this is the *streaming approach*.

In the first part of the presentation of the method we propose, we study the problem of online compressed sensing based on a stream of data; we term the proposed method *streaming* or *recursive compressed sensing* where we solve a LASSO (ℓ_1-regularized) optimization problem using the solution we obtained at the previous time instant. We then extend the proposed methodology to the case of online dynamical modeling which, in fact, turns out to be a specific case of the methodology we propose for compressed sensing.

2.1.1 Online compressed sensing using streaming data

In signal processing, continuous signals are typically sampled at discrete instances in order to store, process, and share. In doing so, signals of interest are assumed sparsely representable in an appropriately selected orthonormal basis, i.e., they can be reconstructed by storing few non-zero coefficients in the given basis. For instance, the Fourier basis is used for bandlimited signals – with prominent applications in communications – while wavelet bases are suitable for representing piecewise smooth signals, such as bitmap images. Traditionally, the celebrated Nyquist-Shannon sampling theorem suggests a sampling rate that is at least twice the signal bandwidth; however this rate may be unnecessarily high compared to the signal's *innovation*,

i.e., the minimum number of coefficients sufficient to accurately represent it in an appropriately selected basis.

Compressed Sensing (CS) [1, 2] is a relatively new sampling paradigm that was introduced for sampling signals based entirely on their innovation, and has become ever since a major field of research in signal processing and information theory. The major contribution of this framework is a lower sampling rate compared to the classical sampling theory for signals that have sparse representation in some fixed basis [3], with notable applications in imaging [4], such as MRI. Compressed sensing is a signal processing methodology for the reconstruction of sparsely sampled signals and it offers a new paradigm for sampling signals based on their innovation, that is, the minimum number of coefficients sufficient to accurately represent it in an appropriately selected basis. Compressed sensing leads to a lower sampling rate compared to theories using some fixed basis and has many applications in image processing [5], medical imaging and MRI [4], photography, holography [6], facial recognition [7], radio astronomy [8], radar technology [9] and more.

Compressed sensing can also be directly used for system identification as recently shown in [10]. For model identification we will follow a more straighforward approach which we present in Sectionn 2.1.2.

Compressed sensing. For a vector $x \in \mathbb{R}^n$ we define the ℓ_0 pseudo-norm as the cardinality of its support $||x||_0 := |\mathrm{supp}(x)|$, where the support is the set of non-zero entries $\mathrm{supp}(x) := \{i : x_i \neq 0\}$. A vector x is s-*sparse* if and only if it has at most s non-zero entries, $||x||_0 \leq s$.

Compression. CS performs linear sampling $y = Ax$, where $A \in \mathbb{R}^{m \times n}$. Compression is performed by obtaining $m \ll n$ measurements. The main result states that *any* s-sparse vector $x \in \mathbb{R}^n$, can be sampled by a *universal*[1] matrix A with $m = \Theta(s \log(\frac{n}{s}))$, and may then be reconstructed perfectly solely from (noiseless) measurements y [11]. For this purpose, matrix A needs to satisfy a certain *Restricted Isometry Property*: there exists $\delta_s \in [0, 1)$ sufficiently small so that:

$$(1 - \delta_s)||x||_2^2 \leq ||Ax||_2^2 \leq (1 + \delta_s)||x||_2^2 \tag{2.1}$$

holds for all s-sparse vectors x. For many practical applications, taking $m = 4s$ works well. The success of CS lies in that the sampling matrix can be generated very efficiently as a random matrix, e.g., having i.i.d. Gaussian entries $\mathcal{N}(0, 1/m)$, or i.i.d. Bernoulli entries taking values $\{-1, 0, 1\}$ with probabilities $1/6, 2/3, 1/6$ respectively. The Bernoulli matrix has the additional advantage of being sparse, which makes matrix-vector multiplications more economical. Note that in practice measurements are typically noisy, $y = Ax + w$.

[1] The matrix A is universal in the sense it may be used to sample any vector with no more than s non-zeroes, regardless of their positions.

Decompression. In order to retrieve the original vector x from noisy measurements $y = Ax + w$, one needs to solve the ℓ_1-regularized least squares problem:

$$\text{minimize } \tfrac{1}{2}\|Ax - y\|_2^2 + \lambda\|x\|_1, \tag{2.2}$$

where λ is the regularization parameter that controls the trade-off between sparsity and reconstruction error. This is best known as Least Absolute Selection and Shrinkage Operator (LASSO) in the statistics literature [12]. There are several results analyzing the reconstruction accuracy of LASSO; for example [13] states that if $w \sim \mathcal{N}(0, \sigma^2 I)$, $\min_{i\in\text{supp}(x)}|x_i| > 8\sigma\sqrt{2\log n}$, and we choose $\lambda = 4\sigma\sqrt{2\log n}$ then a solution $x^\star$ to (2.2) has the same support as x, and its non-zero entries the same sign with their corresponding ones of x, with high probability. Additionally, the ℓ_2 reconstruction error is proportional to the standard deviation of the noise σ [3].

Remark 1 (Algorithms for LASSO)**.** *LASSO can easily be recast as a quadratic program which can be handled by interior point methods [14]. Additionally, iterative algorithms have been developed specifically for LASSO; all these are inspired by proximal methods [15] for non-smooth convex optimization: FISTA [16] and SpaRSA [17] are accelerated proximal gradient methods [15], SALSA [18] is an application of the alternative direction method of multipliers (ADMM). These methods are* first-order *methods, in essence generalizations of the gradient method and feature* sublinear convergence*. The error defined as $G(\hat{x}(t)) - G(x^*)$ where $G(\cdot)$ is the objective function of LASSO, and $\hat{x}(t)$ is the estimate at iteration number t, decays as $1/t^2$ for FISTA, SpaRSA and SALSA. Here, we devise a proximal Newton-type method with substantial speedup exploiting the fact that its convergence rate is locally* quadratic *(i.e., goes to zero roughly like e^{-ct^2} at the vicinity of an optimal solution).*

Problem statement. The traditional CS framework is naturally *offline* and requires compressing and decompressing an entire given dataset at one shot. The Recursive Compressed Sensing [11, 19] was developed as a new method for performing CS on an infinite data stream. The method consists of successively sampling the data stream via applying traditional CS to sliding overlapping windows in a recursive manner. Consider an infinite sequence, $\{x_i\}_{i=0,1,\ldots}$. Let

$$S := \limsup_{j\to\infty} j^{-1}|\text{supp}(\{x_i\}_{i=0,\ldots,j-1})| \in [0, 1]$$

be the *average* sparsity. We define successive windows of length n:

$$x^{(i)} := \begin{bmatrix} x_i & x_{i+1} & \ldots & x_{i+n-1} \end{bmatrix}^\top \tag{2.3}$$

and take $s = Sn$ the average sparsity parameter for a window of length n. The sampling matrix $A \in \mathbb{R}^{m\times n}$ (where we may take $m = \Theta(s\log(\frac{n}{s}))$) is only generated once. Each window is compressively sampled given a matrix $A^{(i)}$:

$$y^{(i)} = A^{(i)}x^{(i)} + w^{(i)}.$$

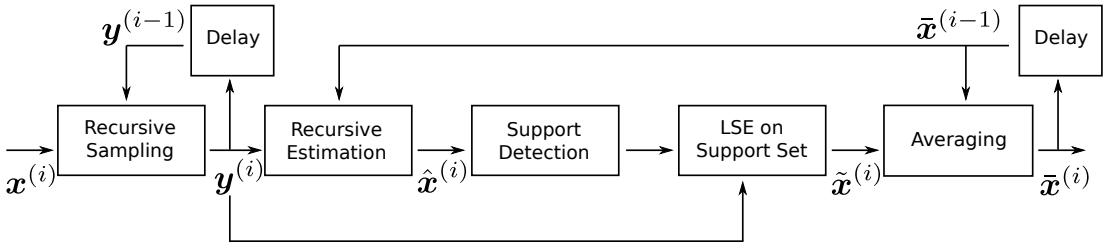

Figure 2.1: Architecture of the recursive compressed sensing algorithm: We propose a two-step estimation procedure for recovering the data stream: At first, we obtain the LASSO estimates which are fed into a de-biasing algorithm. For de-biasing, we estimate the signal support and then perform least squares estimation (LSE) on the support set in order to obtain estimates. The estimates obtained over successive windows are subsequently averaged.

The sampling matrix $A^{(i)}$ is recursively computed by $A^{(0)} = A$ and $A^{(i)} = A^{(i-1)}P$, where P is a permutation matrix which when right-multiplying a matrix cyclically rotates its columns to the left. This gives rise to an efficient recursive sampling mechanism for the input stream, where measurements for each window are taken via a rank-1 update [11].

For *decompression*, we need to solve the LASSO for each separate window:

$$\text{minimize } \tfrac{1}{2}\|A^{(i)}x^{(i)} - y^{(i)}\|_2^2 + \lambda\|x^{(i)}\|_1. \tag{2.4}$$

Note that the windows overlap, hence for a given stream entry, multiple estimates (one per each window that contains it) may be obtained. These estimates are then combined to boost estimation accuracy using non-linear support detection, least-squares debiasing and averaging with provable performance amelioration [11].

The overlap in sampling can further be exploited to speed-up the stream reconstruction: we use the estimate from a previously decompressed window to warm-start the numerical solver for LASSO in the next one. This simple idea provides a mechanism for efficient *recursive estimation*.

Formally, let $x^{(i)} = [\hat{x}_0^{(i)} \ \dots \ \hat{x}_{n-1}^{(i)}]^\top$ denote the optimal estimate obtained by LASSO in the i-th window, where we use $\hat{x}_j^{(i)}$ to denote the j-th entry of the $i-$th window (which according to our definitions corresponds to the $(i+j)$-th stream entry). In order to solve LASSO to obtain $\hat{x}^{(i+1)}$ we may use:

$$\hat{x}_{[0]}^{(i)} = \begin{bmatrix} \hat{x}_1^{(i-1)} & \hat{x}_2^{(i-1)} & \dots & \hat{x}_{n-1}^{(i-1)} & 0 \end{bmatrix}^\top,$$

as the starting point in the iterative optimization solver for LASSO in the $(i+1)-$th window, where $\hat{x}_j^{(i-1)}$, for $j = 1, \dots, n-1$, is the portion of the optimal solution based on the previous window; we set $\hat{x}_j^{(i-1)}, j = 0, 1, \dots, n-1$ to be the estimate of the $(j+1)$-th entry of the previous window, i.e., of x_{i-1+j}. The last entry $\hat{x}_{n-1}^{(i)}$ is set to 0, since we are considering sparse streams with most entries being zero.

Contribution. We devise a new numerical algorithm for solving LASSO based on the recently developed idea of proximal envelopes [20,21]. The new method demonstrates favorable convergence properties when compared to first order methods (FISTA, SpaRSA, SALSA, L1LS [14]);

in particular it has local quadratic convergence. Furthermore, this scheme is very efficient as each iteration boils down to solving a linear system of low dimension. Using this solver in RCS along with warm-starting leads to substantive acceleration of stream decompression. We verify this with a rich experimental setup.

Forward Backward Newton Algorithm. Splitting methods in general are optimization methods for problems which can be written in the general form

$$\text{minimize}\, \varphi(x) := f(x) + g(x), \tag{2.5}$$

where f and g are convex functions, f is typically a *loss function* and g is a *regularizer* or is used to encode constraints, as in $g(x) = \delta(x \mid C)$, where $\delta(\cdot \mid C)$ is the *indicator function* of a set C. The solution of these problems typically involves two steps: a gradient step which involves ∇f and a proximal step which requires computations involving the proximal operator of g, that is

$$\text{prox}_{\gamma g}(z) = \underset{x \in \mathbb{R}^n}{\arg\min} \left\{ g(x) + \tfrac{1}{2\gamma}\|x - z\|_2^2 \right\}. \tag{2.6}$$

The corresponding minimizer is the *Moreau envelope* of g, that is

$$g^{\gamma}(z) = \min_{x \in \mathbb{R}^n} \left\{ g(x) + \tfrac{1}{2\gamma}\|x - z\|_2^2 \right\}. \tag{2.7}$$

Many algorithms which follow this pattern have been proposed in the literature and are considered well-established and are typically based on operator splittings such as the Forward-Backward Splitting [22] and the Douglas-Rachford Splitting [23, 24]. Accordingly, many optimization methods are available in the literatures, such as AMM and ADMM. Tough they are simple to implement, they exploit the problem structure very well and are amenable to distributed implementations, they are very sensitive to preconditioning and can often be very slow as any first-order method (i.e., methods which use only gradient information).

In the well-known Forward-Backward Splitting (FBS) f is required to be proper, closed (lower semi-continuous), and convex with L_f-Lipschitz gradient and g needs to be proper, closed, convex and "prox-friendly," i.e., its proximal operator $\text{prox}_{\gamma g}$ should be easy to compute. The basic iteration of the algorithm is

$$x^{k+1} = \text{prox}_{\gamma g}(x^k - \gamma \nabla f(x^k)) := T_{\gamma}(x^k), \tag{2.8}$$

for $\gamma \in (0, 2/L_f)$. This algorithm is known to have a global convergence rate of $\mathcal{O}(1/k)$, while its accelerated variant (Nesterov's accelerate method) has a rate of $\mathcal{O}(1/k^2)$. Notice that it can be written as

$$x^{k+1} = \underset{z \in \mathbb{R}^n}{\arg\min} \{ \underbrace{f(x^k) + \langle \nabla f(x^k), z - x^k \rangle + \tfrac{1}{2\gamma}\|z - x^k\|^2}_{Q_{\gamma}^{f}(z; x^k)} + g(z) \}, \tag{2.9}$$

and it is illustrated in Figure 2.2.

This algorithm can often be particularly slow, even in its accelerated variant, as its practical

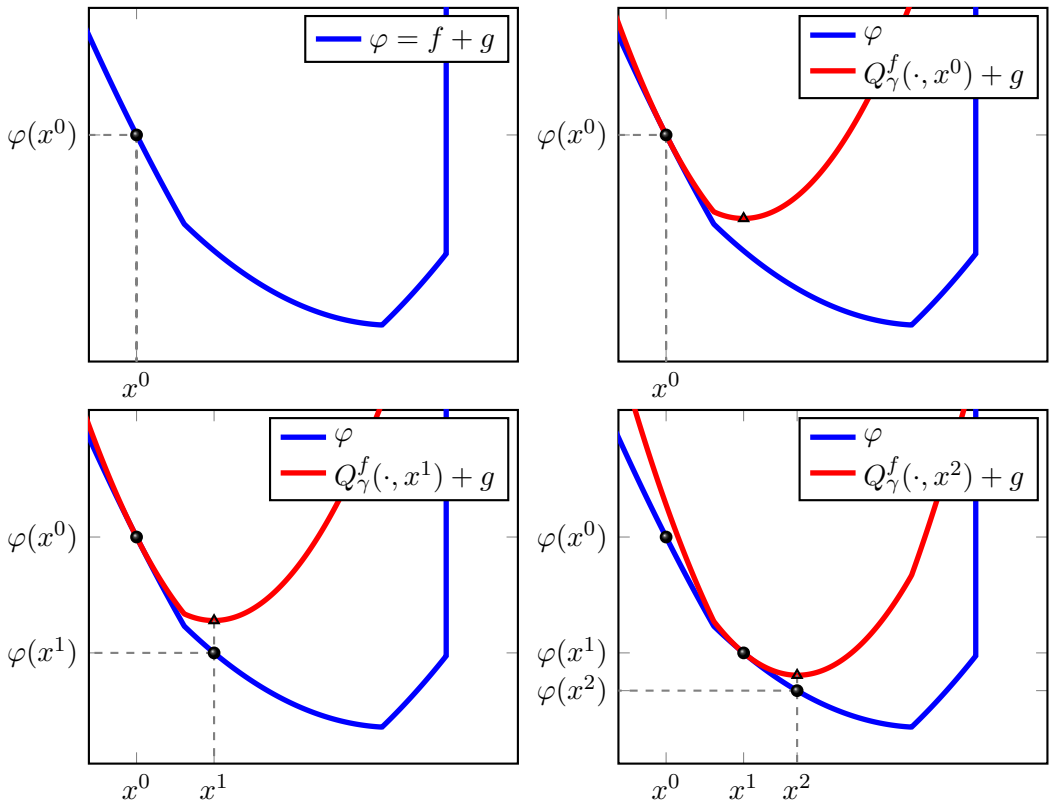

Figure 2.2: Illustration of the concept of the forward-backward envelope method. The blue line corresponds to the function $f + g$ which is to be minimized and the red line is the function $Q_\gamma^f(x; x^k)$

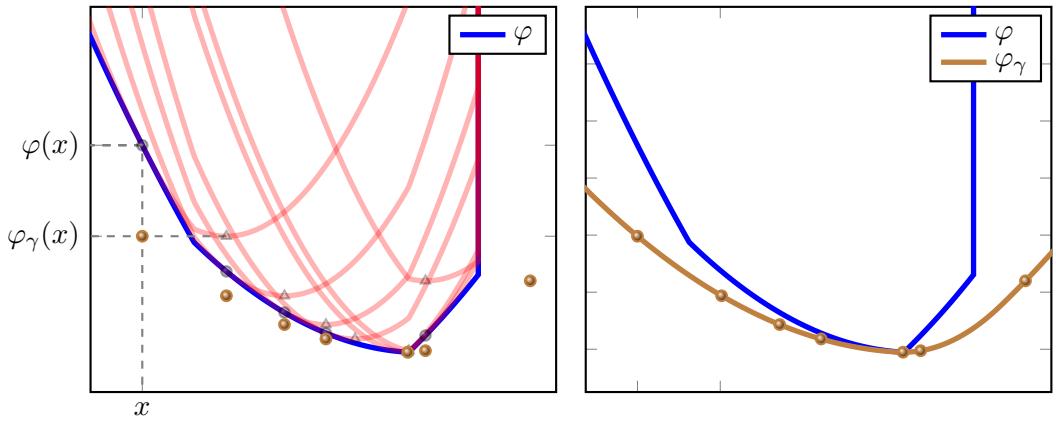

Figure 2.3: The forward-backward envelope function φ_γ.

speed is strongly affected by the preconditioning, which cannot always improve the speed of convergence to a satisfactory level. To remedy this shortcoming, we may use a better quadratic model for f as follows

$$x^{k+1} = \underset{z\in\mathbb{R}^n}{\arg\min}\{f(x^k) + \langle \nabla f(x^k), z - x^k\rangle + \tfrac{1}{2\gamma}\|z - x^k\|^2_{B_k} + g(z)\}, \tag{2.10}$$

where B_k is $\nabla^2 f(x^k)$ or an approximation thereof. This approach, however, comes with the limitation that there is no closed form for the solution of the minimization problem at each iteration. The Forward-Backward Envelope method combines the simplicity of FBS with the merits of second order methods.

We define the forward-backward envelope (FBE) of φ to be the value function of the FBS minimization problem

$$\varphi_\gamma(x) = \min_{z\in\mathbb{R}^n}\{f(x^k) + \langle \nabla f(x^k), z - x^k\rangle + \tfrac{1}{2\gamma}\|z - x^k\|^2 + g(z)\}, \tag{2.11}$$

for $\gamma \leq 1/L_f$. Function φ_γ is presented in Figure 2.3.

FBE possesses certain favorable properties such as that it is a lower bound of φ

$$\varphi_\gamma(x) \leq \varphi(x) - \frac{\gamma}{2}\|R_\gamma(x)\|^2, \tag{2.12a}$$

where $R_\gamma(x) = \gamma^{-1}(x - T_\gamma(x))$ is the *fixed-point residual* of the problem and T_γ as in (2.8). Furthermore, φ_γ satisfied the following inequality

$$\varphi_\gamma(x) \geq \varphi(T_\gamma(x)) + \frac{\gamma}{2}(1 - \gamma L_f)\|R_\gamma(x)\|^2. \tag{2.12b}$$

From the above two equations we may deduce that φ and φ_γ share the same minimizers, that is

$$\min \varphi = \min \varphi_\gamma, \tag{2.13a}$$

$$\arg\min \varphi = \arg\min \varphi_\gamma. \tag{2.13b}$$

FBE can be conveniently expressed as

$$\varphi_\gamma(x) = f(x) - \frac{\gamma}{2}\|\nabla f(x)\|^2 + g^\gamma(x - \gamma\nabla f(x)). \tag{2.14}$$

The Moreau envelope of a function g serves as a smooth approximation of g (which may be non-smooth). In particular, as γ becomes smaller and smaller, g^γ approximates g from below and is smooth with gradient

$$\nabla g^\gamma(x) = \gamma^{-1}(x - \mathrm{prox}_{\gamma g}(x)). \tag{2.15}$$

It can now be easily observed that φ_γ inherits the smoothness properties of g^γ so long as f is twice continuously differentiable. Then, we have that

$$\nabla\varphi_\gamma(x) = \gamma^{-1}(I - \gamma\nabla^2 f(x))(x - \mathrm{prox}_{\gamma g}(x - \gamma\nabla f(x))) \tag{2.16}$$

$$= (I - \gamma\nabla^2 f(x))R_\gamma(x). \tag{2.17}$$

Despite the fact that φ_γ is globally continuously differentiable, it cannot be assumed that it is $\mathcal{C}^2$. It actually fails to be twice continuously differentiable in most cases because of the fact that g is typically a nonsmooth function. Since the Hessian of φ_γ cannot be assumed to exist, we need to resort to generalized notions of differentiability such as the Bouligand (B) differentiability and the Clarke (C) differentiability [25].

Application of the algorithm. Let $f(x) = \frac{1}{2}\|Ax - y\|_2^2$, $g(x) = \lambda\|x\|_1$. Then $x^\star$ is optimal for

$$\text{minimize } \varphi(x) = f(x) + g(x) \tag{2.18}$$

if and only if it satisfies

$$-\nabla f(x^\star) \in \partial g(x^\star) \tag{2.19}$$

where $\partial g(x)$ is the subdifferential of g at x, defined by:

$$\partial g(x) = \left\{ v \in \mathbb{R}^n \mid g(w) \geq g(x) + v^\top(w - x), \forall w \in \mathbb{R}^n \right\}, \tag{2.20}$$

Since g is a multiple of the ℓ_1-norm, $v = (v_1, \ldots, v_n) \in \partial g(x)$ is equivalent to which, in our case, is given by: $v_i = \lambda\,\mathrm{sign}(x_i)$, for $x_i \neq 0$ and $|v_i| \leq \lambda$, for $x_i = 0$. Therefore the optimality conditions (2.19) for the LASSO problem (2.18) become

$$-\nabla_i f(x^\star) = -\lambda\,\mathrm{sign}(x_i^\star), \qquad \text{if } x_i^\star \neq 0, \tag{2.21a}$$

$$|\nabla_i f(x^\star)| \leq \lambda, \qquad \text{if } x_i^\star = 0, \tag{2.21b}$$

Suppose for a moment that we knew the partition of indices α and β corresponding to the nonzero and zero components of an optimal solution $x^\star$ respectively, as well as the signs of the nonzero components. Then we would be able to compute the nonzero components of $x^\star$ by

solving the following linear system corresponding to (2.21a):

$$A_\alpha^\top A_\alpha x_\alpha^\star = A_\alpha^\top y - \lambda \operatorname{sign}(x_\alpha^\star). \tag{2.22}$$

Notice that the support set α of $x^\star$ is much smaller compared to its dimension n. Hence, provided that α (as well as the signs of non-zero entries) has been identified, the problem becomes very easy. Roughly speaking, the algorithm we will develop can be interpreted as a fast procedure for automatically identifying the partition $\{\alpha, \beta\}$ corresponding to an optimal solution by solving a sequence of linear systems of the form (2.22). On the other hand, it can be seen as a Newton method for solving the following reformulation of the optimality conditions (2.19):

$$x = \operatorname{prox}_{\gamma g}(x - \gamma \nabla f(x)), \tag{2.23}$$

where $\operatorname{prox}_{\gamma g}$ is the *proximal mapping* which we introduced previously and γ is taken smaller than the Lipschitz constant of f, i.e., $\gamma < 1/\|A\|^2$. To see that (2.19) and (2.23) are equivalent $-\nabla f(x) \in \partial g(x) \iff x - \gamma \nabla f(x) \in x + \gamma \partial g(x) \iff x = (I + \gamma \partial g)^{-1}(x - \gamma \nabla f(x)) = \operatorname{prox}_{\gamma g}(x - \gamma \nabla f(x))$. In the case where $g = \lambda \|\cdot\|_1$, $\operatorname{prox}_{\gamma g}$ is the soft-thresholding operator:

$$(\operatorname{prox}_{\gamma g}(z))_i = \operatorname{sign}(z_i)(|z_i| - \gamma\lambda)_+, \quad i = 1, \ldots, n. \tag{2.24}$$

The iterative soft-thresholding algorithm (ISTA) is a fixed point iteration for solving (2.23). In fact, it is just an application of the well known forward-backward splitting technique for solving (2.19). On the other hand, FISTA is an accelerated version of ISTA where an extrapolation step between the current and the previous step precedes the forward-backward step [16, 26].

To motivate our algorithm, instead of a fixed point problem, we view (2.23) as a problem of finding a zero of the so-called *fixed point residual*:

$$R_\gamma(x) := x - \operatorname{prox}_{\gamma g}(x - \gamma \nabla f(x)). \tag{2.25}$$

One then would be tempted to apply Newton's method for finding a root of (2.25). Unfortunately the fixed point residual is not everywhere differentiable, hence the classical Newton method is not well-defined. However, it is well known that R_γ is nonexpansive, hence globally Lipschitz continuous [21]. Therefore the machinery of nonsmooth analysis can be employed to devise a generalized Newton method for $R_\gamma(x) = 0$, namely the *semismooth* Newton method. Due to a celebrated theorem by Rademacher, Lipschitz continuity of R_γ implies almost everywhere differentiability. Let $\mathcal{F}$ stand for the set of points where R_γ is differentiable. The B-differential of the nonsmooth mapping R_γ at x is defined by

$$\partial_B R_\gamma(x) := \left\{ B \in \mathbb{R}^{n \times n} \;\middle|\; \begin{array}{l} \exists \{x_n\} \in \mathcal{F} : x_n \to x, \\ R_\gamma'(x_n) \to B \end{array} \right\}. \tag{2.26}$$

If R_γ is continuously differentiable at a point $x \in \mathbb{R}^n$ then $\partial_B R_\gamma(x) = \{R_\gamma'(x)\}$. Otherwise $\partial_B R_\gamma$ may contain more than one elements (matrices in $\mathbb{R}^{n \times n}$). The semismooth Newton

method for solving $R_\gamma(x) = 0$ is simply

$$x^{k+1} = x^k - H_k^{-1} R_\gamma(x^k), \qquad H_k \in \partial_B R_\gamma(x^k). \tag{2.27}$$

Since in the case of LASSO the fixed point residual is piecewise affine, it is also strongly semismooth. Provided that solution $x^\star$ of (2.18) is unique (which is the case if for example the entries of A are drawn Gaussian i.i.d [27]) and that the initial iterate x^0 is close enough to $x^\star$, the sequence of iterates defined by (2.27) is well defined (any matrix in $\partial_B R_\gamma(x^k)$ is nonsingular) and converges to $x^\star$ at a quadratic rate, *i.e.*, $\limsup_{k\to\infty} \frac{\|x^{k+1}-x^\star\|}{\|x^k-x^\star\|^2} < \infty$ [21]. In fact, since the fixed point residual is piecewise affine for the LASSO problem, it can be shown that (2.27) converges in a *finite number of iterations*, in exact arithmetic. Specializing iteration (2.27) to the LASSO problem, an element H_k of $\partial_B R_\gamma(x_k)$ takes the form $H_k = I - P_k(I - \gamma A^\top A)$. Here P_k is a diagonal matrix with $(P_k)_{ii} = 1$ for $i \in \alpha_k$ and $(P_k)_{ii} = 0$, for $i \in \beta_k$, where

$$\alpha_k = \{i \mid |x_i^k - \gamma \nabla_i f(x^k)| > \gamma\lambda\}, \tag{2.28a}$$

$$\beta_k = \{i \mid |x_i^k - \gamma \nabla_i f(x^k)| \leq \gamma\lambda\}. \tag{2.28b}$$

Computing the Newton direction amounts to solving the so-called *Newton system*, $H_k d^k = -R_\gamma(x^k)$. Taking advantage of the special structure of P_k and applying some permutations, this simplifies to

$$d_{\beta_k}^k = -(R_\gamma(x^k))_{\beta_k}, \tag{2.29a}$$

$$\gamma A_{\alpha_k}^\top A_{\alpha_k} d_{\alpha_k}^k = -(R_\gamma(x^k))_{\alpha_k} - \gamma A_{\alpha_k}^\top A_{\beta_k} d_{\beta_k}^k. \tag{2.29b}$$

Taking into consideration (2.24), we obtain

$$(R_\gamma(x))_i = \begin{cases} \gamma(\nabla_i f(x) + s_i(x)\lambda), & i \in \alpha, \\ x_i, & i \in \beta, \end{cases} \tag{2.30}$$

where $s_i(x) = \operatorname{sign}(x_i - \gamma \nabla_i f(x))$, and α, β are defined using (2.29) (dropping index k). Therefore, after further rearrangement and simplification, the Newton system becomes

$$d_{\beta_k}^k = -x_{\beta_k}^k, \tag{2.31a}$$

$$A_{\alpha_k}^\top A_{\alpha_k}(x_\alpha^k + d_{\alpha_k}^k) = A_{\alpha_k}^\top y - \lambda s_{\alpha_k}(x^k). \tag{2.31b}$$

Summing up, an iteration of the semismooth Newton method for solving the LASSO problem takes the form $x^{k+1} = x^k + d^k$, which becomes:

$$x_{\beta_k}^{k+1} = 0, \tag{2.32a}$$

$$x_{\alpha_k}^{k+1} = (A_{\alpha_k}^\top A_{\alpha_k})^{-1}(A_{\alpha_k}^\top y - \lambda s_{\alpha_k}(x^k)). \tag{2.32b}$$

Note a connection with (2.22): the index sets α_k, β_k serve as estimates for the nonzero and zero components of $x^\star$.

The second block of equations describes the optimality conditions for

$$\text{minimize } \tfrac{1}{2}\|A_\alpha z - y\|_2^2 + c^\top z$$

where $z = x_\alpha + d_\alpha$ and $c = s_\alpha(x)\lambda$. This problem has a solution if and only if

$$A_\alpha^\top A_\alpha d = 0 \implies c^\top d = 0$$

This is true if and only if $c \in \text{range}(A_\alpha^\top A_\alpha)$. Therefore, solving the Newton system amounts to solving

$$A_\alpha^\top A_\alpha z = A_\alpha^\top y - c$$

and setting $d_\alpha = z - x_\alpha$, $d_{\alpha_c} = -x_{\alpha_c}$. There are two obstacles that we need to overcome before we arrive at a sound, globally convergent algorithmic scheme for solving the LASSO problem. The first one is that the semismooth Newton method (2.32) converges only when started close to the solution. If the initial iterate x^0 is far from the solution then (2.32) might produce a divergent sequence. The second obstacle concerns the fact that (2.32) might not be well defined, in the sense that A_{α_k} might not have full column rank, hence $A_{\alpha_k}^\top A_{\alpha_k}$ will be singular. Indeed when λ is very small or when x^k is far from the solution, the index set α_k defined in (2.28a) might have cardinality larger than m (which is the only case where a singularity may arise given our construction of A), the number of rows of A. The next two subsections are devoted to proposing strategies to overcome these two issues.

Overall, the algorithm can be seen as an active set strategy where large changes on the active set are allowed in every iteration (instead of only one index) leading to faster convergence.

Globalization strategy. To enforce global convergence of Newton-type methods for solving nonlinear systems of equations it is customary to use a merit function based on which a step τ_k is selected which guarantees that

$$x^{k+1} = x^k + \tau_k d^k \tag{2.33}$$

decreases the merit function sufficiently. If R_γ was smooth, a first candidate for a merit function would be $\frac{1}{2}\|R_\gamma(x)\|^2$. However, non-smoothness of R_γ since in general this function is not differentiable and a stepsize τ_k that ensures $\|R_\gamma(x^{k+1})\|^2 < \|R_\gamma(x^k)\|^2$ might not exist.

Recently the following merit function, namely the *Forward-Backward Envelope* (FBE), was proposed for problems of the form (2.18)

$$\varphi_\gamma(x) = \inf_{z \in \mathbb{R}^n} \{f(x) + \nabla f(x)^\top (z - x) + g(z) + \tfrac{1}{2\gamma}\|z - x\|_2^2\}. \tag{2.34}$$

It is easy to see that function inside the infimum is strongly convex with respect to z and the infimum is uniquely achieved by the forward-backward step $T_\gamma(x) = \text{prox}_{\gamma g}(x - \gamma \nabla f(x))$. Therefore, in order to evaluate φ_γ at a point x one simply needs to be be able to perform the same operations required by FISTA. Furthermore, function φ_γ is continuously differentiable with gradient given by

$$\nabla \varphi_\gamma(x) = \gamma^{-1}(I - \gamma \nabla^2 f(x)) R_\gamma(x). \tag{2.35}$$

If $\gamma < 1/\|A\|^2$, then solutions of (2.18) are exactly the stationary points, *i.e.* the points for which $\nabla\varphi_\gamma(x)$ becomes zero. In fact one can additionally show that minimizing the FBE (which is an unconstrained smooth optimization problem) is entirely equivalent to solving (2.18), in the sense that $\inf\ \varphi = \inf\varphi_\gamma$ and $\arg\min\ \varphi = \arg\min\ \varphi_\gamma$. In the case of LASSO where f is quadratic the FBE is convex.

It is not hard to check that, provided it is well defined, the semismooth Newton direction d^k given by (2.29) is in fact a direction of descent for the FBE, *i.e.*, $\nabla\varphi_\gamma(x^k)^\top d^k < 0$. Therefore one can perform a standard backtracking line-search to find a suitable step that guarantees the Armijo condition and hence global convergence: Pick the first nonnegative integer i_k such that $\tau_k = 2^{-i_k}$ satisfies

$$\varphi_\gamma(x_k + \tau_k d_k) \leq \varphi_\gamma(x_k) + \zeta\tau_k\nabla\varphi_\gamma(x_k)^\top d_k. \tag{2.36}$$

where $\zeta \in (0, 1/2)$, and then set $x_{k+1} = x_k + \tau_k d_k$. Furthermore, this step choice guarantees that as soon as x^k is close enough to the solution, $\tau_k = 1$ will always satisfy (2.36) and the iterates will be given by the (pure) semismooth Newton method (2.32) inheriting all its convergence properties.

Continuation strategy. The FBE, through a simple line-search strategy, guarantees global convergence of the iterates generated by (2.33), where τ_k is given by (2.36) and d^k by (2.29), provided that A_{α_k} have full column rank. Notice that $m \ll n$, *i.e.*, A is a wide matrix. However, since the goal of solving the LASSO problem is to recover the sparsest solution of $Ax = y$, we know that such a solution will have $\|x^\star\|_0$ much smaller than m, hence as soon as x^k is close to the solution, A_{α_k} in practice will most likely have full column rank according to our experience. In fact, iterates for which α_k contains more than m indices are of no interest to us since we know for sure that at the solution the cardinality of $\alpha(x^\star) = \{i \mid |x_i^\star - \gamma\nabla f(x^\star)| > \gamma\lambda\} = \{i \mid x_i^\star \neq 0\}$ will be (much) smaller than m.

In order to guarantee that α_k contains few elements, a simple continuation strategy that gradually reduces λ_k to the target value λ is employed. Following [28], we start with $\lambda_0 = \max\{\lambda, \|A^\top\nabla f(x_0)\|_\infty\}$ and we decrease $\lambda_k = \max\{\lambda, \eta\lambda_{k-1}\}$, for some $\eta \in (0, 1)$ whenever $\|R_\gamma(x_k)\| \leq \lambda_k\epsilon_k$ with $\epsilon_k \to 0$. Not only does this ensure that (2.29) is well defined, but it allows to solve linear systems of small dimension to determine the Newton direction. A conceptual pseudo-algorithm summarizing the basic steps of the proposed algorithm is shown hereafter.

Algorithm 1 Forward-Backward Newton with continuation

Require: A, y, $x^0 \in \mathbb{R}^n$ (initial guess), $\gamma \in (0, 1/\|A\|^2)$, $\lambda > 0$, $\eta \in (0, 1)$, ϵ (tolerance)
Ensure: $x^\star$
  $\bar\lambda \leftarrow \max\{\lambda, \|A^\top\nabla f(x_0)\|_\infty\}$, $\bar\epsilon \leftarrow \epsilon$
  while $\bar\lambda > \lambda$ or $\|R_\gamma(x^k; \bar\lambda)\| > \epsilon$ **do**
    $x^{k+1} = x^k + \tau_k d^k$,
    where d^k solves (2.31) and τ_k satisfies (2.36)
    if $\|R_\gamma(x^k; \bar\lambda)\| \leq \lambda\bar\epsilon$ **then**
      $\bar\lambda \leftarrow \max\{\lambda, \eta\bar\lambda\}$, $\bar\epsilon \leftarrow \eta\bar\epsilon$
    end if
  end while

In Algorithm 1 we denote by $R_\gamma(x;\bar{\lambda})$ the fixed point residual introduced in (2.25) replacing g by $\bar{g}(x) = \bar{\lambda}\|x\|_1$.

Accelerating the Cholesky factorization. Assume we know the Cholesky factorisation of $A^\top A = L^\top L$ and we remove a column from matrix A. Let us write A as follows

$$A = \begin{bmatrix} A_{1:\kappa} & c & A_{\kappa+1:n} \end{bmatrix} \tag{2.37}$$

Then

$$\begin{aligned} A^\top A &= \begin{bmatrix} A_{1:\kappa} & c & A_{\kappa+1:n} \end{bmatrix}^\top \begin{bmatrix} A_{1:\kappa} & c & A_{\kappa+1:n} \end{bmatrix} && \text{(2.38a)} \\ &= \begin{bmatrix} A_{1:\kappa}^\top A_{1:\kappa} & A_{1:\kappa}^\top c & A_{1:\kappa}^\top A_{\kappa+1:n} \\ c^\top A_{1:\kappa} & c^\top c & c^\top A_{\kappa+1:n} \\ A_{\kappa+1:n}^\top A_{1:\kappa} & A_{1:\kappa} c & A_{\kappa+1:n}^\top A_{\kappa+1:n} \end{bmatrix} && \text{(2.38b)} \\ &= LL^\top && \text{(2.38c)} \\ &= \begin{bmatrix} L_{11} & & \\ \ell_{12} & \ell_{22} & \\ L_{31} & \ell_{32} & L_{33} \end{bmatrix} \begin{bmatrix} L_{11} & & \\ \ell_{12} & \ell_{22} & \\ L_{31} & \ell_{32} & L_{33} \end{bmatrix}^\top && \text{(2.38d)} \end{aligned}$$

Once we delete the column c at $\kappa+1$ we have

$$\begin{bmatrix} A_{1:\kappa}^\top A_{1:\kappa} & A_{1:\kappa}^\top A_{\kappa+1:n} \\ A_{\kappa+1:n}^\top A_{1:\kappa} & A_{\kappa+1:n}^\top A_{\kappa+1:n} \end{bmatrix} = \begin{bmatrix} \bar{L}_{11} & \\ \bar{L}_{31} & \bar{L}_{33} \end{bmatrix} \begin{bmatrix} \bar{L}_{11} & \\ \bar{L}_{31} & \bar{L}_{33} \end{bmatrix}^\top \tag{2.39}$$

from which we have that

$$\bar{L}_{11} = L_{11} \tag{2.40a}$$

$$\bar{L}_{31} = L_{31} \tag{2.40b}$$

and

$$\bar{L}_{33}\bar{L}_{33}^\top = L_{33}L_{33}^\top + \ell_{32}\ell_{32}^\top. \tag{2.40c}$$

The last equation is a rank-1 update (see Golub and Van Loan) and it comes at the cost of $2(n-\kappa)^2 + 4(n-\kappa)$.

When Cholesky factorizations are involved in the algorithm we described above, instead of performing these factorizations at every iteration of the algorithm, we may save a considerable amoount of operations by simply updating them. From one iteration to the other, usually few indices enter and/or leave the set α (the set of active indices described above), so, α_k and α_{k+1} are not expected to be much different. It is then computationally cheap to update the Cholesky factorization of $A_{\alpha_{k+1}}^\top A_{\alpha_{k+1}}$ if we know the Cholesky factorization of $A_{\alpha_k}^\top A_{\alpha_{k+1}}$. It may yet happen that an *ab initio* Cholesky factorization is computationally cheaper than an update in which α_k and α_{k+1} are a lot different (i.e., either $\alpha_{k+1} \setminus \alpha_k$ and/or $\alpha_k \setminus \alpha_{k+1}$ is a set of high cardinality).

How may insertions and or removals can the update afford so that it remains computationally cheaper from a factorization instead of an update? This question can be answered easily if we can predict the exact number of flops of the update; we can then choose between a Cholesky update and a Cholesky factorization. Here we show how this can be done.

Having computed a Cholesky factorisation with permutation for the matrix $A^\top A$, that is

$$A^\top A = PLL^\top P^\top, \tag{2.41}$$

we need to compute the Cholesky factorisation of $\bar{A}^\top \bar{A}$ where $\bar{A} = [\, A \; c \,]$. We have

$$\bar{A}^\top \bar{A} = \begin{bmatrix} A & c \end{bmatrix}^\top \begin{bmatrix} A & c \end{bmatrix} \tag{2.42a}$$

$$= \begin{bmatrix} A^\top A & A^\top c \\ c^\top A & c^\top c \end{bmatrix} \tag{2.42b}$$

$$= \begin{bmatrix} P & \\ & 1 \end{bmatrix} \begin{bmatrix} L_1 & \\ \ell_1^\top & \ell_2 \end{bmatrix} \begin{bmatrix} L_1 & \\ \ell_1^\top & \ell_2 \end{bmatrix}^\top \begin{bmatrix} P & \\ & 1 \end{bmatrix}^\top \tag{2.42c}$$

From which we have that

$$A^\top A = PL_1L_1^\top P^\top \Rightarrow L_1 = L, \tag{2.43a}$$

and ℓ_1 is computed from the following nice linear system

$$PL\ell_1 = A^\top c \Leftrightarrow L\ell_1 = P^\top A^\top c, \tag{2.43b}$$

and, provided that $c^\top c - \ell_1^\top \ell_1 > 0$, $\bar{A}^\top \bar{A}$ is positive definite and

$$\ell_2^2 = c^\top c - \ell_1^\top \ell_1. \tag{2.43c}$$

Note. to solve $A^\top Ax = b$ we do

$$PLL^\top P^\top x = b \tag{2.44}$$

and we set $x = Py$ so we then need to solve

$$PLL^\top y = b \Leftrightarrow LL^\top y = P^\top b. \tag{2.45}$$

The permuted Cholesky factor of $\bar{A}^\top \bar{A} = \bar{P}\bar{L}\bar{L}^\top \bar{P}^\top$ is

$$\bar{L} = \begin{bmatrix} L_1 & \\ \ell_1^\top & \ell_2 \end{bmatrix}, \tag{2.46}$$

and the corresponding permutation matrix is

$$\bar{P} = \begin{bmatrix} P & \\ & 1 \end{bmatrix}. \tag{2.47}$$

The computational cost of this update is $n^2 + 3n$.

Let us now insert a column in A, so we shall define the matrix

$$\tilde{A} = \begin{bmatrix} A_{1:\kappa} & c & A_{\kappa+1:n} \end{bmatrix}, \tag{2.48}$$

where $\kappa \in \mathbb{N}_{[1,n]}$. There is then a permutation matrix $\tilde{P}$ so that $\tilde{A}\tilde{P}^\top = [\, A \; c \,]$. It is then easy to update the factorisation

$$(\tilde{A}\tilde{P})^\top(\tilde{A}\tilde{P}) = \tilde{L}\tilde{L}^\top, \tag{2.49}$$

$$\Leftrightarrow \tilde{A}^\top\tilde{A} = \tilde{P}\tilde{L}\tilde{L}^\top\tilde{P}^\top \tag{2.50}$$

which is the updated Cholesky factorisation of $\tilde{A}^\top\tilde{A}$ with permutation matrix $\tilde{P}$.

Now we are going to insert many columns, recursively, at various positions in $\tilde{A}$. Essentially, we are going to update a permuted Cholesky factorisation by inserting a column in any position in A, so the updated matrix becomes

$$\bar{A} = \begin{bmatrix} A_{1:\kappa} & c & A_{\kappa+1:n} \end{bmatrix} \tag{2.51}$$

There is a permutation matrix $\tilde{P}$ so that

$$\bar{A}\tilde{P} = \begin{bmatrix} A & c \end{bmatrix} \triangleq \tilde{A} \tag{2.52}$$

Say we have $A^\top A = PLL^\top P^\top$. We can then compute a permuted Cholesky factorisation for $\tilde{A}^\top\tilde{A}$,

$$\tilde{A}^\top\tilde{A} = \bar{P}\tilde{L}\tilde{L}^\top\bar{P}^\top \tag{2.53a}$$

$$\Leftrightarrow \tilde{P}^\top\bar{A}^\top\bar{A}\tilde{P} = \bar{P}\tilde{L}\tilde{L}^\top\bar{P}^\top \tag{2.53b}$$

$$\Leftrightarrow \bar{A}^\top\bar{A} = \tilde{P}\bar{P}\tilde{L}\tilde{L}^\top\bar{P}^\top\tilde{P}^\top, \tag{2.53c}$$

which is the permuted Cholesky factorisation of $\bar{A}^\top\bar{A}$ with permutation matrix $\tilde{P}\bar{P}$.

Let now $A \in \mathbb{R}^{m\times n}$ be a given matrix and α be a collection of column indexes of A, and $A_\alpha \in \mathbb{R}^{m\times|\alpha|}$ and we know the Cholesky factorisation of $A_\alpha^\top A_\alpha = L_\alpha L_\alpha^\top$. Let now $\bar{\alpha} = \alpha \cup \{\alpha^*\}$, i.e., the α^*-th column of A is added to A_α to form the new matrix

$$A_{\bar{\alpha}} = \begin{bmatrix} A_\alpha & A_{\alpha^*} \end{bmatrix}. \tag{2.54}$$

Having augmented A_α we can now update the factorisation of $A_\alpha^\top A_\alpha$ and compute an $L_{\bar{\alpha}}$ such that

$$A_{\bar{\alpha}}^\top A_{\bar{\alpha}} = L_{\bar{\alpha}} L_{\bar{\alpha}}^\top. \tag{2.55}$$

Efficient line search. Define $f(x) = \frac{1}{2}\|Ax - y\|^2$ and $g(x) = \lambda\|x\|_1$. The forward-backward envelope of $F(x) = f(x) + g(x)$ is given by

$$F_\gamma(x) = f(x) - \frac{\gamma}{2}\|\nabla f(x)\|^2 + g^\gamma(x - \gamma\nabla f(x)), \tag{2.56}$$

where g^γ is the Moreau envelope of $g(x) = \lambda\|x\|_1$ given by

$$g^\gamma(x) = \sum_{i=1}^{n} h_\gamma(x_i), \tag{2.57}$$

where $h_{\gamma\lambda}$ is the Huber function with paramter $\gamma\lambda$

$$h_{\gamma\lambda}(t) = \begin{cases} \frac{t^2}{2\gamma}, & \text{if } |t| \leq \gamma\lambda \\ |t| - \frac{\gamma\lambda}{2}, & \text{otherwise} \end{cases} \tag{2.58}$$

Notice that

$$f(x + \tau d) - f(x) = \frac{\tau^2}{2}\|Ad\|^2 + \tau\langle Ax - y, Ad\rangle \tag{2.59}$$

Furthermore, we have $\|\nabla f(x)\|^2 = \|A^\top(Ax - y)\|^2$ and

$$\|\nabla f(x + \tau d)\|^2 - \|\nabla f(x)\|^2 = \tau^2\|A^\top Ad\|^2 + 2\tau\langle A^\top(Ax - y), A^\top Ad\rangle \tag{2.60}$$

Now for given x and d define the following quantities

$$\xi = Ad \tag{2.61a}$$
$$\psi = A^\top\xi \tag{2.61b}$$
$$r = Ax - y \tag{2.61c}$$
$$\pi = A^\top r. \tag{2.61d}$$

In the Newton algorithm we update x as follows

$$x \leftarrow x + \tau d, \tag{2.62}$$

and r and π are updated as follows so that we don't need to recompute them and we don't have to perform any matrix-vector products

$$r \leftarrow r + \tau\xi \tag{2.63a}$$
$$\pi \leftarrow \pi + \tau\psi \tag{2.63b}$$

We then have

$$f(x + \tau d) - f(x) = \frac{\tau^2}{2}\langle\xi, \xi\rangle + \tau\langle r, \xi\rangle \tag{2.64}$$

and

$$\|\nabla f(x+\tau d)\|^2 - \|\nabla f(x)\|^2 = \tau^2\langle\psi,\psi\rangle + 2\tau\langle\pi,\psi\rangle \tag{2.65}$$

Now, overall we have

$$\begin{aligned} F_\gamma(x+\tau d) - F_\gamma(x) &= \frac{\tau^2}{2}\langle\xi,\xi\rangle + \tau\langle r,\xi\rangle - \frac{\gamma}{2}\left(\tau^2\langle\psi,\psi\rangle + 2\tau\langle\pi,\psi\rangle\right) \\ &\quad + g^\gamma(x+\tau d - \gamma\nabla f(x+\tau d)) - g^\gamma(x-\gamma\nabla f(x)) \\ &= \tau\left(\langle r,\xi\rangle - \gamma\langle\pi,\psi\rangle\right) + \tau^2\cdot\left(\langle\xi,\xi\rangle - \gamma\langle\psi,\psi\rangle\right)/2 \\ &\quad + g^\gamma((x-\gamma\pi) + \tau(d-\gamma\xi)) - g^\gamma(x-\gamma\pi) \end{aligned} \tag{2.66}$$

Now let

$$u = x - \gamma\pi \tag{2.67}$$

$$v = d - \gamma\psi \tag{2.68}$$

Then

$$F_\gamma(x+\tau d) - F_\gamma(x) = \beta_0 + \beta_1\tau + \beta_2\tau^2 + g^\gamma(u+\tau v), \tag{2.69}$$

where

$$\beta_0 = -g^\gamma(u), \tag{2.70a}$$

$$\beta_1 = \langle r,\xi\rangle - \gamma\langle\pi,\psi\rangle, \tag{2.70b}$$

$$\beta_2 = \left(\langle\xi,\xi\rangle - \gamma\langle\psi,\psi\rangle\right)/2, \tag{2.70c}$$

and u in the end is updated as

$$u \leftarrow u + \tau v. \tag{2.71}$$

Let us define $R_\gamma(x) = x - \mathrm{prox}_{\gamma g}(x - \gamma\nabla f(x))$. This operator is useful for the computation of the gradient of the forward barckward envelope according to

$$\nabla F_\gamma(x) = \gamma^{-1}(I_n - \gamma A^\top A)R_\gamma(x) \tag{2.72}$$

Vector $R_\gamma(x)$ is evaluated as follows

$$(R_\gamma(x))_\alpha = \gamma(\pi_\alpha + \lambda\,\mathrm{sign}(x_\alpha)) \tag{2.73a}$$

$$(R_\gamma(x))_{\alpha_c} = x_{\alpha_c} \tag{2.73b}$$

Simulations. In this section we apply the proposed methodology to various data streams and we compare it to standard algorithms such as FISTA, ADMM [29] and the interior point method of Kim *et al.* [14], also known as the L1LS method. In our approach we used the continuation

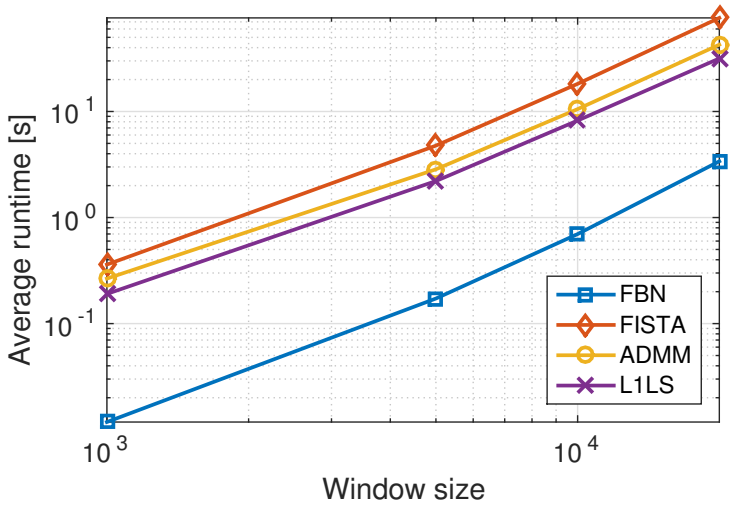

Figure 2.4: Average runtimes varying the window size. FBN: proposed forward-backward Newton method, L1LS [14], ADMM [18], FISTA [16] for sparsity equal to 0.1.

strategy described above with $\eta = 0.5$ and an Armijo line search. The required tolerance for the termination of all algorithms was set to 10^{-8}.

We observed that after decompressing the first window, the number of iterations required for convergence was remarkably low (in most cases, around 4 iterations for each window were sufficient). It should be highlighted that after the first decompression, the computational cost of the algorithm decreases significantly. This is first because of the aforementioned warm-start and second because the value of the residual $r = Ax - y$ is updated by using only vector-vector operations. Updating $A^{(i+1)} \leftarrow A^{(i)}P$ and $x_0^{(i+1)} \leftarrow P^\top \hat{x}^{(i)}$ and using the fact that P is orthogonal, we have that $r^{(i+1)} = A^{(i+1)}x_0^{(i+1)} - y^{(i+1)} = r^{(i)} + y^{(i)} - y^{(i+1)}$.

A stream of total length $N = 10^6$ was generated as follows: its entries are drawn from $\mathrm{Ber}(S)$, where S is average stream sparsity. Then, the non-zero entries are taken uniform in $[-2, -1] \cup [1, 2]$ and multiplied by $8\sigma\sqrt{2 \log N}$ (dynamic range assumption [13]), based on selected noise variance σ^2. We then select window size n, let $s = nS, m = 4s$, and generate sampling matrix A with i.i.d. entries $\mathcal{N}(0, 1/m)$. For LASSO, we pick $\lambda = 8\sigma\sqrt{2 \log N}$ following [13].

In Figures 2.4 and 2.5 we observe that the proposed algorithm outperforms all state-of-the-art methods by an order of magnitude.

The results presented in Figure 1 were obtained for a fixed sparsity 10% and $w^{(i)}$ being a zero-mean normally distributed noise with variance 0.01. In Figure 2, for the same noise level and using a window of size $n = 5000$ we show how the runtime is affected by the sparsity of the data stream.

The remarkable speed-up of the proposed algorithm is a result of two factors: first it is the local quadratic convergence rate of the Newton method in combination with the warm start. Second, several terms such as the residual (i.e., the term $Ax - y$) and certain Cholesky factorizations can be passed to the next instance of FBN; this saves a considerable number of FLOPS at every iteration.

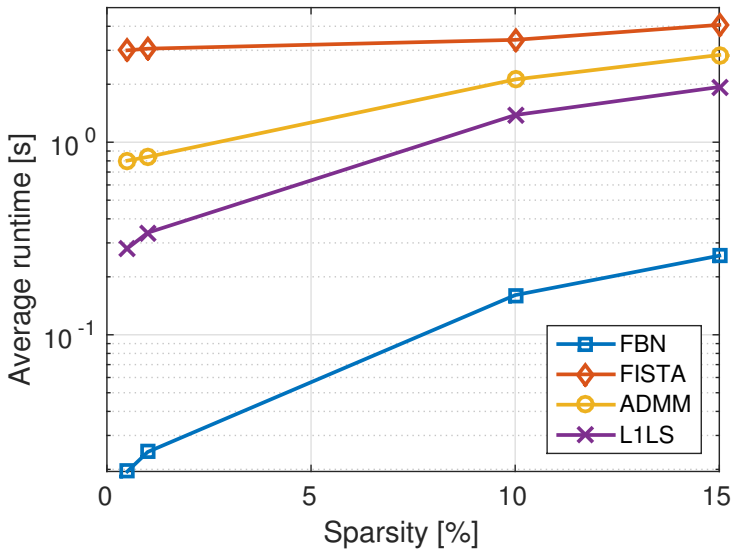

Figure 2.5: Average runtimes varying the sparsity of the data stream for a window size of 8000.

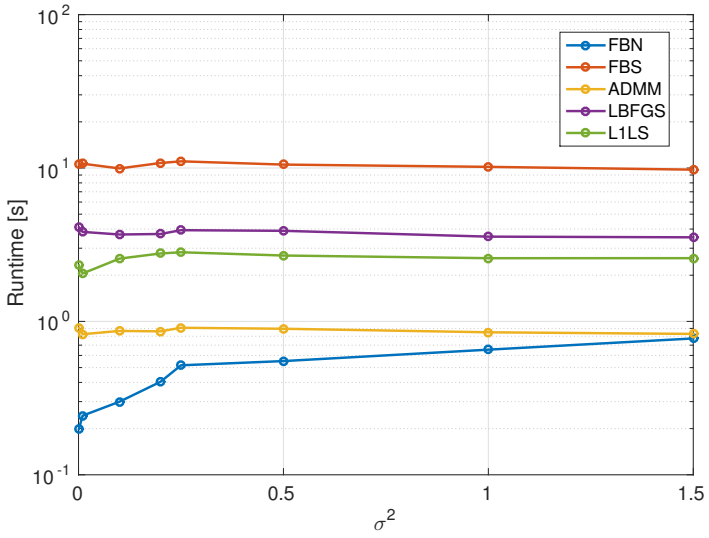

Figure 2.6: Average runtimes varying σ^2 (i.e., the signal-to-noise ratio) using a constant window size of 8000 and a constant sparsity 0.05. Reported runtimes are averaged over 50 measurements.

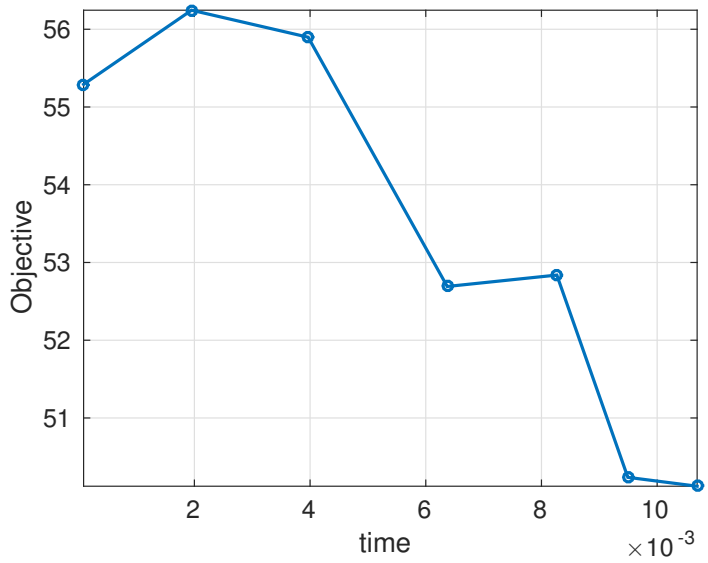

Figure 2.7: Convergence of the primal objective to its optimal value with respect to runtime for a random problem with $m = 600$, $n = 5000$ a sparse matrix A with density 1%. The solution was determined in $10.6ms$ and the FP residual was $1.2265 \cdot 10^{-14}$.

Conclusions We have proposed an efficient method for successively decompressing the entries of a data stream sampled using Recursive Compressed Sensing [11]. We have proposed a second-order proximal method for solving LASSO with accelerated convergence over state-of-art methods. Our scheme is very efficient as each iteration nails down to solving a linear system of low dimension, which we may further avoid by a single Cholesky factorization at a pre-processing step. We have tested our algorithm against the state-of-art for various windows sizes and sparsity patterns; our experiments depict notable speed-up which may renders RCS suitable for an online implementation under stringent time constraints. Finally, it should be noted that all reported results are performed in MATLAB, yet the proposed method exhibits a ten-fold speed-up compared to all existing approaches. We believe that an optimized C or C++ implementation is likely to accelerate to lead to an even higher speed-up.

2.1.2 Online identification using streaming data

Online identification on a stream of data can be performed along the lines of the methodology we proposed for compressed sensing since, in some sense, LASSO-based identification is a problem dual or converse to that of compressed sensing. Unlike compressed sensing, in identification the objective is to derive a simplified representation of a multitude of data where the measurements outnumber the parameters of the model. A model, therefore, can be seen as a simplified and coarse representation of a set of data.

In the case of identification of linear models, the output $\hat{y}_k$ of the model at time k is represented by the *auto-regressive with exogenous input* form:

$$\hat{y}_k = \sum_{i=1}^{N_y} \alpha_i y_{k-i} + \sum_{i=1}^{N_u} \beta_i u_{k-i}, \tag{2.74}$$

where N_y and N_u are the output and input history lengts — all past values from y_{k-N_y} to y_{k-1} and from u_{k-N_u} up to u_{k-1} are used to predict the output at time k. Given an N-length sequence of observed input/output pairs $\{u_k, y_k\}_{k=1}^{N}$, the considered identification problem aims at estimating the model parameters α_i and β_i describing the relationship between the observed input/output sequence.

The input and output lags N_u and N_y have to be a-priori specified by the user, and this represents a critical step in the estimation of the ARX model (2.74) from data. In fact, in order to adequately capture the underlying dynamic dependence of the current output y_k on the past inputs and outputs, large values for N_u and N_y are typically chosen. On the one hand, this increases the flexibility of the model, thus avoiding a structural model bias. At the same time, this leads to over-parameterized models, which tend to overfit the training data, leading to poor generalization on unseen data. This is a well-known concept in machine learning and system identification [30], commonly referred as bias-variance tradeoff.

Thus, in order to select the ARX model structure (i.e., the model parameters α_i and β_i in (2.74) which are actually needed for an accurate approximation of the underlying relationship among the variables), a LASSO-based identification problem can be formulated, where the fitting error between the observed output y_k and the model output $\hat{y}_k$ is minimized, along with a regularization term penalizing the ℓ_1-norm of the model parameters. Such a penalty is used to minimize the number of nonzero model parameters, thus selecting the model structure directly from the data. It worth remarking that, similarly to the compressed sensing case, once the model structure is selected solving a LASSO problem, the non-zero model parameters α_i and β_i should have re-estimated solving a least-squares problem, without any regularization.

In the following paragraph, we provide technical details on model structure detection (through LASSO) using a buffer of data (i.e., a moving window). This relies on the principle that old data become outdated, therefore, the system dynamics should only be determined by recent data. The window length is a tuning knob of the proposed methodology, and it should be chosen to balance the tradeoff between the speed of adaptation to parameter variation and insensitivity to noise. In fact, a large window length uses the long memory of data and it provides a robust but slowly convergent estimate, while a smaller window raises the level of adaptation but also increases the sensitivity to noise. We will also explain how the window length can be extended dynamically and during runtime. Again, we will follow an approach based on a stream of data in contrast to the batch approach where modelling takes place only offline.

Model order selection. First, let us introduce the regressor vector:

$$a_k = \begin{bmatrix} y_{k-1} & \cdots & y_{k-N_y} & u_{k-1} & \cdots & u_{k-N_u} \end{bmatrix}^\top \in \mathbb{R}^{n_a}, \tag{2.75}$$

with $n_a = N_y + N_u$. By stacking the model parameters α_i and β_i in a vector $\theta \in \mathbb{R}^{n_a}$:

$$\theta = \begin{bmatrix} \alpha_1 & \cdots & \alpha_{N_y} & \beta_1 & \cdots & \beta_{N_u} \end{bmatrix}^\top, \tag{2.76}$$

the ARX model in (2.74) can be written in the compact form:

$$\hat{y}_k = a_k^\top \theta. \tag{2.77}$$

To detect the structure (i.e., non-zero parameters) of the ARX model (2.74), we use input/output pairs $\{u_k, y_k\}$ from time $k - \max\{N_y, N_u\}$ to time $k + H$, where H represents the length of the rolling window. Based on these data, we can construct the regressor matrix:

$$A = \begin{bmatrix} y_{k-1} & y_{k-2} & \cdots & y_{k-N_y} & u_{k-1} & u_{k-2} & \cdots & u_{k-N_u} \\ y_k & y_{k-1} & \cdots & y_{k-N_y+1} & u_k & u_{k-1} & \cdots & u_{k-N_u+1} \\ \vdots & \cdots & & \vdots & \vdots & \vdots & & \vdots \\ y_{k+H-1} & y_{k+H-2} & \cdots & y_{k+H-N_y} & u_{k+H-1} & u_{k+H-2} & \cdots & u_{k+H-N_u} \end{bmatrix} = \begin{bmatrix} a_k^\top \\ a_{k+1}^\top \\ \vdots \\ a_{k+H}^\top \end{bmatrix}, \tag{2.78}$$

and stack the observed outputs in the vector

$$b = \begin{bmatrix} y_k & y_k & \cdots & y_{k+H} \end{bmatrix}^\top. \tag{2.79}$$

It is always recommended that $H > N_y + N_u = n_a$ (at least). This said, matrix A will be always a tall matrix. Then, the LASSO problem for support detection is formulated as follows

$$\text{minimize}_\theta \, \frac{1}{2}\|A\theta - b\|^2 + \lambda\|\theta\|_1. \tag{2.80}$$

The term $\|A\theta - b\|^2$ aims at minimizing the norm of the prediction error $\hat{y}_k - y_k$, while $\lambda\|\theta\|_1$ tends to enforce sparsity in the parameter vector θ. The hyper-parameter $\lambda > 0$ thus control the tradeoff between data fitting and model complexity, and it can be tuned by the user through cross-validation.

Once problem (2.80) is solved, the model structure of the ARX model (2.74) is then selected by getting rid of all parameters θ_i such that $|\theta_i^\star| < \epsilon$, where $\theta^\star$ is the optimizer of problem (2.80) and ϵ is a (small) threshold. Having detected the model structure, the non-zero model parameters of the vector $\theta^\star$ are then re-estimated by solving the least-square problem

$$\text{minimize}_{\bar{\theta}} \, \frac{1}{2}\|\bar{A}\bar{\theta} - b\|^2, \tag{2.81}$$

where $\bar{A}$ and $\bar{\theta}$ are the reduced-dimension variants of A and θ using only the indices corresponding to the non-zero parameters in $\theta^\star$.

Using a forgetting factor. Instead of using a rolling window, it often makes sense to perform the training using a *forgetting factor* — more recent data are to be taken into account with higher confidence than older ones as the system dynamics may have changed. In that case,

the LASSO problem we formulate is

$$\text{minimize } \frac{1}{2}\|\tilde{e}(\theta)\|_2^2+\lambda\|\theta\|_1, \tag{2.82}$$

where now instead of $e(\theta) = A\theta - b$ we have $\tilde{e}(\theta)$ being a vector whose j-th entry is

$$\tilde{e}_j = \kappa^j \left(y_{k+j} - \sum_{i=1}^{N_y} \alpha_i y_{k+j-i} - \sum_{i=1}^{N_u} \beta_i u_{k+j-i} \right), \tag{2.83}$$

with $\kappa \in (0, 1)$ being the forgetting factor. Alternatively, we may use the fact that every symmetric positive definite matrix Q defines a norm as follows $\|x\|_Q^2 = x'Qx$, to recast the above problem as

$$\text{minimize } \frac{1}{2}\|A\theta - b\|_Q^2+\lambda\|\theta\|_1, \tag{2.84}$$

where Q is the matrix

$$Q = \begin{bmatrix} 1 & & & & \\ & \kappa & & & \\ & & \kappa^2 & & \\ & & & \ddots & \\ & & & & \kappa^H \end{bmatrix}. \tag{2.85}$$

2.1.2.1 Implementation as a toolbox

The forward-backward Newton envelope algorithm which we proposed is available as an open-source toolbox and it is available online at github.com/alphaville/DISIRE-data-driven (In particular, see the file `lasso/FBNewtonL1LS.m` which exports its functionality via the simple interface `result = FBNewtonL1LS(A, b, lambda0, opt)`). The code is well-structured and thoroughly documented and tested with unit tests.

2.1.3 PWA regression

2.1.3.1 Identification of PWARX models

An extension of the ARX models in (2.77) is given by *PieceWise Affine autoRegressive with eXogenous inputs* (PWARX) models, which provide simple yet flexible structures for describing the behavior of nonlinear and time-varying systems.

The output $\hat{y}_k$ of a PWARX model is given by a *PieceWise Affine* (PWA) map f of the current regressor a_k (defined in eq. (2.75)), i.e.,

$$\hat{y}_k = f(a_k) = \begin{cases} \begin{bmatrix} 1 & a_k^\top \end{bmatrix}^\top \theta^{(1)} & \text{if } a_k \in \mathcal{A}_1, \\ \vdots & \\ \begin{bmatrix} 1 & a_k^\top \end{bmatrix}^\top \theta^{(s)} & \text{if } a_k \in \mathcal{A}_s, \end{cases} \tag{2.86}$$

where $s \in \mathbb{N}$ denotes the number of (affine) submodels defining the PWA function f, and $\theta^{(i)} \in \mathbb{R}^{n_a+1}$ are real parameters. The regressor a_k is assumed to belong to a set $\mathcal{A} \subseteq \mathbb{R}^{n_a}$, denoted as regressor space. The sets $\mathcal{A}_i$ $(i = 1, \dots, s)$ in (2.86) are polyhedra that form a complete polyhedral partition[2] of the regressor space $\mathcal{A}$. Each polyhedron $\mathcal{A}_i$ is defined by the linear inequalities

$$\mathcal{A}_i \doteq \{a \in \mathbb{R}^{n_a} : \mathcal{H}_i a \leq \mathcal{D}_i\}, \tag{2.87}$$

with $\mathcal{H}_i$ and $\mathcal{D}_i$ being real-valued matrices.

Problem statement. Given a set of input/output observations $\{u_k, y_k\}_{k=1}^N$ generated by a dynamical system, PWA regression aims at finding a PWA function f as in (2.86) describing relationship between the observed input/output sequence. Note that estimating a PWARX model f from data requires (i) choosing the number of submodels s, (ii) computing the parameter vectors $\{\theta^{(i)}\}_{i=1}^s$ that characterize the local affine models of the PWA map f, and (iii) finding the polyhedral partition $\{\mathcal{A}_i\}_{i=1}^s$ of the regressor space $\mathcal{A}$ where those local affine models are defined.

When choosing the value of s, one must take into account the tradeoff between data fitting and model complexity. For small values of s, the PWA map f cannot accurately capture the nonlinear and time-varying dynamics of the system. On the other hand, increasing the number of local affine submodels also increases the degrees of freedom in the description of the PWA map f, which may cause overfitting and poor generalization to unseen data (i.e, the final estimate is sensitive to the noise corrupting the observations), besides increasing the complexity of the estimation procedure and of the resulting PWA model. In the PWA regression algorithm described in the following, we assume that s is fixed by the user. The value of s can be chosen through cross-validation based procedures, with a possible upper-bound dictated by the maximum tolerable complexity of the estimated model.

Contribution. We device a novel two-stage approach for identification of PWA functions. The main idea of the developed algorithm is to first process the observed regressor/output pairs sequentially and assign each pair to the affine submodel that is most compatible to it. At the same time, while doing the assignment of a pair, the parameters of the corresponding submodel are updated via a recursive least-squares techniques. The second stage starts once all the observations have been classified: the regressor domain is partitioned into polyhedral regions through a piecewise linear separator, which is computed by a regularized piecewise-smooth Newton method, derived from the one proposed in [31]. The method is alternative to (and more efficient than) the classical robust linear programming approach of [32] and does not involve smooth approximations as in the algorithm proposed in [33]. In addition, an averaged stochastic gradient descent algorithm is proposed to solve the same problem recursively. This latter method is particularly appealing for on-line identification and adaptation of the PWA map, as it processes one data sample at the time. Overall, we show that the advantage of the pro-

[2] A collection $\{\mathcal{A}_i\}_{i=1}^s$ is a complete partition of the regressor domain $\mathcal{A}$ if $\bigcup_{i=1}^s \mathcal{A}_i = \mathcal{A}$ and $\mathring{A}_i \cap \mathring{A}_j = \emptyset$, $\forall i \neq j$, with $\mathring{A}_i$ denoting the interior of $\mathcal{A}_i$.

posed PWA regression algorithm over other existing methods available in the literature [34–41] is twofold: computationally very effective for offline learning, very suitable for online learning.

2.1.3.2 Identification algorithm

Technical details of the developed algorithm for PWA regression are provided in this paragraph. As already mentioned, the proposed approach consists of two stages:

S1. The simultaneous *clustering* of the regressor vectors $\{a_k\}_{k=1}^N$ and *estimation* of the parameter matrices $\theta^{(1)}, \dots, \theta^{(s)}$ describing the PWA map in Eq. (2.86). This is performed recursively, by processing the training pairs $\{a_k, y_k\}$ sequentially, and could be done either offline (on a batch of data), or online, while data are acquired in real-time;

S2. The *computation of a polyhedral partition* of the regressor space $\mathcal{A}$ through an efficient multi-class linear separation method. This stage is performed after stage S1, and it can be performed either offline (batch) via a Newton-like method, or online (recursively) via an averaged stochastic gradient descent algorithm.

Iterative clustering and parameter estimation. Stage S1 is carried out as described in Algorithm 2. The algorithm is an extension to the case of multiple linear regressions and clustering of the (computationally very efficient) approach proposed in [42] for solving recursive least squares problems using inverse QR decomposition.

Algorithm 2 requires an initial guess for the parameter matrices $\theta^{(i)}$ and cluster centroids c_i, $i = 1, \dots, s$. Zero matrices $\theta^{(i)}$, randomly chosen centroids c_i, and identity centroid covariance matrices R_i are a possible choice. In alternative, if Algorithm 2 can be executed in a batch fashion on a subset of data, one can initialize the parameter matrices $\theta^{(1)}, \dots, \theta^{(s)}$ all equal to the best linear model

$$\theta^{(i)} \equiv \arg\min_{\theta} \sum_{k=1}^{N} \|y_k - [1 \;\; a_k^\top]^\top \theta\|_2^2, \; \forall i = 1, \dots, s \tag{2.90}$$

that fits all data, and classify the regressors $\{a_k\}_{k=1}^N$ through k-means clustering, to compute the cluster centroids and the centroid covariance matrices. Otherwise, the initial guess for the parameters could be obtained executing Algorithm 2 without the first term in (2.88) and updating the cluster centroids and covariance matrices only once all the data have been classified. Clearly, the final estimate of the parameter matrices $\theta^{(1)}, \dots, \theta^{(s)}$ and of the clusters $\mathcal{C}_1, \dots, \mathcal{C}_s$ depends on their initial values. When working offline on a batch of identification data, estimation quality may be improved by repeating Algorithm 2 iteratively, by using its output as initial condition for its following execution.

After computing the estimation error $e_i(k)$ for all models i at Step 2.1, Step 2.2 picks up the "best" mode $i(k)$ to which the current sample a_k must be associated with, based on a tradeoff between reducing the prediction error $e_i(k)$ and penalizing the distance (weighted by matrix R_i^{-1}) between a_k and the corresponding centroid c_i.

Step 2.4 updates the parameter matrix $\theta^{(i(k))}$ associated to the selected mode $i(k)$ using the recursive least-squares algorithm of [42] based on inverse QR factorization. Note that only

Algorithm 2 Recursive clustering and parameter estimation algorithm

Input: Sequence of observations $\{a_k, y_k\}_{k=1}^N$, desired number s of modes, noise covariance matrix Λ_e; initial condition for matrices $\theta^{(i)}$, cluster centroids c_i, and centroid covariance matrices R_i, $i = 1, \ldots, s$.

1. **let** $\mathcal{C}_i \leftarrow \emptyset$, $i = 1, \ldots, s$;
2. **for** $k = 1, \ldots, N$ **do**
   2.1. **let** $e_i(k) \leftarrow y_k - \theta^{(i)} \left[\begin{smallmatrix} 1 \\ a_k \end{smallmatrix}\right]$, $i = 1, \ldots, s$;
   2.2. **let**

$$i(k) \leftarrow \arg \min_{i=1,\ldots,s} (a_k - c_i)^\top R_i^{-1} (a_k - c_i) + e_i(k)^\top \Lambda_e^{-1} e_i(k); \tag{2.88}$$

   2.3. **let** $\mathcal{C}_{i(k)} \leftarrow \mathcal{C}_{i(k)} \cup \{a_k\}$;
   2.4. **update** the parameter matrices $\theta^{(i(k))}$ using the inverse QR factorization approach of [42];
   2.5. **let**

$$\delta c_{i(k)} \leftarrow \frac{1}{|\mathcal{C}_{i(k)}|} (a_k - c_{i(k)});$$

   2.6. **update** the centroid $c_{i(k)}$ of cluster $\mathcal{C}_{i(k)}$

$$c_{i(k)} \leftarrow c_{i(k)} + \delta c_{i(k)}; \tag{2.89}$$

   2.7. **update** the centroid covariance matrix $R_{i(k)}$ for cluster $\mathcal{C}_{i(k)}$

$$R_{i(k)} \leftarrow \frac{|\mathcal{C}_{i(k)}| - 2}{|\mathcal{C}_{i(k)}| - 1} R_{i(k)} + \delta c_{i(k)} \delta c_{i(k)}^\top + \frac{1}{|\mathcal{C}_{i(k)}| - 1} \left[a_k - c_{i(k)}\right] \left[a_k - c_{i(k)}\right]^\top;$$

3. **end for**;
4. **end**.

Output: Estimated matrices $\theta^{(1)}, \ldots, \theta^{(s)}$, clusters $\mathcal{C}_1, \ldots, \mathcal{C}_s$.

the parameters of the matrix $\theta^{(i(k))}$ associated to the selected mode $i(k)$ are updated at time k, while the parameters associated to the other modes are not.

It is worth remarking that a possible choice for the noise covariance matrix Λ_e is $\Lambda_e = I$ (i.e., the regression errors are not weighted in (2.88)). Alternatively, if Algorithm 2 is executed in a batch fashion and iteratively repeated by using the output as initial condition for the next execution, an estimate $\hat{\Lambda}_e$ of Λ_e can be computed at the end of each execution as the sample covariance of the modeling error, i.e.,

$$\hat{\Lambda}_e = \frac{1}{N} \sum_{i=1}^{s} \sum_{\substack{k=1 \\ a_k \in \mathcal{C}_i}}^{N} \left(y_k - [1 \;\; a_k^\top]^\top \theta^{(i)} \right) \left(y_k - [1 \;\; a_k^\top]^\top \theta^{(i)} \right)^\top .$$

Partitioning the regressor space. When one is interested in getting also the partition $\{\mathcal{A}_i\}_{i=1}^s$, besides the affine models $\theta^{(i)}$ and centroids c_i identified by Algorithm 2, the clusters $\{\mathcal{C}_i\}_{i=1}^s$ must be separated via a multicategory discrimination algorithm. We propose here a variation of the discrimination technique of [32] to partition the regressor space that is much more efficient from a computational point of view, especially when dealing with a large number N of data points.

For $i = 1, \ldots, s$, let M_i be a $m_i \times n_a$ dimensional matrix (with m_i denoting the cardinality of cluster $\mathcal{C}_i$) obtained by stacking the regressors $a_k^\top$ belonging to $\mathcal{C}_i$ in its rows. The piecewise affine multicategory discrimination problem aims at computing a (convex) piecewise affine separator function $\phi : \mathbb{R}^{n_a} \to \mathbb{R}$ discriminating between the clusters $\mathcal{C}_1, \ldots, \mathcal{C}_s$. The piecewise affine separator ϕ is defined as the maximum of s affine functions $\{\phi_i(a)\}_{i=1}^s$, i.e.,

$$\phi(a) = \max_{i=1,\ldots,s} \phi_i(a). \tag{2.91}$$

The affine functions $\phi_i(a)$ are described by the parameters $\omega^i \in \mathbb{R}^{n_a}$ and $\gamma^i \in \mathbb{R}$, namely:

$$\phi_i(a) = \begin{bmatrix} a^\top & -1 \end{bmatrix} \begin{bmatrix} \omega^i \\ \gamma^i \end{bmatrix}, \tag{2.92}$$

and, in case of piecewise linearly separable clusters, they should satisfy the inequality constraints

$$[M_i \;\; -\mathbf{1}_{m_i}] \begin{bmatrix} \omega^i \\ \gamma^i \end{bmatrix} > [M_i \;\; -\mathbf{1}_{m_i}] \begin{bmatrix} \omega^j \\ \gamma^j \end{bmatrix}, \quad i, j = 1, \ldots, s, \; i \neq j, \tag{2.93}$$

or, equivalently,

$$[M_i \;\; -\mathbf{1}_{m_i}] \begin{bmatrix} \omega^i \\ \gamma^i \end{bmatrix} \geq [M_i \;\; -\mathbf{1}_{m_i}] \begin{bmatrix} \omega^j \\ \gamma^j \end{bmatrix} + \mathbf{1}_{m_i}, \quad i, j = 1, \ldots, s, \; i \neq j. \tag{2.94}$$

The piecewise-affine separator ϕ thus satisfies the conditions:

$$\begin{cases} \phi(a) = \begin{bmatrix} a^\top & -1 \end{bmatrix} \begin{bmatrix} \omega^i \\ \gamma^i \end{bmatrix}, \ \forall a \in \mathcal{C}_i, \ \ i = 1, \dots, s \\ \phi(a) \geq \begin{bmatrix} a^\top & -1 \end{bmatrix} \begin{bmatrix} \omega^j \\ \gamma^j \end{bmatrix} + 1, \ \forall a \in \mathcal{C}_i, \ i \neq j. \end{cases} \tag{2.95}$$

Rather than solving a linear program as in [32], we determine $\{\omega^i, \gamma^i\}_{i=1}^s$ by solving the convex unconstrained optimization problem

$$\begin{aligned} \min_{\{\omega^i,\gamma^i\}_{i=1}^s} \frac{\lambda}{2} \sum_{i=1}^{s} \left(\|\omega^i\|_2^2 + (\gamma^i)^2\right) + \\ \sum_{i=1}^{s} \sum_{\substack{j=1 \\ j \neq i}}^{s} \frac{1}{m_i} \left\| \left(\begin{bmatrix} M_i & -\mathbf{1}_{m_i} \end{bmatrix} \begin{bmatrix} \omega^j - \omega^i \\ \gamma^j - \gamma^i \end{bmatrix} + \mathbf{1}_{m_i} \right)_+ \right\|_2^2, \end{aligned} \tag{2.96}$$

where $\lambda \geq 0$ is an ℓ_2-regularization term (no regularization in case $\lambda = 0$).

Problem (2.96) generates a piecewise-affine function that minimizes the (averaged) squared 2-norm of the violation of the inequalities (2.94). The problem is solved by using a regularized piecewise-smooth Newton method with Armijo's line search similar to the one proposed in [31] for functions $g : \mathbb{R}^{n_\xi} \to \mathbb{R}$ of the form

$$g(\xi) = \frac{\lambda}{2} \|\xi\|_2^2 + \sum_{j=1}^{n_g} \|g_j(\xi)_+\|_2^2, \tag{2.97}$$

where $g_j : \mathbb{R}^{n_\xi} \to \mathbb{R}$ are convex and twice continuously differentiable functions. In particular, we exploit the linearity of functions g_i's. In fact, for the special case of solving Problem (2.96), the optimization vector is $\xi = [(\omega^1)^\top \ \dots \ (\omega^s)^\top \ \gamma^1 \ \dots \ \gamma^s]^\top \in \mathbb{R}^{n_\xi}$, $n_\xi = s(n_a + 1)$, and g_j's are affine functions:

$$g_j(\xi) = t_j^\top \xi + r_j, \ j = 1, \dots, n_g, \tag{2.98}$$

where $n_g = N(s-1)$ and $t_j \in \mathbb{R}^{n_\xi}$, $r_j \in \mathbb{R}$ are easily obtained from (2.96) as a function of matrices $\{M_i\}_{i=1}^s$ and coefficients $\{m_i\}_{i=1}^s$. By letting

$$\mathcal{T} = [t_1 \ \dots \ t_{n_g}]^\top, \ \mathcal{R} = [r_1 \ \dots r_{n_g}]^\top, \tag{2.99}$$

given a vector $\xi \in \mathbb{R}^{n_\xi}$, let

$$I(\xi) = \{i \in \{1, \dots, n_g\} : \ \mathcal{T}_i \xi - \mathcal{R}_i > 0\}. \tag{2.100a}$$

Then,

$$g(\xi) = \frac{\lambda}{2} \xi^\top \xi + \sum_{i \in I(\xi)} (\mathcal{T}_i \xi - \mathcal{R}_i)^2 \tag{2.100b}$$

$$\nabla g(\xi) = \lambda \xi + \mathcal{T}_{I(\xi)}^\top (\mathcal{T}_{I(\xi)} \xi - \mathcal{R}_{I(\xi)}) \tag{2.100c}$$

$$\nabla^2 g(\xi) = \lambda I + \mathcal{T}_{I(\xi)}^\top \mathcal{T}_{I(\xi)} = \lambda I + \sum_{i \in I(\xi)} \mathcal{T}_i^\top \mathcal{T}_i \tag{2.100d}$$

are, respectively, the function to minimize, its gradient, and its generalized Hessian at ξ.

The proposed approach to solve (2.96) is summarized in Algorithm 3. The algorithm uses the solution d of the linear system

$$(\nabla^2 g(\xi) + \delta(\xi) I)d = -\nabla g(\xi) \tag{2.101}$$

at the current ξ as a search direction, where $\delta(\xi) = \zeta \|\nabla g(\xi)\|$ and $\zeta \in (0, 1)$. Due to the special structure of $\nabla^2 g$ in (2.100d), the linear system (2.101) is solved at Steps 5.1–5.2 as the least squares problem

$$\min_d \frac{1}{2} \left\| \begin{bmatrix} \mathcal{T}_{I(\xi)} \\ \sqrt{\lambda + \delta(\xi)} I_{n_\xi} \end{bmatrix} d + \begin{bmatrix} \mathcal{T}_{I(\xi)} \xi - \mathcal{R}_{I(\xi)} \\ \frac{\lambda}{\sqrt{\lambda+\delta(\xi)}} \xi \end{bmatrix} \right\|_2^2 \tag{2.102}$$

using the QR factorization of $\begin{bmatrix} \mathcal{T}_{I(\xi)} \\ \sqrt{\lambda+\delta(\xi)} I_{n_\xi} \end{bmatrix}$.

Note that since $\nabla g(\xi) > 0$ during iterations, $\delta(\xi)$ is also positive, and therefore R is full column rank, so that the upper-triangular linear system in Step 5.2 is always solvable.

A good initial guess for $\xi \in \mathbb{R}^n$ can be obtained by running Algorithm 3 first on decimated clusters, that is on maximum $\bar{N}_i$ regressors a_k per cluster and therefore $\bar{N} = \sum_{i=1}^s \bar{N}_i \ll N$ regressors, then use the result as the new initial condition in Algorithm 3 for the full problem with N regressors.

Numerical experiments have shown that allowing a varying $\zeta = \zeta_0 \frac{\min\{1, \|\nabla g\|\}}{\|\nabla g\|}$, where $0 < \zeta_0 \ll 1$, reduces the number of iterations and prevents excessive regularization in (2.101) when $\|\nabla g\|$ is large. Moreover, while setting $\lambda > 0$ complicates the numbers of operations required by the algorithm at each iteration (in particular to compute (2.103)) and bias the solution, it leads to a smaller number k of iterations, and overall to a reduced computation time.

Recursive multicategory discrimination via on-line convex programming. As an alternative to Algorithm 3, or in addition to it for refining the partition ϕ on line based on streaming data, we introduce a recursive approach to solve problem (2.96) based on techniques of on-line convex programming.

Let us treat the data-points $a \in \mathbb{R}^{n_a}$ as random vectors and assume that an oracle function $i : \mathbb{R}^{n_a} :\to \{1, \dots, s\}$ exists that to any $a \in \mathbb{R}^{n_a}$ assigns the corresponding mode $i(a) \in \{1, \dots, s\}$. Function i implicitly defines clusters in the data-point space $\mathbb{R}^{n_a}$. Let us also assume that the following values

$$\pi_i = \mathrm{Prob}[i(a) = i] = \int_{\mathbb{R}^{n_a}} \delta(i, i(a)) p(a) da$$

are known for all $i = 1, \dots, s$, where $\delta(i, j) = 1$ if $i = j$ or zero otherwise, $i, j \in \{1, \dots, s\}$. Each value π_i represents the relative "volume" of cluster i, where clearly

$$\sum_{i=1}^s \pi_i = \int_{\mathbb{R}^{n_a}} \sum_{i=1}^s \delta(i, i(a)) p(a) da = \int_{\mathbb{R}^{n_a}} p(a) da = 1.$$

Problem (2.96)–(2.97) can be generalized to the following unconstrained convex stochastic

Algorithm 3 Piecewise-smooth Newton method for solving the piecewise affine multicategory discrimination problem (2.96)

Input: Regressors $\{a_k\}_{k=1}^N$, clusters $\mathcal{C}_i$, $i = 1, \ldots, s$; scalars $\sigma \in (0, 1/2)$, $\zeta \in (0, 1)$; ℓ_2-regularization term $\lambda \geq 0$; initial guess $\xi \in \mathbb{R}^n$; maximum number K of iterations; tolerances $g_{\rm tol} > 0$ and $\delta_{\rm tol} > 0$.

1. **Initialize** matrices $M_i \in \mathbb{R}^{m_i \times n_a}$, whose rows are the transposed regressors $a_k \in \mathcal{C}_i$, $i = 1, \ldots, s$; $n_\xi \leftarrow s(n_a + 1)$, $n_g \leftarrow N(s-1)$; define $\mathcal{T}$, $\mathcal{R}$ as in (2.98)–(2.99), $j = 1, \ldots, n_g$;
2. $k \leftarrow 0$;
3. $c \leftarrow \mathcal{T}\xi - \mathcal{R}$; $I \leftarrow \{i \in \{1, \ldots, n_g\} : \ c_i \geq 0\}$;
4. $g \leftarrow c_I^\top c_I + \frac{\lambda}{2}\xi^\top \xi$; $\nabla g \leftarrow \mathcal{T}_I^\top c_I + \lambda \xi$; $\delta \leftarrow \zeta \|\nabla g\|$;
5. **while** $g > g_{\rm tol}$ **and** $\delta > \delta_{\rm tol}$ **and** $k < K$ **do**
   5.1. $(Q, R) \leftarrow$ QR factorization of matrix $\left[\begin{array}{c} \mathcal{T}_I \\ \sqrt{\lambda+\delta} I_{n_\xi} \end{array}\right]$;
   5.2. **solve** the upper-triangular linear system

$$\begin{aligned} R_{\{1,\ldots,n_\xi\}} d = & - \left(Q_{\{1,\ldots,|I|\},\{1,\ldots,n_\xi\}}\right)^\top c_I \\ & - \frac{\lambda}{\lambda+\delta} \left(Q_{\{|I|+1,\ldots,|I|+n_\xi\},\{1,\ldots,n_\xi\}}\right)^\top \xi; \end{aligned} \tag{2.103}$$

   5.3. $\alpha \leftarrow 1$; $q \leftarrow \mathcal{T} d$; $\xi_\alpha \leftarrow \xi + d$;
   5.4. $I_\alpha \leftarrow \{i \in \{1, \ldots, n_g\} : \ c + q \geq 0\}$;
   5.5. $g_\alpha \leftarrow (c_{I_\alpha} + q_{I_\alpha})^\top (c_{I_\alpha} + q_{I_\alpha}) + \frac{\lambda}{2}\xi_\alpha^\top \xi_\alpha$;
   5.6. **while** $g_\alpha > g + \alpha\sigma \nabla g^\top d$ **do**
      5.6.1. $\alpha \leftarrow \frac{1}{2}\alpha$; $\xi_\alpha \leftarrow \xi + \alpha d$
      5.6.2. $c^\alpha \leftarrow c + \alpha q$;
      5.6.3. $I_\alpha \leftarrow \{i \in \{1, \ldots, n_g\} : \ c_i^\alpha \geq 0\}$;
      5.6.4. $g_\alpha \leftarrow (c_{I_\alpha}^\alpha)^\top c_{I_\alpha}^\alpha + \frac{\lambda}{2}\xi_\alpha^\top \xi_\alpha$;
   5.7. **end while**;
   5.8. $\xi \leftarrow \xi_\alpha$; $g \leftarrow g_\alpha$; $I \leftarrow I_\alpha$; $c \leftarrow c_\alpha$;
   5.9. $\nabla g \leftarrow \mathcal{T}_{I_\alpha}^\top c_{I_\alpha}^\alpha + \lambda \xi$; $\delta \leftarrow \zeta \|\nabla g\|$;
   5.10. $k \leftarrow k + 1$;
6. **retrieve** ω^i, γ^i, $i = 1, \ldots, s$, from the solution ξ;
7. **end**.

Output: Coefficients ω^i, γ^i, $i = 1, \ldots, s$ defining the piecewise affine separator ϕ in (2.91)–(2.92).

optimization problem

$$\xi^* = \min_{\xi} E_{a \in \mathbb{R}^{n_a}} \left[\ell(a, \xi) \right] + \frac{\lambda}{2} \|\xi\|_2^2 \tag{2.104a}$$

$$\ell(a, \xi) = \sum_{\substack{j=1 \\ j \neq i(a)}}^{s} \frac{1}{\pi_{i(a)}} \left(a^\top (\omega^j - \omega^{i(a)}) - \gamma^j + \gamma^{i(a)} + 1 \right)_+^2 \tag{2.104b}$$

whose solution provides the piecewise affine multicategory discrimination function (2.91)–(2.92). This aims at violating the least, on average over a, the condition in (2.94) for $i = i(a)$. We assume that the regularization parameter λ is such that $\lambda > 0$, so that the objective function in (2.104a) is strongly convex.

When learning the discrimination function ϕ on-line, the data-points a_k are acquired in real-time and one would like to update ϕ recursively, without the need of storing all past data-points $a_0, \ldots, a_{k-1}$. We achieve this by solving Problem (2.104) by online convex optimization, and in particular the averaged stochastic gradient descent method of [43] as proposed in [44] (cf. also [45]), whose application to the piecewise affine multicategory discrimination problem (2.104) is described in Algorithm 4.

The initial estimate ξ_0 can be either zero (or any other value), or the result of the execution of the batch Algorithm 3 on a subset of data preprocessed offline. The coefficients π_i can also be estimated from offline data, namely $\pi_i = \frac{m_i}{N}$, and also possibly updated while Algorithm 4 is running by updating $m_{i(a_k)} \leftarrow m_{i(a_{k-1})} + 1$, $N \leftarrow N + 1$, and setting $\pi_{i(a_k)} = \frac{m_i}{N}$. Numerical experiments have shown however that constant and uniform coefficients $\pi = \frac{1}{s}$ work equally well.

Simulations. Extensive numerical experiments have been carried out to assess the performance of the proposed algorithm for PWARX system identification. The obtained results are reported in the recently submitted papers [46] and [47] and in the accompanying technical report [48]. Some of the obtained results are also reported in Table 2.1, which shows the CPU time required to generate the polyhedral partition of the regressor space in an academic example, with a 3-dimensional regressor space (i.e., $n_a = 3$). Specifically, the following four multicategory discrimination algorithms are used to generate the partition of the regressor space:

- *robust linear programming* (RLP) [32]. The solver *Gurobi* is used to compute the solution of the formulated linear programming problem;
- *regularized piecewise-smooth Newton* (RPSN) method (Algorithm 3);
- *averaged stochastic gradient descent* (ASGD) method (Algorithm 4), with $\lambda = 10^{-5}$ and $\nu_0 = 0.01$. The weights π_i and the initial estimate ξ_0 are computed by executing the batch Algorithm 3 on the first 50 training samples. The remaining training samples are processed recursively.
- *multicategory support vector machines* (MSVM) with linear kernels [49], implemented in the MSVMpack 1.5 toolbox [50]. The performance of the MSVM approach is evaluated only in relation to small/medium training sets, as large data sets take too long to be processed.

Algorithm 4 Averaged stochastic gradient descent algorithm for solving the piecewise affine multicategory discrimination problem (2.104)

Input: Regressor flow $a_0, a_1, \ldots$; cluster assignment function $i : \mathbb{R}^{n_a} \to \{1, \ldots, s\}$; ℓ_2-regularization term $\lambda > 0$; scalar $\nu_0 \geq 0$; initial guess $\xi \in \mathbb{R}^n$;

1. **for** $k = 0, 1, \ldots$ **do**:
   1.1. **compute** the gradient $\nabla_\xi \ell(\xi_k, a_k)$ as follows:
      1.1.1. $I_k \leftarrow \{j \in \{1, \ldots, s\}, j \neq i(a_k) : \ a_k^\top(\omega_k^j - \omega_k^{i(a_k)}) - \gamma_k^j + \gamma_k^{i(a_k)} \geq -1\}$;
      1.1.2. **set**

$$\frac{\partial \ell(\xi_k, a_k)}{\partial \left[\begin{smallmatrix}\omega^j \\ \gamma^j\end{smallmatrix}\right]} \leftarrow \lambda \left[\begin{smallmatrix}\omega_k^j \\ \gamma_k^j\end{smallmatrix}\right] + \frac{1}{\pi_{i(a_k)}} \times \tag{2.105}$$

$$\begin{cases} \displaystyle\sum_{j \in I_k} \left(a_k^\top(\omega_k^j - \omega_k^{i(a_k)}) - \gamma_k^j + \gamma_k^{i(a_k)} + 1\right) \left[\begin{smallmatrix}-a_k \\ 1\end{smallmatrix}\right] \\ \qquad \text{if } j = i(a_k) \\ \left(a_k^\top(\omega_k^j - \omega_k^{i(a_k)}) - \gamma_k^j + \gamma_k^{i(a_k)} + 1\right) \left[\begin{smallmatrix}a_k \\ -1\end{smallmatrix}\right] \\ \qquad \text{if } j \neq i(a_k),\ j \in I_k \\ 0 \text{ otherwise.} \end{cases}$$

   1.2. **compute**

$$\nu_k \leftarrow \nu_0(1 + \nu_0 \lambda k)^{-\frac{3}{4}}; \tag{2.106a}$$
$$\mu_k \leftarrow 1/\max\{1, k - n_a, k - n_\xi\}; \tag{2.106b}$$
$$\xi_{k+1} \leftarrow \xi_k - \nu_k \nabla_\xi \ell(\xi_k, a_k); \tag{2.106c}$$
$$\bar{\xi}_{k+1} \leftarrow \bar{\xi}_k + \mu_k(\xi_{k+1} - \bar{\xi}_k); \tag{2.106d}$$

   1.3. **retrieve** ω_k^i, γ_k^i, $i = 1, \ldots, s$, from $\bar{\xi}_k$;
2. **end**.

Output: Coefficients ω_k^i, γ_k^i, $i = 1, \ldots, s$ defining the piecewise affine separator ϕ in (2.91)–(2.92) at each step $k = 0, 1, \ldots$.

Note that, for a large training set (i.e., $N = 125000$), Algorithms 3 and 4 are about 454x and 65200x faster, respectively, than the robust linear programming method of [32].

Conclusions. The strengths of the PWA regression approach developed during the project and described above are its computational efficiency and the ability to be run both in a batch and in a recursive way. Future research will be devoted to generalize the approach to piecewise-nonlinear models (such as piecewise polynomial) by feeding regression data manipulated through nonlinear basis functions to Algorithms 2, 3 and 4.

We have proposed a novel approach for the regression of PWA maps, that is successfully employed for identi

fication of nonlinear dynamical systems in a PWARX form. The approach consists of two stages: first, simultaneously cluster the observed regressors and adaptive estimate an affine submodel for each cluster; second, solve a convex multi-category discrimination problem via a regularized piecewise-smooth Newton method to determine a piecewise linear separator function and a polyhedral partition of the regressor space. This algorithm is more than two orders of magnitude faster than classical piecewise affine multicategory discrimination methods available in the literature. Furthermore, we have developed an averaged stochastic gradient descent algorithm to solve the same problem recursively, by processing one data sample at the time, so that the overall algorithm can be implemented in real-time, such as for adaptive control purposes.

Future research will be devoted to generalize the approach to piecewise-nonlinear models (such as piecewise polynomial) by feeding regression data manipulated through nonlinear basis functions to Algorithms 2, 3 and 4.

2.2 Control configuration selection

Prior to the synthesis of controller parameters for a system with several manipulated and controlled variables, a low complexity control configuration is often selected [52]. In Control Configuration Selection (CCS), one approach is to simplify the complete system model by selecting the most important input-output interconnections of the system and composing a reduced model from the selected components. The resulting reduced model will then be used in the control design, and all interconnections of the reduced model should be considered in the feedback structure of the closed loop system.

	$N = 1250$	$N = 12500$	$N = 125000$
RLP [32]	1.336 s	125 s	8541 s
RPSN (Algorithm 3)	0.257 s	1.762 s	18.8 s
ASGD (Algorithm 4)	0.0014 s	0.018 s	0.131 s
MSVM [51]	6.545 s	3870 s	–

Table 2.1: CPU time required to partition the regressor space vs length N of the training set. The computations are carried out on an i7 2.40-GHz Intel core processor with 4 GB of RAM running MATLAB R2014b.

When it comes to industry applications, the normal approach is to select control configurations based on previous knowledge, intuition & common sense, rules of thumb, or even geographical proximity between sensors and actuators. Therefore control configurations are often selected without the use of theoretical tools or systematic procedures.

By applying CCS methods on the processes under study, we will obtain indications on the adequacy of the control configurations used at the processes prior to the start of DISIRE, as well as an indication of the potential of creating controllers of larger complexity with e.g. an MPC scheme.

2.2.1 Introduction to Control Configuration Selection using Interaction Measures

The selection of the reduced model in CCS is often performed with the use of the so-called Interaction Measures (IMs), which include the popular Relative Gain Array (RGA) introduced by Bristol in [53]. A limitation of the RGA is that it is only applicable in the design of decentralized control configurations, where sensors and actuators are grouped in pairs and the loops are closed using only Single-Input-Single-Output controllers. This limitation led to the later introduction of the gramian-based IMs in [54].

Whilst the RGA can be used for an initial analysis based on DC-gains, it is recommended to complete the analysis with the so-called gramian-based IMs, which provide information based on the dynamics of the model and can be used to design sparse control configurations. In the automatic control framework, it is well understood that system gramians can be used to identify the most significant portions of a system model, in terms of controllability and observability.

In DISIRE, we created a gramian-based IM named Prediction Error Index Array (PEIA), which is based both on controllability analysis and on measuring the prediction error committed by the model simplifications performed during CCS. The results are based on the use of the $\mathcal{H}_2$-norm, which was first used in the Gramian-based IM known as Σ_2 for quantifying the output controllability [55]. This norm was further studied in [56], where additional interpretations of its square value were derived, making it suitable for CCS. Based on these interpretations, methods for CCS using weighted graphs were proposed in [57], in which the square of the $\mathcal{H}_2$-norm used to determine the weights for the graph edges. We demonstrate in this report that using the squared $\mathcal{H}_2$-norm to define an IM allows the selection of a reduced model in terms of the power of the prediction error committed by a white Gaussian excitation signal.

PEIA has been created as synergy between DISIRE and the EU project OPTi. The results presented in this document for linear systems have also been extended to nonlinear systems using Volterra series. Volterra series are often used to generalize concepts for their application on nonlinear systems and have been introduced in [58]. It will be pursued by LTU to use this nonlinear theory for CCS in the future of DISIRE. A requirement for this purpose is the availability of nonlinear process models in the shape of Volterra series.

In 2.2.2 we give the preliminaries on the RGA, and the extension of the RGA to the frequency domain called DRGA is given in 2.2.3.

2.2.2 Relative Gain Array

The RGA of a continuous process $G(s)$ with equal number of inputs and outputs is defined as

$$RGA(G) = G(0) \otimes G(0)^{-\top} \tag{2.107}$$

where $G(0)^{-\top}$ is the transpose of the inverse of $G(0)$, and $\otimes$ denotes element by element multiplication. The original definition of the RGA to squared systems was expanded for its application on non-square plants with the use of the pseudo-inverse in [59].

Properties of the RGA

- The RGA is normalized, so the sum of all the elements of each row or column add up to one.
- The RGA is scaling invariant.
- The RGA of a triangular or diagonal matrix is the identity.

Interpretation of the RGA In order to explain the reasoning about the pairing of the inputs and outputs based on the steady-state RGA we will use the definition of RGA used by Bristol in [53].

For each input-output pairing u_j, y_i, the DC gains in a multivariable system have to be evaluated in two extreme cases:

– All the other loops opened, with all the other inputs $u_k, \forall k \neq j$ kept constant. This is equivalent to obtain the DC gain of the plant $G(s)$ from the g_{ij} element.

$$\left(\frac{\partial y_i}{\partial u_j}\right)_{u_k, \forall k \neq j} = g_{ij}$$

– All the other loops closed, with all the other outputs $y_k, \forall k \neq i$ kept constant in what is assumed to be perfect control. A change in the input u_j will yield a change in y_i, but also to a change in all the other outputs which are controlled under perfect control; the other inputs $u_k, \forall k \neq j$ will also change in order to compensate the variation of the outputs $y_k, \forall k \neq i$, and this will lead to a new change in the observed output y_i due to interaction. Then, we evaluate:

$$\left(\frac{\partial y_i}{\partial u_j}\right)_{u_k, \forall k \neq j} = \widehat{g}_{ij}$$

The element λ_{ij} of the RGA is then defined as

$$\lambda_{ij} = \frac{g_{ij}}{\widehat{g}_{ij}} = \frac{((\partial y_i)/(\partial u_j))_{u_k, \forall k \neq j}}{((\partial y_i)/(\partial y_i))_{y_k, \forall k \neq i}}$$

The RGA is related to the condition number of a plant [60], in such a way that large numbers of the RGA are related to ill-conditioned systems, which are hard to control with decentralized controllers.

Pairing rules of the RGA The pairing rules for the static RGA can be summarized as:

- A value of 1 in λ_{ij} means that the value of g_{ij} is not affected for the closing of the other loops, so there is no interaction effects in the pairing $u_j - y_i$. Pairings with values of λ close to one will be preferred.
- A value of λ_{ij} close to 0, means that the input u_j should not be used to control the output y_i.
- Pairings with negative values of the RGA should be avoided; a negative value in λ_{ij} means that the gain of the subsystem formed by the j^{th} input and the i^{th} output changes its sign when all the other loops are closed.
- Pairings with large numbers in the RGA should be avoided; large numbers are related to ill-conditioned plants, so in this case, some other control techniques may be considered.

Limitations of the RGA

- It is based on DC gains and therefore is not able to capture process dynamics in the decision making. These process dynamics include the existence of time delays or non-minimum phase zeroes. To deal with this limitation, several variants of the RGA have been introduced, like the DRGA, ERGA or RNGA.
- The RGA is not directly applicable to systems with pure integrators, since these systems have an infinite DC-gain. However, an alternative for these kind of systems have been proposed [61–63].
- Inability to capture loop interactions in triangular structures.

A detailed description of the RGA as well as several examples can be found in [64].

2.2.3 Dynamic Relative Gain Array (DRGA)

Several authors created indicators under the name Dynamic RGA in order to obtain indications in the frequency domain based on relative gains. The indicator discussed here is the straightforward approach of evaluating the RGA in the frequency domain, and can be used to design control structures at any desired frequency [65]. The DRGA of a continuous process described by a transfer function $G(s)$ is:

$$DRGA(\omega) = G(j\omega) \otimes G(j\omega)^{-\top} \tag{2.108}$$

The DRGA is a complex number and has a more obscure interpretation than that of the RGA: it is usually preferred to use its magnitude as indicator due to the gain interpretation, however only the sums of the rows or columns of the resulting complex array (or its real part) add up to 1. Moreover, by evaluating the magnitude alone, the sign of the DRGA is lost as an indicator, which is often used to rule out certain input-output pairings.

The DRGA is sensitive to time delays. Time delays in continuous time systems can be considered when computing the DRGA by using approximations (i.e., Padé).

2.2.4 Prediction Error Index Array

This subsection introduces an IM for linear systems which gives indications based on the prediction error of the reduced model selected during CCS.

Preliminaries on linear systems are first given, followed by the introduction of the new linear IM named Prediction Error Index Array (PEIA). Later, the derivation of relationships of PEIA with absolute and a relative measures of the prediction error are given.

2.2.4.1 Preliminaries on gramians and $\mathcal{H}_2$-norm

Assume a multivariable linear process with n inputs and m outputs described by the following state space representation:

$$\begin{aligned} \dot{x(t)} &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

where $u \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ and $x \in \mathbb{R}^p$ are the input, output and state vectors. The process can alternatively be represented by the mutivariable transfer function $G(s) = C(sI - A)^{-1}B$ or by the impulse response matrix $g(t)$ which is calculated as the inverse Laplace transform of $G(s)$.

The IM Σ_2 has been defined in [55] as:

$$\Sigma_2 = \frac{||G_{ij}||_2}{\sum_{k,l=1}^{m,n} ||G_{kl}||_2}$$

where $||G_{ij}||_2$ is the $\mathcal{H}_2$ norm of $G_{ij}(s)$.

Different calculations of the $\mathcal{H}_2$-norm are:

$$\begin{aligned} ||G_{ij}||_2 &= \sqrt{\frac{1}{2\pi}\int_{-\infty}^{\infty} |G_{ij}(j\omega)|^2 \mathrm{d}\omega} = \sqrt{\int_0^{2\pi} g_{ij}^2(\tau)\mathrm{d}\tau} \\ &= \sqrt{\mathrm{trace}(B_j^\top Q_i B_j)} = \sqrt{\mathrm{trace}(C_i P_j C_i^\top)} \end{aligned}$$

where P_j is the controllability gramian related to the jth input, Q_i is the controllability gramian related to the ith input, B_j is the j-th row of B, and C_i is the i-th row of C, **j** is the imaginary unity, and ω is frequency in rad/sec.

$$P_j = \int_0^{\infty} e^{A\tau} B_j B_j^\top e^{A^\top \tau} \mathrm{d}\tau;\ Q_i = \int_0^{\infty} e^{A^\top \tau} C_i^\top C_i e^{A\tau} \mathrm{d}\tau$$

The calculation of the integral in of the squared magnitude in the frequency domain gives the interpretation of the squared $\mathcal{H}_2$-norm as the output energy when the input is exited with an input signal with unitary flat power spectral density (psd). The calculation from the impulse response gives an interpretation of the squared $\mathcal{H}_2$-norm as the energy of the impulse response of the system. The calculation through the gramians gives an interpretation in terms of output controllability (see [55]).

2.2.4.2 Definition of the linear Prediction Error Index Array.

The discussed interpretations indicate that it is appropriate to square the $\mathcal{H}_2$-norm when used to quantify the significance of the input-output channels of a process. We therefore adapt the original definition of Σ_2, and define an IM named Prediction Error Index Array (PEIA) as:

$$\mathrm{PEIA} = \frac{||G_{ij}||_2^2}{\sum_{k,l=1}^{m,n} ||G_{kl}||_2^2} = \frac{||G_{ij}||_2^2}{||G||_2^2} \tag{2.109}$$

The name PEIA refers to the direct relationship of each element of this IA with the prediction error committed when neglecting the corresponding input-output channel. This relationship will be proven in Subsection 2.2.4.5.

In addition to the more direct interpretations with the use of the squared $\mathcal{H}_2$-norm, the sum of the individual metrics of the input-output channels in PEIA is equal to the metric of the complete system:

$$\sum ||G_{ij}||_2^2 = ||G||^2$$

This property is a consequence of the gramian decomposition, and therefore the elements in PEIA express the contribution of each input-output channel as a fraction of the global contribution. This property is preserved by the first introduced gramian-based IMs named Participation Matrix and Hankel Interaction Index Array (see [54, 66]), but not by Σ_2.

To achieve a well-decoupled system with a diagonal sensitivity function, it is necessary that the structure of the controller is the inverse of the plant. The control configuration is therefore designed by selecting the most important input-output channels resulting in a model with a reduced structure which can achieve a sufficiently close to diagonal sensitivity function. The user can comprehend the amount of the total process dynamics that the reduced model is reflecting by evaluating the closeness of the total contribution of the selected channels to 1, which corresponds to 100%.

Details can be found in the literature on procedures and rules to follow during the selection of a control configuration from an gramian-based IA [54]. Since this selection is often trivial from the direct interpretation of the IA, no details are given in this report. The resulting configurations in the examples are related to the simplest reduced model with a contribution larger than 70%. This would be a preliminary configuration which has to be tested, leading to a possible redesign in favor to a configuration with larger contribution.

2.2.4.3 Analysis using the $\mathcal{H}_2$-norm with the Prediction Error Index Array (PEIA)

The Prediction Error Index Array (PEIA) is an array which quantifies the importance of the input-output channels. The sum of all the elements in this array equals 100%. It can be used to select a reduced model with the most important input-output interconnections and base control on that model. If you sum the contribution of all the selected input-output channels, you obtain a measure (in %) of how much of the reduced model is able to capture the total process dynamics. There are three different interpretations of the PEIA:

- PEIA can be interpreted in terms of energy or power transfer. If we assume that the inputs are excited by uncorrelated white noise of unit variance, then each of the numbers in the PEIA is the energy transfer for each input-output channel in relation to the total energy transfer. As an analogy, we can think of each of the input-output channels as being a pipe which is able to transfer flows, and the value of the PEIA indicates the thickness of the pipe, we would therefore choose the larger pipes for control and focus future modeling efforts on them.
- PEIA can be interpreted in terms of output controllability.
- PEIA can be interpreted in terms of the prediction error committed when selecting a reduced model. Lets say that we select a subset of the input-output channels as reduced model denoted by G^*, being the original model denoted as G. Assume that both G^* and G are excited by the same white noise sequence, and denote as $y^*(t)$ and $y(t)$ as the outputs from these models. Then adding the PEIA value of the elements considered in G^* is a relative measure of the energy of the prediction error $y^*(t) - y(t)$

2.2.4.4 Absolute measure of the prediction error.

This subsection aims to find a measure of the prediction error committed by the reduced model selected during CCS, and relate it this measure to PEIA.

Denote by:

- $\hat{G}(\omega)$: reduced model on which control will be based.
- $\Delta G(\omega)$: model composed by the disregarded IO channels.
- $\hat{y}(t) \in \mathbb{R}^{m,1}$: output from the reduced model $\hat{G}$.
- $y_\Delta(t)\mathbb{R}^{m,1}$: output from the model composed by the disregarded IO channels in ΔG.

The prediction error is defined as the difference $y_\Delta = y - \hat{y}$, and the considered metric in the prediction error is its average power.

$$\begin{aligned} P(y(t) - \hat{y}(t)) &= \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} y_\Delta^\top \cdot y_\Delta \mathrm{d}t \\ &= \int_{-\infty}^{\infty} \mathrm{trace}(S_{y_\Delta y_\Delta}(f))\mathrm{d}f \end{aligned}$$

where $S_{y_\Delta y_\Delta} \in \mathbb{R}^{m,m}$ is the power spectral density (psd) of the prediction error $y_\Delta(t)$. The psd of the output of a linear system can be expressed as a function of the psd of its input S_{uu}, leading to:

$$P(y(t) - \hat{y}(t)) = \int_{-\infty}^{\infty} \mathrm{trace}(\Delta G(-f) \cdot S_{uu}(f) \cdot \Delta G(f)^\top)\mathrm{d}f$$

We will assume at this step that the input $u_i(t)$ are uncorrelated sequences and have a flat psd. We also assume that the input have been scaled in such a way that $S_{uu}(f) = I$. Then, the average power of the prediction error is:

$$\begin{aligned} P(y(t) - \hat{y}(t)) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathrm{trace}(\Delta G(-\omega) \cdot \Delta G(\omega)^\top)\mathrm{d}\omega \\ &= ||\Delta G||_2^2 = \sum_{i,j} ||\Delta G_{ij}||_2^2 \end{aligned}$$

Therefore, the squared $\mathcal{H}_2$-norm of the disregarded IO channels is the average power of the prediction error when the input signals are uncorrelated sequences and have flat unitary psd.

For the case of continuous-time systems, an input signal with flat psd over all frequencies is not realizable since it has infinite energy. A band limited noise with flat band in an interval $[a, b]$ can be used, leading to the following integral:

$$P(y(t) - \hat{y}(t)) = \frac{1}{\pi} \int_a^b \mathrm{trace}(\Delta G(-\omega) \cdot \Delta G(\omega)^\top) \mathrm{d}\omega$$

which is aligned to the concept of frequency-limited $\mathcal{H}_2$-norm (see [67]), which was used in [68] to design control configurations for a restricted set of frequencies.

2.2.4.5 Relative measure of the prediction error

A relative measure of the prediction error will be sought under the assumption that the inputs are uncorrelated and have flat psd.

Start by relating the power of the output y to the power of $\hat{y}$ and y_Δ.

$$\begin{aligned} P(y(t)) &= \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} (y_\Delta(t) + \hat{y}(t))^\top \cdot (y_\Delta(t) + \hat{y}(t)) \mathrm{d}t \\ &= \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} \left(y_\Delta(t)^\top y_\Delta(t) + \hat{y}(t)^\top \hat{y} + 2 y_\Delta(t)^\top \hat{y}(t) \right) \mathrm{d}t \end{aligned}$$

Each of the outputs $[y_\Delta(t)]_i$ and $[\hat{y}(t)]_i$ from ΔG and G, are zero mean stochastic processes for being the output of linear systems to a 0 mean stochastic input. Additionally $[y_\Delta(t)]_i$ and $[\hat{y}(t)]_i$ are clearly uncorrelated, since the structures of ΔG and G are complementary, and therefore $[y_\Delta(t)]_i$ and $[\hat{y}(t)]_i$ have contributions from different inputs. Therefore $E([y_\Delta(t)]_i \cdot [\hat{y}(t)]_i) = 0$, and:

$$\begin{aligned} P(y(t)) &= \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} \left(y_\Delta(t)^\top y_\Delta(t) + \hat{y}(t)^\top \hat{y} \right) \mathrm{d}t \\ &= P(y_\Delta(t)) + P(\hat{y}(t)) \end{aligned}$$

The average power of the output is then the power of the output from the reduced model plus the average power of the output from the disregarded channels ΔG. A relative measurement of the prediction error is constructed as:

$$\frac{P(y(t) - \hat{y}(t))}{P(y(t))} = \frac{||\Delta G||_2^2}{||G||_2^2} = \sum_{i,j} \frac{||\Delta G_{ij}||_2^2}{||G||_2^2} \tag{2.110}$$

The sum of the disregarded elements of the PEIA is then a relative measure of the average power of the prediction error in relation to the average power of the original model when the inputs are uncorrelated zero mean processes with flat psd.

Another possible measure to use is the ratio of the power of the output of the original model

which is explained by the reduced model:

$$\frac{P(\hat{y}(t))}{P(y(t))} = \sum_{i,j} \frac{||\hat{G}_{ij}||_2^2}{||G||_2^2} = 1 - \frac{P(y(t) - \hat{y}(t))}{P(y(t)))} \tag{2.111}$$

2.3 Sparse Control Using Convex Optimization Techniques

In this section, we introduce a method for CCS using convex optimization which finds a trade-off between the achievable performance of the resulting configuration and its sparsity.

The relationship of the CCS to sparsity is that CCS be formulated as finding a low complexity controller which minimizes the amount of interconnections between measurements and actuators while achieving acceptable performance. Since the controller is often formulated as a transfer function matrix which connects measurements with control actions, minimizing the amount of the interconnections in the control system is equivalent to maximizing the sparsity of the controller matrix.

In addition to the maximization of the controller sparsity, an important breakthrough of the proposed method in comparison with the gramian-based IMs is the explicit use of closed-loop performance in the selection of the control configuration.

Formulating the CCS problem as a convex problem is also important in terms of practical implementation, even if this requires the use of approximations. This is due to the fact that CCS is naturally a combinatorial problem which leads to the use of mixed integer linear programming, where the achievable performance and complexity of all possible control configurations should be evaluated. The best control configuration would be the one which results in a better trade-off between complexity and achievable performance.

As example, the walking beam furnace from MEFOS which is considered in this project has 7 actuators and 4 sensors. Considering at each of the sections the combustion oil and combustion air as a single actuating pair for the design of the configuration, then the number of possible configurations to test is in the order of 29 million. This number will dramatically increase with the addition of the sensors which are under construction in DISIRE for the measurement of local temperatures at the slabs.

For the theory described in this section,we consider a discrete-time system represented by the equation:

$$y(t+1) = \sum_{r=0}^{N} A_r(t) \cdot y(t-r) + \sum_{r=0}^{N} B_r(t) \cdot u(t-r) + \sum_{r=0}^{N+1} W_r(t)\dot{\omega}(t-r+1) \tag{2.112}$$

were $y(t) \in \mathbb{R}^m$, $u(t) \in \mathbb{R}^n$ and $\omega(t) \in \mathbb{R}^{p+1}$ are the output, input and disturbance vectors at time t, and $A_r \in \mathbb{R}^{m \times m}$, $B_r \in \mathbb{R}^{m \times n}$ and $W_r \in \mathbb{R}^{m \times p}$. For the formulation of LMIs for Lyapunov stability, it is convenient to keep the same index r for the sets of matrices $\{A_r\}$, $\{B_r\}$ and $\{W_r\}$. Note that this process equation allows to represent the output as a function of: i) the history of the output (see A_r), ii) the history of the input (see B_r) and iii) the history of the process disturbances (see W_r).

We assume a control law of the form:

$$u(t) = \sum_{s=1}^{Ns} K_s(t) \cdot y(t-s) \tag{2.113}$$

where the matrices K_r represent the linear dependance of the current control action from the error history $e(t)$.

We seek a convex formulation of the form:

$$\{K_r\} = \arg\min \Gamma(\{A_r\}, \{B_r\}, \{K_p\}) + \lambda \cdot \mathrm{str}(\{K_p\}) \tag{2.114}$$

$$\text{s.t. } R(A_r, B_r, K_p) \leq \beta \tag{2.115}$$

where Γ is a performance function which is minimized, and $\mathrm{str}(\{K_p\})$ is a function which quantifies the complexity of the controller configuration, λ is a regularization parameters which regulates the trade-off in the cost function, N is the number of time samples which influence in the process dynamics, and $R(A_r, B_r, K_p) \leq \beta$ are a set of inequalities which can be formulated to express e.g. stability. We seek that the set of constrains R formalize Lyapunov stability of the closed loop system.

Each of the following sections formalizes a sub-problem which has to be resolved.

2.3.1 Find a convex performance function $\Gamma(u(t), y(t))$

The goal is to formulate a convex cost function of the manipulated variables $u(t)$ inputs $y(t)$. In closed loop, $u(t)$ and $y(t)$ would be a function of the process disturbances $\omega(t)$, which would be the exogeneous input to the closed loop process. Then we can choose an arbitrary convex function Γ, like e.g., a quadratic cost. The first step is therefore to find a function relating the process disturbances to the outputs and manipulated variables. For this purpose we start by collecting the T steps history of the process variables as $\bar{y}$, $\bar{u}$ and $\bar{\omega}$.

$$\bar{y} = \begin{bmatrix} y_1(0) \\ \vdots \\ y_m(0) \\ y_1(1) \\ \vdots \\ y_m(1) \\ \vdots \\ y_1(T+1) \\ \vdots \\ y_m(T) \end{bmatrix} \in \mathbb{R}^{(T+1)m},\ \bar{u} = \begin{bmatrix} u_1(1) \\ \vdots \\ u_n(1) \\ u_1(2) \\ \vdots \\ u_n(2) \\ \vdots \\ u_1(T-1) \\ \vdots \\ u_n(T-1) \end{bmatrix} \in \mathbb{R}^{(T-1)n},\ \bar{\omega} = \begin{bmatrix} \omega_1(0) \\ \vdots \\ \omega_p(0) \\ \omega_1(1) \\ \vdots \\ \omega_p(1) \\ \vdots \\ \omega_1(T) \\ \vdots \\ \omega_n(T) \end{bmatrix} \in \mathbb{R}^{(T+1)p} \tag{2.116}$$

Therefore, finding a function connecting the process disturbances to the manipulated variables and outputs is equivalent to find $P_{\bar{y}\bar{\omega}}$ and $P_{\bar{u}\bar{\omega}}$ in:

$$\begin{bmatrix} \bar{y} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} P_{\bar{y}\bar{\omega}} \\ P_{\bar{u}\bar{\omega}} \end{bmatrix} \cdot \bar{\omega} \tag{2.117}$$

Assuming for simplicity that $y(0) = 0$, we have that:

$$\bar{y} = \alpha\bar{y} + \beta\bar{u} + \Omega\bar{\omega} \tag{2.118}$$

where

$$\alpha = \begin{bmatrix} 0 & 0 & 0 & \dots \\ A_0(0) & 0 & 0 & \dots \\ A_1(1) & A_0(1) & 0 & \dots \\ A_2(2) & A_1(2) & A_0(2) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \beta = \begin{bmatrix} 0 & 0 & 0 & \dots \\ B_0(0) & 0 & 0 & \dots \\ B_1(1) & B_0(1) & 0 & \dots \\ B_2(2) & B_1(2) & B_0(2) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

$$\Omega = \begin{bmatrix} 0 & 0 & 0 & \dots \\ W_1(0) & W_0(0) & 0 & \dots \\ W_2(1) & W_1(1) & W_0(1) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{2.119}$$

Solving (2.118) for $\bar{y}$:

$$\bar{y} = (I - \alpha)^{-1}\beta\bar{u} + (I - \alpha)^{-1}\Omega\bar{w} \tag{2.120}$$

where we will denote $\phi = (I - \alpha)^{-1}$ for simplicity in the following derivations.

The history of the control action can then be expressed as:

$$\bar{u} = \kappa \cdot \bar{y} \tag{2.121}$$

where

$$\kappa = \begin{bmatrix} K_0(1) & 0 & 0 & \dots \\ K_1(2) & K_0(2) & 0 & \dots \\ K_2(1) & K_1(1) & K_0(1) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{2.122}$$

Substituting (2.121) in (2.120) and solving for $\bar{y}$, we find $P_{\bar{y}\bar{\omega}}$:

$$P_{\bar{y}\bar{\omega}} = (I - \phi\beta\kappa)^{-1}\phi\Omega \tag{2.123}$$

Substituting (2.120) in (2.121) and solving for $\bar{y}$, we find $P_{\bar{u}\bar{\omega}}$:

$$P_{\bar{u}\bar{\omega}} = (I - \kappa\phi\beta)^{-1} \cdot \kappa\phi\Omega \tag{2.124}$$

The problem of using $P_{\bar{y}\bar{\omega}}$ and $P_{\bar{u}\bar{\omega}}$ is that they are not convex functions of the design parameters κ. We seek a change of variable which makes $P_{\bar{u}\bar{\omega}}$ convex. To this aim, we first use the following matrix identities to rearrange $P_{\bar{u}\bar{\omega}}$ and $P_{\bar{y}\bar{\omega}}$:

$$(I + AB)^{-1} = I - A(I + BA)^{-1}B \tag{2.125}$$

$$(I + AB)^{-1} \cdot A = A(I + BA)^{-1} \tag{2.126}$$

resulting in:

$$P_{\bar{y}\bar{w}} = \phi\Omega + \phi\beta \underbrace{(I + \kappa\phi\beta)^{-1}\kappa}_{Q} \phi\Omega \tag{2.127}$$

$$P_{\bar{u}\bar{\omega}} = \underbrace{(I - \kappa\phi\beta)^{-1}\kappa}_{Q} \phi\Omega \tag{2.128}$$

We can then make the change of variable $Q = (I + \kappa\phi\beta)^{-1}\kappa$ and then $P_{\bar{y}\bar{w}}$ and $P_{\bar{u}\bar{w}}$ become affine functions of Q.

$$P_{\bar{y}\bar{w}} = \phi\Omega + \phi\beta Q\phi\Omega \tag{2.129}$$

$$P_{\bar{u}\bar{\omega}} = Q\phi\Omega \tag{2.130}$$

We can then propose any convex cost function $\Gamma(\bar{u}, \bar{y})$ as a function of Q:

$$\Gamma(\bar{y}, \bar{u}) = \Gamma(P_{\bar{y}\bar{\omega}} \cdot \bar{\omega}, P_{\bar{u}\bar{\omega}}\bar{\omega}) = \Gamma(Q\phi\Omega\bar{\omega}, \phi\Omega\bar{\omega} + \phi\beta Q\phi\Omega\bar{\omega}) \tag{2.131}$$

The selection of $\Gamma(\bar{y}, \bar{u})$ is arbitrary, and could be for example a quadratic cost function of the measured and manipulated variables. The optimization problem to be resolved would be

$$\begin{aligned} &\min_{Q} \Gamma(Q\phi\Omega\bar{\omega}, \phi\Omega\bar{\omega} + \phi\beta Q\phi\Omega\bar{\omega}) \\ &\qquad s.t.\ Q \text{ lower triangular} \end{aligned} \tag{2.132}$$

when the optimal Q is found we can then retrieve κ as:

$$\kappa = (I - Q\phi\beta)^{-1}Q = Q(I - \phi\beta Q)^{-1} \tag{2.133}$$

The problem now is that in the cost function we wanted to add sparsity of κ, but we did a change of variable and our design parameter in the cost function is Q.

2.3.2 Find a convex function for the structure of the controller

A traditional complexity cost function is that one which counts the number of interconnections used in the control system, that is, the total number of times that the history of an error signal $e_i(t)$ is used to calculate a control action $u_j(t)$. For the control law in (2.113), we would ideally use the following function:

$$\text{str}(\{K_p\}) = ||\sum_{p=o} K_p||_0 \tag{2.134}$$

where each non-zero element in the matrix $\sum_{p=0} K_p$ translates intro a connection between an error signal and an actuator. Therefore, the 0-norm quantifies the number of these interconnections. The problem is however that the 0-norm is not a convex function, and therefore the 0-norm problem is normally approximated by the 1-norm leading to the following structure function:

$$\text{str}(\{K_p\}) = ||\sum K_p||_1 \tag{2.135}$$

After stating the performance cost function, the problem is that we performed a change of variable, and whilst we would like to minimize on κ, only Q is accessible for optimization. Minimizing the 1-norm on Q doesn't imply sparsity on κ, and an alternative has to be sought. The idea is that post-multiplication by a matrix which includes a column of zeros results in a matrix with a column of zeros in the same position. The same happens with the rows when we have a matrix pre-multiplication. As it can be observed in (2.133), in the calculation of κ from Q, the matrix Q can be either pre-multiplying or post-multiplying, which indicates tat rows and columns with zeros in Q are translated as rows and columns with zeros in κ. We can then impose row and column sparsity on Q and it will be translated into row and column sparsity in κ. The pseudo-norms for row and column sparsity for $X \in \mathcal{R}^{m\times n}$ are:

Row sparsity

$$||X||_{r,0}= \sum_{i=1}^{m} \max_{1\leq j\leq n} |\text{sign}(X_{ij})| \tag{2.136}$$

Column sparsity

$$||X||_{c,0}= \sum_{j=1}^{n} \max_{1\leq i\leq m} |\text{sign}(X_{ij})| \tag{2.137}$$

These pseudo-norms calculate the number of rows/columns whih have elements different than zero. The problem is that these pseudo-norms are not convex, so we have to use following 1-norm approximations

Row sparsity

$$||X||_{r,1}= \sum_{i=1}^{m} \max_{1\leq j\leq n} |X_{ij}| \tag{2.138}$$

Column sparsity

$$||X||_{c,0}= \sum_{j=1}^{n} \max_{1\leq i\leq m} |X_{ij}| \tag{2.139}$$

$$\min_{Q} \Gamma(Q\phi\Omega\bar{\omega}, \phi\Omega\bar{\omega} + \phi\beta Q\phi\Omega\bar{\omega}) + \lambda(||Q||_{c,1}+||Q||_{r,1})$$
$$s.t.\ Q \text{ lower triangular} \tag{2.140}$$

2.3.3 Application of the theoretical results on Sparse Control

The theoretical results on sparse control introduced in this section have been generated in synergy with the research project WARP funded by the Swedish program PiiA. The project WARP deals with the control of industrial processes over wireless sensors and actuators, where the introduced method is expected to be able to select which sensors or actuators would be used depending on possible communication problems like jitter or packet loss.

Chapter 3

Control problem statement

In this chapter, we are going to go through the building blocks of the basic closed-loop control paradigm we present in Figure 3.1. An identification module produces simple control-oriented dynamical models of the controlled plant (more details for which we provide in Chapters 4, 5 and 5). In Section 3.1 we introduce the scenario approach which we later use in Section 3.2 to formulate the scenario-based control problem. Notice that the system identification module provides to the controller both a description of the system dynamics (modelled part) as well as a representation of its uncertainty (unmodelled part). Further, a controller calibrator module will fine-tune the SMPC controller giving more gravity either to the fulfillment of the control objectives, the reduction of fuel consumption or other objectives, so as to lead to a good closed-loop performance.

3.1 Representation of modelling uncertainty

In this section we introduce the scenario-based modelling and control approach. It is rather common in stochastic approaches to assume that the uncertainty assumes an algebraicly favourable form, e.g., it is log-concave or, very often, it is assumed that is follows a normal distribution. The requirement that modelling errors are independent and identically distributed is frequently further imposed. It is easy to understand that such assumptions are rather restrictive, do not reflect how the plant is operated and are likely to lead to erroneous results, bad performance and little protection from unexpected realisation of the uncertainty.

As we can see in Figure 3.2 and later in Chapters 4 and 5, the predicted temperatures of the walking beam furnace and the ethylene cracking furnace, do not follow the normal distribution and are definitely not time-invariant[1].

3.1.1 Scenario fans and trees

Scenario fans and scenario trees are structures with which we try to capture how certain uncertain parameters of the system dynamics evolve in time, that is, to describe a time-dependent

[1] It is intuitive that our predictive ability is better in the near future and it worsens the further into the future we try to predict. What is more, as more data are collected, this will change: the model will improve and adapt to the new data and these distributions will also change.

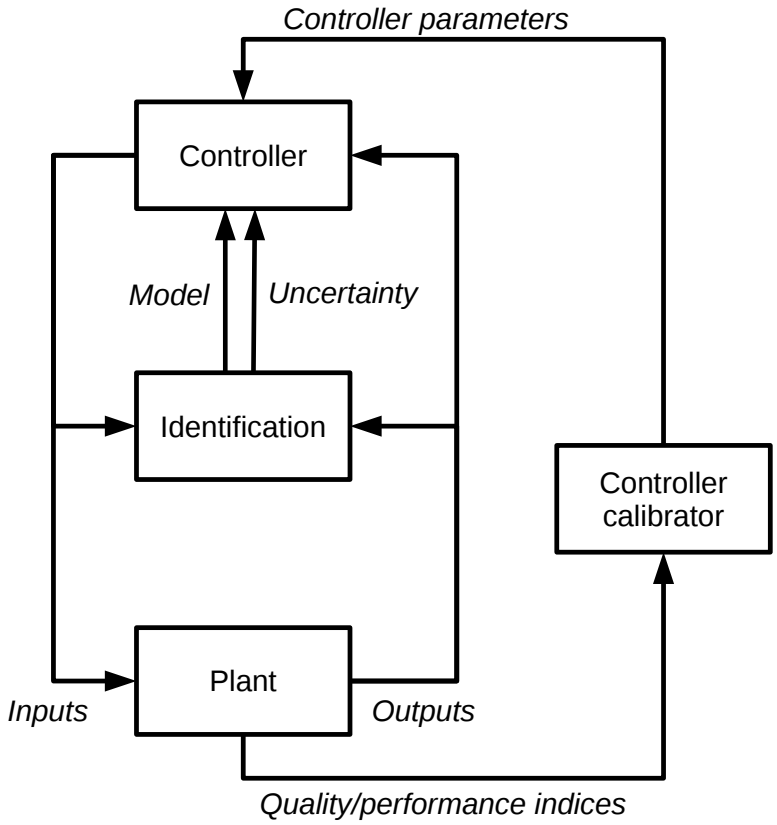

Figure 3.1: Control concept comprising the controlled plant, an identification module which estimates the system dynamics and the uncertainty acting on the system, a controller running stochastic model predictive control or some other advanced control algorithm and a controller calibrator which fine-tunes the controller so as to optimise the closed-loop behaviour of the system.

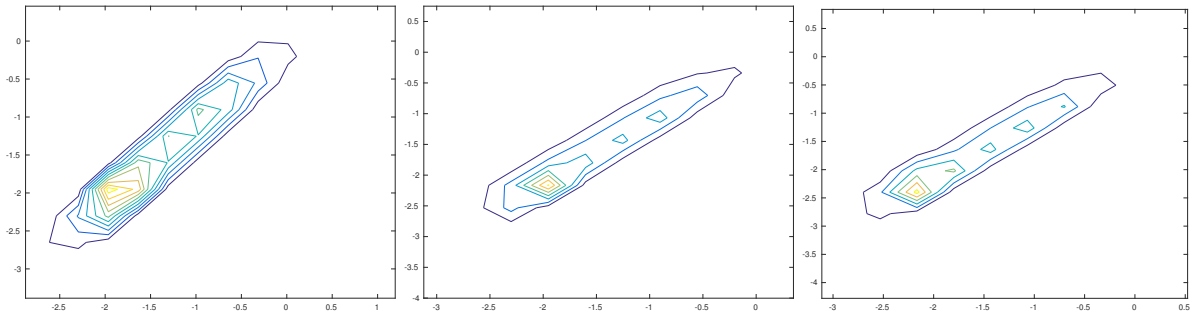

Figure 3.2: Contour plots of frequency of prediction error. The three figures from left to right represent contours of the joint error density function, that is the probability density function of $(\epsilon_{k+j|k},\epsilon_{k+j+1|k})$ with $j \in \{7,8,9\}$, i.e., three last prediction steps. These are the prediction errors of a black-box linear model for the WBF — see Section 4.2.5. As it can be seen, these errors are not well described by a Gaussian distribution.

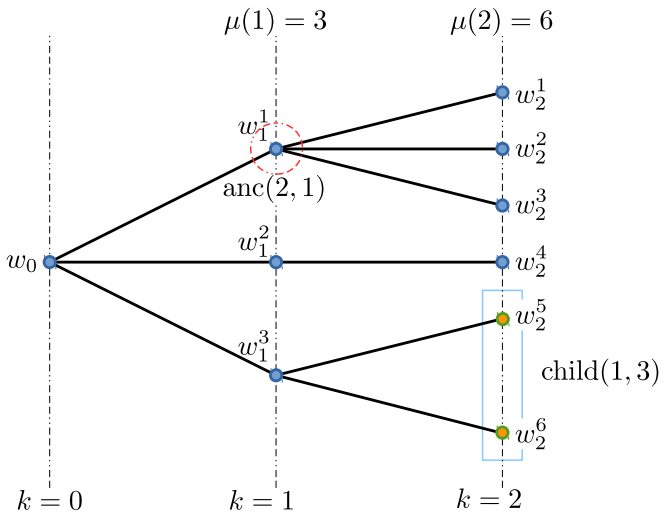

Figure 3.3: Scenario tree describing the uncertainty propagation of $\{w_{k+j|k}\}_j$.

(multi-stage) dynamics inherent to which is the notion of *time causality* on which we will elaborate further later. Let $\{w_j\}_{j=0,\dots,N}$ be a finite-horizon sequence of random vectors w_j. A *scenario fan* is a collection of *equiprobable* realisations of this sequence, i.e., it is a collection $\{w_j^i\}_{j=0,\dots,N;i=1,\dots,M}$. Scenario fans offer a purely data-driven approach for the representation of the uncertainty associated with the random sequence $\{w_k\}_k$ and a way to describe its probability density function. Nonetheless, a good representation of this probability distribution requires a lot of such data which in the context of optimisation and/or when it comes to using such a non-parsimonious representation for (real-time) control purposes, they may deem the problem computationally intractable.

A scenario tree is a *compressed form* of a scenario fan and an alternative representation of the uncertain sequence $\{w_k\}_k$. A scenario tree can be constructed from a scenario fan using some *scenario reduction* algorithm — such algorithms will be discussed in Section 3.1.2. In a scenario tree, realisations are no more equiprobable and a tree-like structure is introduced to capture the temporal distribution of $\{w_k\}_k$ in a parsimonious fashion. A scenario tree is exactly the structure shown in Figure 3.3: The nodes of a scenario tree are partitioned in stages. The (unique) node at stage $k = 0$ is called *root* and the nodes at the last stage are the *leaf nodes* of the tree. We denote the number of leaf nodes by n_s . The number of nodes at stage k is denoted by $\mu(k)$ and the total number of nodes of the tree is denoted by μ. A path connecting the root node with a leaf node is called a *scenario*. Non-leaf nodes define a set of children; at a stage $j \in \mathbb{N}_{[0,N-1]}$ and for $i \in \mathbb{N}_{[1,\mu(j)]}$, the set of children of the i-th node $i \in \mathbb{N}_{[1,\mu(j)]}$ is denoted by $\mathrm{child}(j,i) \subseteq \mathbb{N}_{[1,\mu(j+1)]}$. At a stage $j \in \mathbb{N}_{[1,N]}$, the i-th node $i \in \mathbb{N}_{[1,\mu(j)]}$ is reachable from a single node at stage $k-1$ known as its *ancestor* which is denoted by $\mathrm{anc}(j,i) \in \mathbb{N}_{[1,\mu(j)-1]}$.

The probability of visiting a node i at stage j starting from the root node is denoted by p_j^i. For all $j \in \mathbb{N}_{[0,N]}$ we have that $\sum_{i=1}^{\mu(j)} p_j^i = 1$ and for all $i \in \mathbb{N}_{[1,\mu(k)]}$ it is $\sum_{l \in \mathrm{child}(j,i)} p_{j+1}^l = p_j^i$.

At this point it should be clarified that the structure shown in Figure 3.4 comes with certain assumptions on which parameters are observable, so this tree-like structure is merely indicative

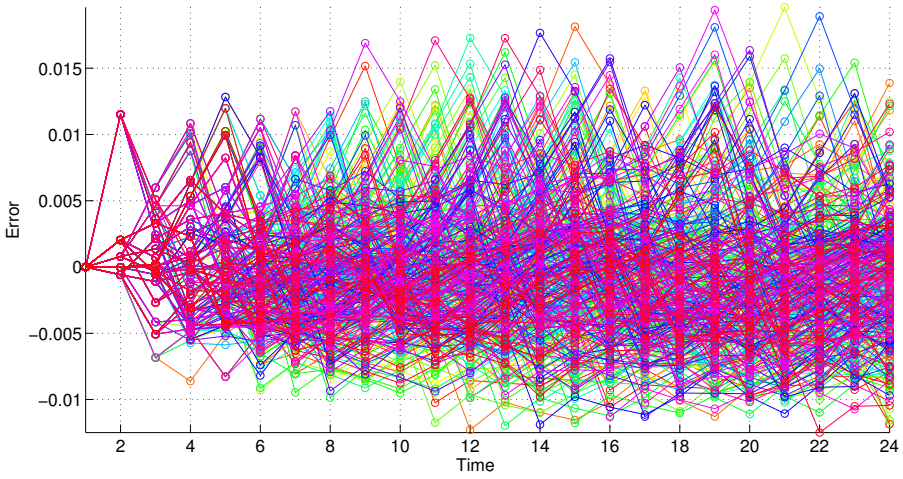

Figure 3.4: Scenario tree describing the uncertainty propagation of $\{w_{k+j|k}\}_j$ at some time instant k constructed using actual modelling error data

as in different applications $w_{k+i|k}$ may or may not be observable at time $k+j$ with $j \geq i$. Oftentimes, at time $k+i \mid k$, information only up to time $k+i-P \mid k$ is available (for some $p \geq 0$). Stochastic model predictive control builds on the very elegant theory of *filtrations* which can concisely describe this *flow of information*, but this is beyond the scope of this report [69–71].

Our discussion so far has been concerned with the scenario tree representation of the uncertain parameter w_k. To the extent that w_k enters the dynamical equation of our system, it induces an uncertain evolution for our state variable x_k. Let us assume that the system state and the disturbance variable w_k are measurable at time k and that the system dynamics is given by a recursion of the form

$$x_{k+j+1|k} = f(x_{k+j|k}, u_{k+j|k}, w_{k+j|k}). \tag{3.1}$$

We should first notice that $x_{k|k} = x_k$ (it is measured) and, likewise, $w_{k+j|k} = w_k$ (which is measured again). The planned control actions $u_{k+j|k}$ will be described hereafter; for now, it should be bore in mind that these are random variables and not single control actions as in conventional deterministic MPC.

To better understand how this uncertainty is reflected onto the (predicted) state sequence, let us take a look at Figure 3.5. Notice that the first move from the current time instant to the next one is deterministic since there is no uncertainty (we know x_k, and w_k). For this transition, the control action u_k is decided using the measured information x_k and w_k, that is $u_{k|k} = \psi_{k|k}(x_k, w_k)$ is a deterministic control action. From that point on, there is multitude of possible outcomes which the controller needs to address by devising a *contingency plan* known as a *control policy*. At stage $k+j \mid k$, the control action $u_{k+j|k}$ is decided as a function of the information that is available up to that future time instant, that is $u_{k+j|k} = \psi_{k+j|k}(x_k, w_k, w_{k+1}, \ldots, w_{k+j})$, where $x_k, w_k, w_{k+1}, \ldots, w_{k+j}$ (i) have already occurred, (ii) have been measured and are (will be) available to the controller, (iii) is exactly the information that is needed to determine the system's state at time $k+j$ and (iv) identify a scenario, so, for different scenarios the controller is allowed to make different decisions.

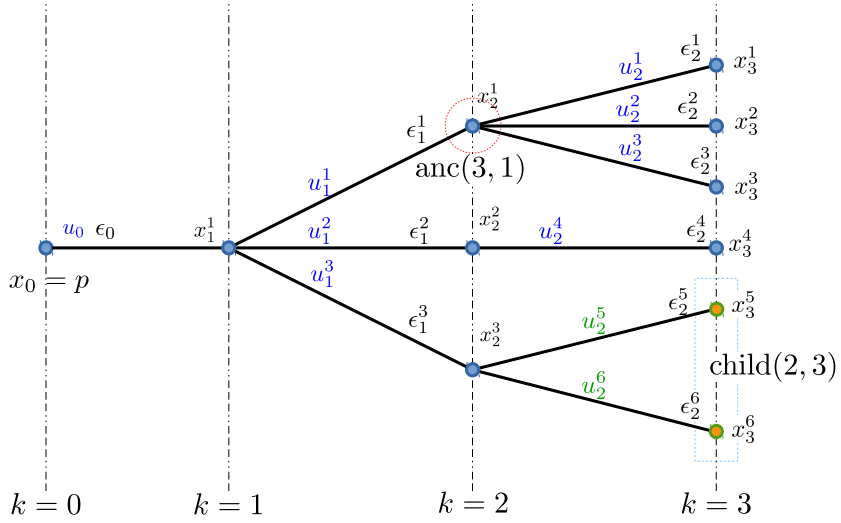

Figure 3.5: Scenario tree describing the uncertainty propagation of $\{x_{k+j|k}\}_j$ at some time instant k.

3.1.2 Scenario reduction

3.1.2.1 What is scenario reduction?

Estimating the joint probability distribution of complex and correlated stochastic processes with time-varying characteristics, such as water demands and/or electricity prices, is a formidable task. The focus of this section is on presenting a systematic procedure for approximating historical data or jointly independent multistage simulation scenarios into scenario trees, a description of uncertainty which is suitable for multistage optimisation. Furthermore, it will be also shown how one can control the complexity of the tree (number of nodes) at the expense of the accuracy of the approximation. This is of great importance for real-time control applications and it actually enable the application of Stochastic MPC for the control of large-scale water networks.

The term *scenario reduction* is to some extent self-explanatory: Scenario reduction is the procedure of simplifying a given scenario tree (in our case a *fan*) whose complexity is prohibitive for its use in stochastic programming. Starting from an initial probability measure with finite support, $\operatorname{supp} P$, scenario reduction consists in determining a probability measure P' whose support is a subset of $\operatorname{supp} P$ and its cardinality is prescribed so that the two probability measures are as close as possible to each other. This distance is quantified by an appropriate metric which we shall define in what follows.

3.1.2.2 The Wasserstein-Kantorovitch Metric

Here we lay the foundations for approximating probability measures with others of reduced complexity and (possibly) more favourable properties. As in every approximation problem, it is of key importance to introduce a metric which measures the distance between elements of the space (in our case, measures). One appropriate notion of distance for probability measures in measur-

able spaces is the *Wasserstein-Kantorovitch* metric. For two probability measures P and Q on a measurable space $(\Omega, \mathfrak{F})$, the Wasserstein-Kantorovitch L_p-metric between P and Q is [72]:

$$W_p(P,Q) \triangleq \inf_{\mu} \left\{ \left(\int_{\Omega\times\Omega} \|x-y\|^p \mathrm{d}\mu(x,y) \right)^{1/p} \middle| \begin{array}{c} \mu \text{ is a probability measure on} \\ (\Omega\times\Omega, \mathfrak{F}\times\mathfrak{F}) \\ \text{with marginals } P \text{ and } Q. \end{array} \right\} \tag{3.2}$$

According to the above definition, the Wasserstein-Kantorovitch metric is the optimal value of an infinite-dimensional optimisation problem which is carried out over a space of measures. Nevertheless, when the sample space is finite, the above metric can be simplified considerably and for given probability measures the computation of their Wasserstein-Kantorovitch distance is computationally tractable and can be done at low complexity. Indeed, let g and h be two random variables living in $\mathbb{R}^n$ which admit values from a finite set and their probability density values are $P(g = g^i) = p_i$ for $i = 1, \ldots, n_g$ and $Q(h = h^i) = q_i$ for $i = 1, \ldots, n_h$. Then, the Wasserstein-Kantorovitch distance between P and Q is

$$W_p(P,Q)^p = \inf_{\substack{\{\eta_{i,j}\}_{i,j} \\ i\in\mathbb{N}_{[1,n_g]} \\ j\in\mathbb{N}_{[1,n_h]}}} \left\{ \begin{array}{l} \displaystyle\sum_{i=1}^{n_g}\sum_{j=1}^{n_h} \eta_{ij}\|g^i - h^j\| \colon \eta_{ij} \geq 0, \\ \displaystyle\sum_{j=1}^{n_h} \eta_{ij} = p_i, \sum_{i=1}^{n_g} \eta_{ij} = q_j \end{array} \right\} \tag{3.3}$$

As we may observe, for discrete measures, the Wasserstein-Kantorovitch metric is the optimal value of a linear program, so it can be computed very efficiently by numerical optimisation software.

The optimal scenario reduction problem reads as follows: Given a discrete probability distribution P with scenarios $\{g_i\}_{i\in I}$, where $I = \mathbb{N}_{[1,n_g]}$, as well as a set of scenarios to be eliminated $J \subset I$, compute a probability distribution Q_J supported by $I \setminus J$ such that its Wasserstein-Kantorovitch distance from P is minimum. The probability distribution Q_J can be computed by solving the following optimisation problem:

$$D_J^\star \triangleq \min \left\{ W_p(P,Q)^p \middle| \sum_{j\notin J} q_j = 1, q_j \geq 0, j \in J \right\}, \tag{3.4}$$

which is again a linear program. By virtue of (3.3), (3.4) is written as follows

$$D_J^\star = \min_{\eta_{i,j},q_j} \sum_{i\in I}\sum_{j\notin J} \eta_{ij}\|g^i - h^j\| \tag{3.5a}$$

subject to:

$$\sum_{i\in I} \eta_{ij} = q_j, \forall j \notin J \tag{3.5b}$$

$$\sum_{j\notin J} \eta_{ij} = p_i, \forall i \in I \tag{3.5c}$$

$$\sum_{j\notin J} q_j = 1, q_j \geq 0, \forall j \notin J \tag{3.5d}$$

$$\eta_{ij} \geq 0, \forall j \notin J, \forall i \in I \tag{3.5e}$$

At this point notice that in (3.5), the constraints (3.5b) and (3.5d) are redundant and can be dropped. This being the case, the above LP can be solved explicitly using duality theory (strong duality holds true for LPs). Its optimal value is given by:

$$D_J^\star = \sum_{i\in J} p_i \min_{j\notin J} \|g^i - h^j\|^p, \tag{3.6}$$

which is obtained at

$$q_j = p_j + \sum_{i\in J\cap\{i|j=\mathcal{J}(i)\}} p_i,\ j \notin J, \tag{3.7}$$

where $\mathcal{J}(i)$ is the *optimal quantisation mapping* defined as

$$\mathcal{J}(i) \triangleq \text{argmin}_{j\notin J}\|g^i - h^j\|^p, i \in J. \tag{3.8}$$

Equation (3.7) is known as the *redistribution rule* following the elimination of a number of scenarios.

Therefore, the optimal "reduced" probability distribution results from adding the probabilities of the deleted scenarios to (one of) the closest remaining scenario(s). This implies directly that the "reduced" probability space comprises non equiprobable scenarios in contrast to the initial one which, in our cases, consists only of scenarios with equal probability.

3.1.2.3 The Backward Elimination Agorithm

Having defined the Wasserstein-Kantorovitch metric between probability measures we shall now describe how this can be the basis for a scenario reduction scheme: the *backward elimination algorithm*. Assume that we are given the cardinality of the set J of scenarios to be eliminated. In order to determine the simplified probability distribution we need to solve the following constrained optimisation problem:

$$\min\{D_J^\star \mid J \subseteq \mathbb{N}_{[1,n_g]}, |J| = n_g - n_h\}. \tag{3.9}$$

Cardinality-constrained optimisation problems can be reformulated as binary optimisation problems which are known to be NP-hard. It is therefore impossible to find a global solution in polynomial time and one must resort to heuristics that offer near-optimal solutions in a computationally tractable way. We proceed as follows: If $|J|= 1$ (one scenario is to be deleted), then the optimisation problem (3.9) assumes the form:

$$\min_{s\in\mathbb{N}_{[1,n_g]}} p_s \min_{j\neq s}\|g^i - h^j\|. \tag{3.10}$$

If the minimum is attained at some $s^\star$, then the corresponding scenario (the one numbered with $s^\star$) is removed and the redistribution rule (3.7) gives the probability distribution function of the reduced measure. The algorithm of backward elimination amounts to applying this same procedure recursively starting from the full set of scenarios $I = \mathbb{N}_{[1,n_g]} \equiv \{1, 2, \ldots, n_g\}$ and deleting/removing scenarios one-by-one in a way that resulting measure is as close as possible (in the Wasserstein-Kantorovitch sense) from the initial one. The algorithm terminates when the produced probability space has the desired number of scenarios.

The algorithm is written as follows: Let P be the initial probability measure described by n_g scenarios and n_h scenarios out of these are to be removed so that the final probability measure counts $n_g - n_h$ scenarios.

Algorithm 5 Backward elimination algorithm

Require: $|J|\equiv n_h$, Initial probability measure P with n_s scenarios.
  $J_0 \leftarrow \varnothing$
  for $i = 1, \ldots, n_g - n_h$ **do**
    $s_i \leftarrow \mathrm{argmin}_{s\notin J_{i-1}} D^\star_{J_{i-1}\cup\{s\}}$
    $J_i \leftarrow J_{i-1} \cup \{s_i\}$
  end for
  Apply the redistribution rule (3.7) to $J_{n_g-n_h}$
  return "Reduced" probability distribution function

The computational complexity of the backward elimination algorithm (Algorithm 5) is:

$$n_h^3 - n_h^2(\frac{3}{2}n_g + \frac{1}{2}) - \frac{3}{2}n_h(n_g + 1) + \frac{1}{2}n_g^3 + \mathcal{O}(n_g^2 \log n_g) + 2n_g^2 + \frac{3}{2}n_g \tag{3.11}$$

The minimisation step in Algorithm 5 is written as follows:

$$s_i = \mathrm{argmin}_{s\notin J_{i-1}} \sum_{j\in J_{i-1}\cup\{s\}} p_j \min_{\ell\notin J_{i-1}\cup\{s\}} \|g^i - h^\ell\|^p \tag{3.12}$$

3.1.2.4 The Forward Selection Agorithm

The forward selection algorithm, as its name suggests, proceeds in the opposite direction: It commences with $J = \mathbb{N}_{[1,S]}$, where S is the number of scenarios to be eliminated and removes indices from J (one by one) so that the Wasserstein-Kantorowitch distance between the two probability measures is minimised.

Algorithm 6 Forward selection algorithm

Require: $|J| \equiv n_h$, Initial probability measure P with n_s scenarios.
  $J_0 \leftarrow \mathbb{N}_{[1,n_g]}$
  for $i = 1, \ldots, n_h$ **do**
    $s_i \leftarrow \text{argmin}_{s \in J_{i-1}} D^{\star}_{J_{i-1} \setminus \{s\}}$
    $J_i \leftarrow J_{i-1} \setminus \{s_i\}$
  end for
  Apply the redistribution rule (3.7) to J_{n_h}
  return "Reduced" probability distribution function

Algorithm 6 is the forward selection algorithm whose complexity is:

$$\frac{2}{3}n_h^3 - n_h^2(2n_s + 1) + n_h(2n_s^2 + 2n_s + \frac{1}{3}). \tag{3.13}$$

The minimisation step in Algorithm 6 can be concisely written as

$$s_i = \text{argmin}_{s \in J_{i-1}} \sum_{j \in J_{i-1} \setminus \{s\}} p_j \min_{\ell \in J_{i-1} \setminus \{s\}} \|g^i - h^\ell\|^p. \tag{3.14}$$

3.1.2.5 Application to Scenarion Tree Generation

When it comes to scenario trees, the causality that is dictated by the arrow of time has to be taken into consideration. This is the reason why the reduction proceeds stage after stage transforming an initial *fan* of independent scenarios to a structured tree. A scenario fan is a set of independent (i.e., they share no nodes) mutlistage scenarios. The initial fan of scenarios can be generated from past demand or price data. The proposed approach is purely data-driven and model-free and no assumptions needs to be taken in regard to the form of the distribution function from which these scenarios have been drawn (e.g., Normal, Weibull etc).

Various scenario tree generation methods have been proposed in the literature for multi-stage problems [72–81]. One way to perform scenario tree generation is to start from the root node of the scenario fan and apply scenario reduction (e.g. backward elimination or forward selection) to the scenarios up to the first stage. This way, clusters of scenarios corresponding to the kept ones are created and the procedure is repeated for each of the cluster by increasing the horizon by one, and so on. This algorithm is known as *forward selection* (not to be confused with the forward selection algorithm as in Algorithm 6 for scenario reduction).

Let us consider a set of independent scenarios $\{\xi^s\}_{s \in I}$, where $I = \mathbb{N}_{[1,\ldots,n_g]}$, where each ξ^s corresponds to a sequence of ξ^s_k for all $k = 0, \ldots, N$, that is $\xi^s = (\xi^s_0, \xi^s_1, \ldots, \xi^s_N)$ for all $s \in I$ while all scenarios share the same root node, i.e., $\xi^s_0 = \xi_0$ for all $s \in I$. Let us denote by $\xi^s_{[1,k]}$ the truncation of the sequence $(\xi^s_0, \xi^s_1, \ldots, \xi^s_N)$ up to its k-th element, that is $\xi^s_{[1,k]} = (\xi^s_0, \xi^s_1, \ldots, \xi^s_k)$. At every time $k \in \mathbb{N}_{[0,N]}$ the forward selection algorithm for scenario tree generation creates progressively finer clusters by applying a scenario reduction method to the truncated sequence $\xi_{[1,k+1]}$ for each of the previously created clusters. This procedure is presented in Algorithm 7.

Algorithm 7 Forward selection algorithm for scenario tree generation

Require: Scenario fan $\{\xi^s\}_{s\in I}$ with $I = \{1, 2, \ldots, n_g\}$, Splitting factor at every stage
  $C_0^1 \leftarrow I$, $\mathcal{C}_0 \leftarrow \{C_0^1\}$
  for $k = 0, \ldots, N-1$ **do**
    for $\ell = 0, \ldots, |\mathcal{C}_k|$ **do**
      Partition $C_k \leftarrow I_{k+1}^\ell \cup J_{k+1}^\ell$ by applying scenario reduction to $\{\xi^s_{[1,k+1]}; s \in C_k^\ell\}$
      for $s \in I_{k+1}^\ell$ **do**
        $C_{k+1}^s \leftarrow s \cup (\mathbb{J}_{k+1}^\ell)^{-1}(s)$
      end for
      $\mathcal{C}_{k+1} \leftarrow \{C_{k+1}^s\}_{s\in I_{k+1}^\ell}$
    end for
  end for
  return Generated scenario tree

3.2 Scenario-based model predictive control

3.2.1 Motivation

Model predictive control (MPC), often referred to as *receding horizon control*, is a class of advanced control techniques where at every time instant k an open-loop control problem is solved along a prediction horizon using a system model of the actual process where a performance index is minimized. The first control action of this control problem is applied to the system. MPC has gained great popularity for its ability to handle systems with multiple inputs and outputs and with input, output and/or state constraints.

MPC computes a sequence of (predicted) control actions based on a predictive model of the underlying system. Inevitably, however, every model comes with uncertainty, either due to inaccurate assumption, or due to external disturbances that cannot be cast by the model. Besides, very accurate models are often too complex to be incorporated in an optimisation problem. Uncertainty, is usually addressed in the literature in the context of *worst-case robust MPC*, where the controller tries to accommodate the worst case scenario assuming certain bounds for the uncertainty. This pessimistic approach can both degrade the quality of the closed-loop performance of the controlled system and shrink its domain of attraction which practically means that stability will be guaranteed only under very stringent assumptions.

Nevertheless, worst-case scenarios are altogether rather unlikely to occur in practice. To give an example, if the output measurements are transferred over a wireless network where it is likely that they are not transmitted, and assuming that this happens with probability 0.001%, a worst-case scheme would assume that measurements are never transmitted! An alternative approach is to introduce to the control problem formulation the notion of *likelihood*, or *probability* of such scenarios actually occurring. In *stochastic* model predictive control, the controller aims at minimising the expected value of the predicted cost along a prediction horizon rather than its worst-case maximum value.

Stochastic MPC is the state of the art in control; we will now use the concept of scenario trees we presented in Section 3.1.1 which can be constructed from process data as detailed in Section 3.1.2 to formulate a control problem which offers the robustness of MPC in a non-pessimistic fashion.

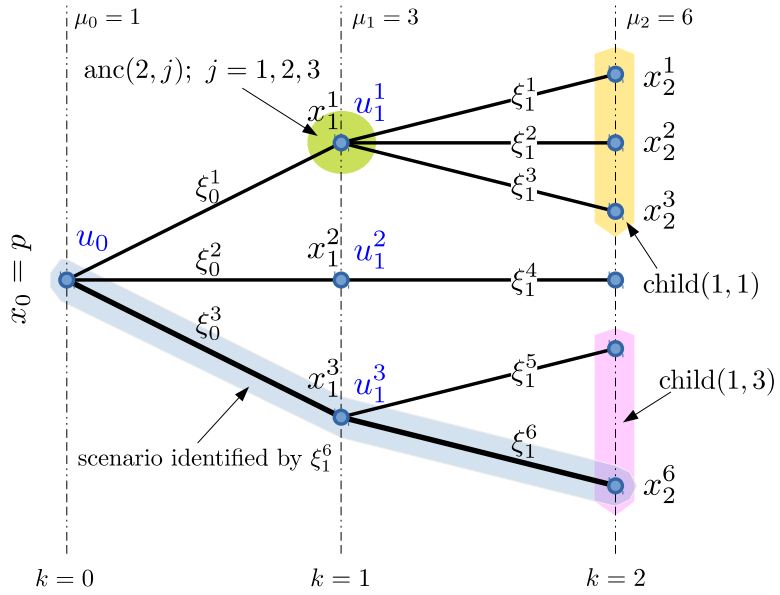

Figure 3.6: Decision making in scenario-based model predictive control.

3.2.2 Problem formulation

Scenario-based model predictive control has become popular for its ability deliver control actions with foresight under uncertainty and has found applications in control of power dispatch [82], control of macroeconomic systems [69], control of supply chains [83] and many another.

In stochastic model predictive control, future control actions $u_{k+j|k}$ are themselves random variables and not single control actions. Instead, in SMPC the goal is to decide a sequence of control actions which depend on the available information up to stage $k + j \mid k$, that is, $u_{k+j|k}$ is a function

$$u_{k+j|k} = \psi_{k+j|k}(x_k, w_k, w_{k+1}, \ldots, w_{k+j}). \tag{3.15}$$

The decision making along the scenarios of a tree is illustrated in Figure 3.6.

Following the discussion in Section 3.1.1, the system state at stage $k+j$ lives in a probability space comprising the values x_j^i for $i \in \mathbb{N}_{[1,\mu(k-1)]}$ and at every *node* of the scenario tree in Figure 3.3 the controller decides a control action u_k^i. Then, across the nodes of the scenario tree, the system dynamics is described by the recursion

$$x_{k+j+1|k}^l = f(x_{k+j|k}^i, u_{k+j|k}^j, w_{k+j|k}^l), \tag{3.16}$$

for $j \in \mathbb{N}_{[0,N-1]}$, $i \in \mathbb{N}_{[1,\mu(j)]}$ and $l \in \mathrm{child}(j, i)$, or alternatively

$$x_{k+j+1|k}^i = f(x_{k+j|k}^{\mathrm{anc}(j+1,i)}, u_{k+j|k}^i, w_{k+j|k}^i). \tag{3.17}$$

We may now formulate the following stochastic model predictive control problem with decision variables $\pi = \{u_{k+j|k}, x_{k+j+1|k}\}_{j \in \mathbb{N}_{[0,N-1]}}$. The model predictive control problem can be

written as

$$V^\star(p, \hat{\mathbf{w}}_k, k) = \min_\pi \mathbb{E}V(\pi, p, k), \tag{3.18}$$

where the parameter p stands for the current state measurement and $\hat{\mathbf{w}}_k$ is a sequence of the predicted values of the random variable w_k, that is $\hat{\mathbf{w}}_k = (\hat{w}_{k|k}, \hat{w}_{k+1|k}, \dots, \hat{w}_{k+N-1|k})$ with $\hat{w}_{k|k} = w_k$ (since $\hat{w}_k$ is observed).

The cost function of the problem as the following random variable which is defined as a stage-wise sum

$$V(\pi, p, k) = \sum_{j=0}^{N-1} \ell(x_{k+j|k}, u_{k+j|k}, k+j), \tag{3.19}$$

subject to the system constraints, i.e., the system dynamics as in (3.16) and (3.17) and possibly additional state and/or input constraints. The *stage cost function* ℓ reflects the control objectives and, possibly, soft constraints, i.e., constraints of the form $x_{k+j|k} \in C$ which are — instead of being imposed as *hard constraints* — replaced by a penalty function of the form

$$\ell_s(x, k) = \gamma_k \operatorname{dist}(x, C). \tag{3.20}$$

In (3.19) $x_{k+j|k}$ are computed according to (3.16) (or (3.17)) with $x_{k|k} = p$.

The expectation $\mathbb{E}V$ can be written in a computationally favourable form taking into account the scenario tree structure discussed above; it is

$$\mathbb{E}V(\pi, p, k) = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} p_j^i \ell(x_{k+j|k}, u_{k+j|k}, k+j). \tag{3.21}$$

Such problems can often be formulated and solved as quadratic programmes, although *ad hoc* algorithms have been proposed which can solve them efficiently and are suitable for real time implementations. For particular large scale problems — where their large scale often accrues from the large number of scenarios that need to be taken into account to adequately describe the probability distribution of $\{w_{k+j|k}\}_j$ — can be efficiently solved on GPUs as in [84].

The scenario-based stochastic MPC formulation has a number of advantages over conventional certainty-equivalent and robust worst-case MPC formulations. First, it does not assume accurate knowledge of the system dynamics, so it becomes resilient to modelling errors. Second, it does not optimise for worst-case outcomes of modelling uncertainty — which in practice are very unlikely to occur — so it is a lot less conservative. Additionally, it offers great flexibility in choosing the control actions as in (3.15). Finally, contrary to SMPC formulations such as [85], the scenario-based approach does not require the assumption of normality of uncertain parameters which is often not realistic and may lead to erroneous results.

Chapter 4

Case study: Walking beam furnace

In this chapter we address the case study of WP7: the walking beam furnace. First, having received data from MEFOS, we set up a MySQL database and we preprocessed and indexed the data. We applied various system identification methodologies and we described their uncertainty using the scenario approach. Finally, we formulated a stochastic model predictive control problem and introduced KPIs which will allow us to evaluate the closed-loop performance of the controlled system.

4.1 Introduction and problem statement

4.1.1 Data storage and retrieval

The data are stored in a MySQL database which is hosted at `147.102.82.32` and is accessible by a user with username `monty`. Access to the DB *may* be provided only under a non-disclosure aggreement with MEFOS and will be given to any of the DISIRE partners upon request.

```
# Connection to the MySQL database
mysql -u monty -p -h 147.102.82.32
```

Data are stored in tabular form in a MyISAM table called `WBFData` which has the following structure

```
+----------+---------------+
| Field    | Type          |
+----------+---------------+
| id       | int(11)       |
| property | varchar(2000) |
| time     | timestamp     |
| value    | double        |
+----------+---------------+
```

These are the original data provided by MEFOS and are not subject to any preprocessing. Each measurement has a unique identifier (ID) which is of type `int(11)`. The reported property is identified by the column `property` to which there corresponds a value stored under the key `value` (of type `double`). Integer-valued properties are also stored as `double` for convenience

(e.g., the door status). A timestamp is assigned to each property-value pair which is the timestamp of its observation. Once a property-value pair is first recorded in the database, no other value will be recorded for the same property unless there has been a change in the value. This is a very simple method of data compression, but makes it necessary that the data are preprocessed to be useful for modelling purposes.

Overall, 223 properties are recorded and stored in this table which can be listed using the command

```
select distinct(property) from WBFData;
```

The most important properties for our analysis are the zone temperatures, the fuel supplies at each zone, the convection air flow, the pressure inside the furnace, the status of the load and unload hatch-doors. In subsequent sections we will list the exact variables we use for model.

For convenience we have created a table called `WBFDataShort` in which we store only those properties that are relevant for modelling. Using the raw data from table `WBFData` we subsampled (using a moving average) and we store the measurements in talbe `WBFDataShort` with a sampling time of $10s$. The original data were unevenly sampled and typical sampling times were about $1s$ which is too small for our purposes.

In `WBFDataShort` we have stored the following properties

1. `WBF_Z0X_OilControl_FIC:HSI.[MV/WSP/POut]`
2. `WBF_Z0X_OilOutput:HSI.MV`
3. `WBF_MainExhaust_ExhaustTemperature:HSI.[MV/WSP/POut]`
4. `WBF_MainExhaust_ExhaustFlow_FIC:HSI.[MV/WSP/POut]`
5. `WBF_Z0X_CombAir_FIC:HSI.[MV/WSP/POut]`
6. `WBF_Z0X_ZoneTemp:HSI.[MV/WSP/POut]`
7. `SU_IML_GB[5/6/29/30]_SGU:HSI.Value`
8. `WBF__PC027:HSI.[MV/WSP/POut]`
9. `WBF_ColdCAirOutputHSI:MV`

Notice that certain values have one of the three suffixes: `MV`, `WSP` or `POut`. Variables ending in `MV` denote a measure value (using a sensor). The ones ending in `WSP` correspong to set-points which are provided to the controllers and `POut` are the signals produced by the corresponding controllers.

4.1.2 Getting in touch with the data

The properties of interest for the walking beam furnace are the following

- Oil Flow (`WBF_Z01_OilControl_FIC_val`) – supply rate of oil to the burners
- Atomized air – air used to atomize the oil (highly correlated to oil-flow)
- Combustion Air – air flow to the furnace
- Pressure – pressure inside the furnace
- Oxygen $\%$ – percentage of O_2 inside the furnace
- Status of loading hatch-door
- Status of unloading hatch-door

Variable	*Description*	*Limits*	*Role*
WBF_Z0[1-3]_.. OilControl_FIC_val	Supply rate of oil to the burners. There is one burner at each zone	$0 \sim 50kg/h$	MV
WBF_Z0[1-3]_.. AtomAir_FIC:HSI.MV	Atomisation air used to dilute the oil.	–	MV
WBF_Z0[1-3]_.. ZoneTemp:HSI.MV	Zone temperature. At each zone there are two thermocouples; one at the wall of the zone and one at its ceiling.	During typical operating the zone temperatures are roughly between $500^{\circ}C$ and $1200^{\circ}C$	T
WBF_MainExhaust_.. ExhaustFlow_FIC:HSI.MV	Exhaust flow	–	M
SU_IML_GB[5/6/29/30]_.. SGU:HSI.Value	Hatch-door status	Open or Closed or Ajar	M/MV
WBF_PC027:HSI.MV	Pressure	$-50 \sim 50Pa$	M
WBF_Z0[1-3]_.. CombAir_FIC:HSI.MV	Combustion air at each zone.		M
WBF_Z0[1-3]_.. O2_QIC:HSI.MV	Oxygen concentration (%). Measured using a ABB-ZDT sensor.	$0 \sim 25\%$	M
WBF_ColdCAir.. OutputHSI:MV	Recirculation of cold air. Recirculation was switched off during the trials. (%).	–	M

Table 4.1: Variables of the WBF. T: target, MV: Manipulated variable, M: Measured variable (e.g., output).

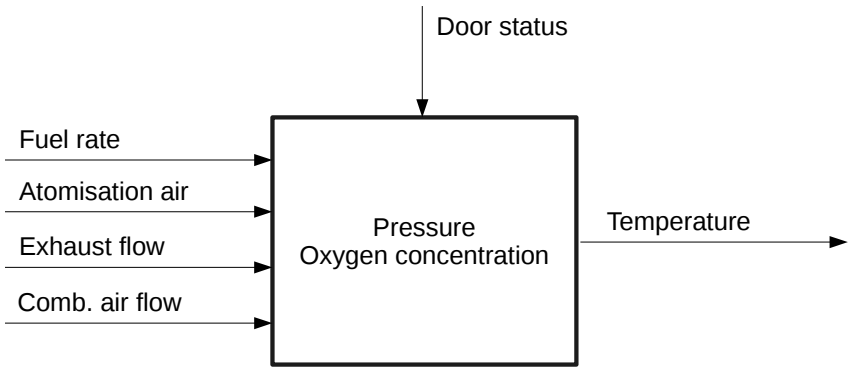

Figure 4.1: Inputs (fuel rate, atomisation air, exhaust flow (controlled by the exhaust damper) and combustion air supply rate), outputs (temperatures) and disturbances (hatch-door status) of the walking beam furnace.

A concise overview of the system variables is provided in Table 4.1. As we may observe both in Figures 4.2 and 4.3, the system variables are highly correlated with one another and, additionally, the fact that the data have been gathered from closed-loop simulations deems the task of identifying a system model rather hard. Nonetheless, as we will show in Section 4.2 it is possible to predict the system trajectories along a horizon of $100s$.

4.1.3 Data preprocessing

Prior to the modelling phase, the data have been preprocessed to eliminate missing values. The original sampling time was approximately $1s$ and it was non-constant. In particular, a new data entry was registered in the database only if it was different from the previous measurement, so the rate of update was not constant. We re-sampled the recorded time-series so that the new sampling time is $10s$; for that purpose we used a moving average filter which served also as a

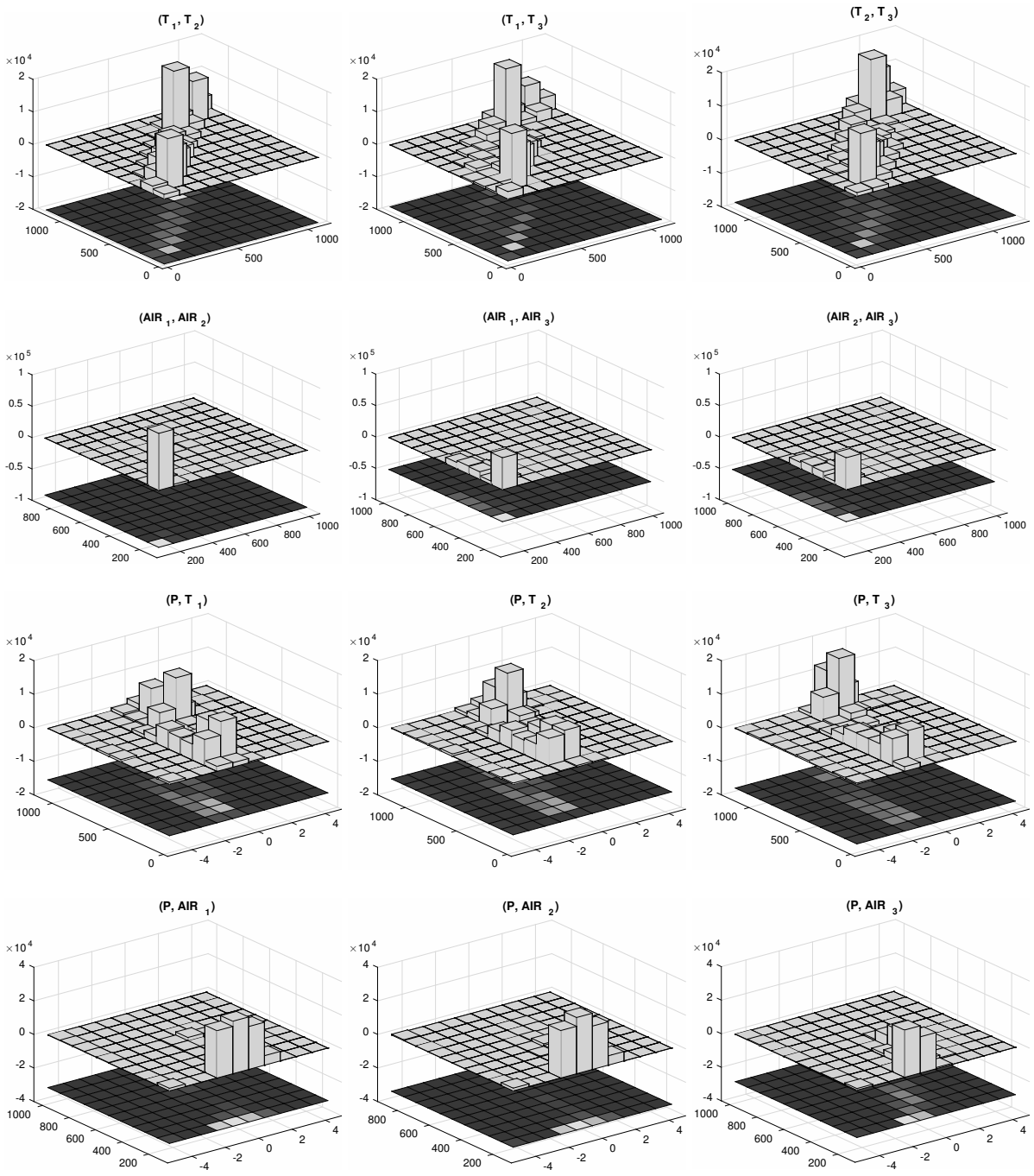

Figure 4.2: Pairwise distributions of the states and input variables of the system. The more the ridge resembles to a straight line, the less the information that the respective pair of variables brings about and the more linearly correlated they are.

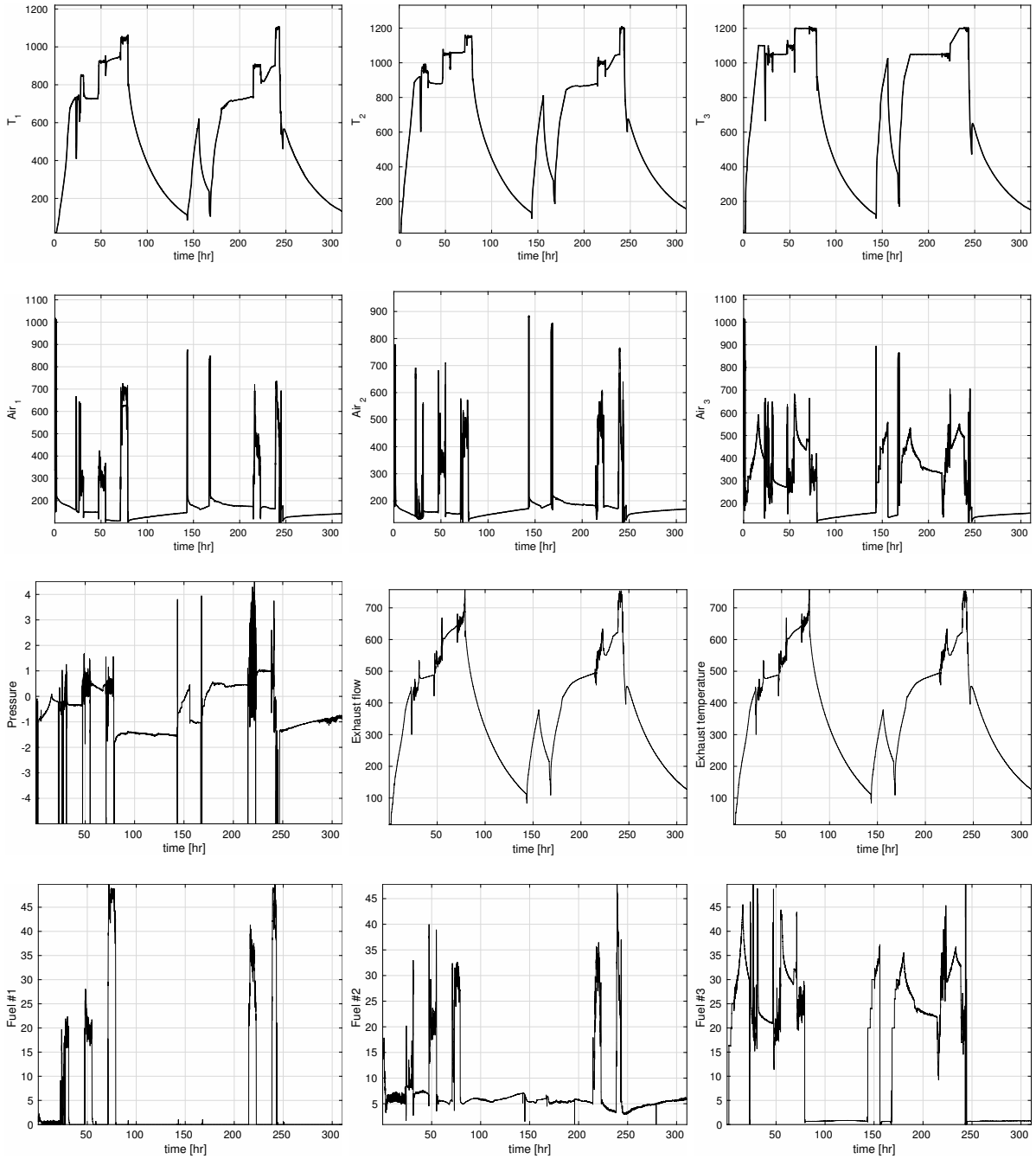

Figure 4.3: Time profiles of the system variables. For each variable we show the historical values which correspond to the actual measurement (fine line) and the ones which correspond to the set-points (thick line). As we will show later, some jitter is observed when the doors of the furnace are open; this acts as a disturbance on the system.

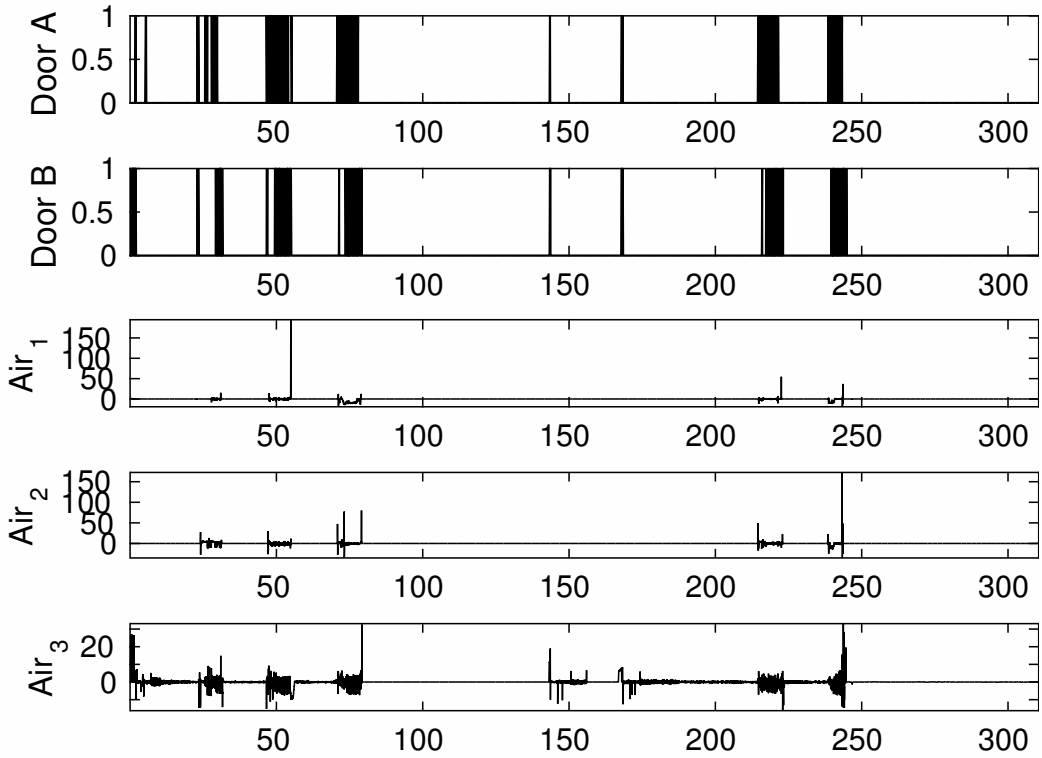

Figure 4.4: Doors' status (open/closed) plotted along with the controlled combustion air flow. The third, fourth and fifth plots show the set-point deviation (error) in $\%$ of the combustion air. Notice that the opening of the hatch-doors causes the set-point deviation to run rampant.

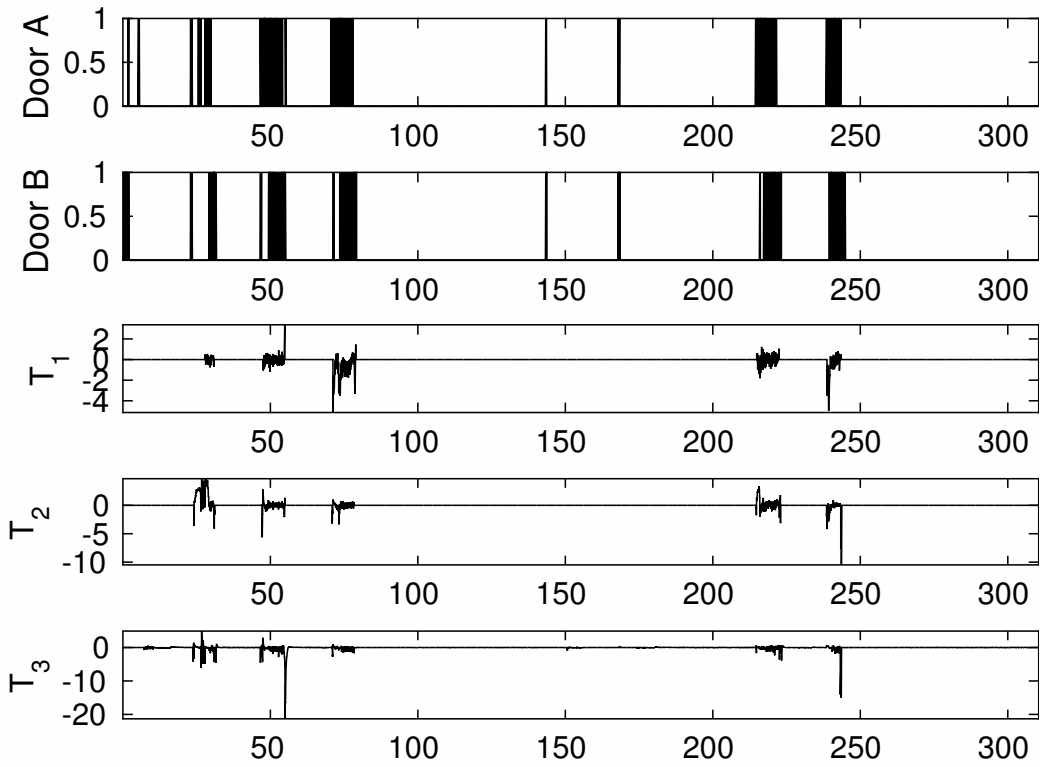

Figure 4.5: Doors' status (open/closed) plotted along with the controlled zone temperature. The third, fourth and fifth plots show the set-point deviation (error) in $\%$ of the combustion air. Notice the large deviations when the doors are open which may be as high as $\pm 20\%$.

noise filter.

4.2 From data to dynamical process models

In this section we elaborate on how data can be used to construct dynamical process models, often exploiting prior information about the process and in other cases pursuing a totally black-box approach where no prior knowledge about the process dynamics is assumed to be known. First, in Section 4.2.1 we lay the foundations for our subsequent analysis stating how we evaluate the predictive ability of a dynamical model using the modern approach of *risk measures* (which, to the best of our knowledge, is applied in model selection for the first time here). Then, in Section 4.2.3 we venture to model the process using first principles and heat balance equations. In Section 4.2.4 we use subspace identification methods to identify linear time-invariant models for the walking beam furnace which are simple in structure and of low complexity and lead to a decent predictive ability. Then, in Sections 4.2.5 and 4.2.6 we develop linear auto-regressive models and sparse models which lead to a good predictive ability. In a data-rich setting, as in normal industrial operation, the abundance of data from the system can be used to iteratively refine the obtained model and detect changes and time-varying behaviours; although, here, we do not have this much of data available, we developed the computational framework for adaptive model learning as we discuss in Section 4.2.7.

Last, we provide a comparison of the merits of all these models and analyse how the prediction uncertainty propagates in time — see Section 4.3. We put together all these pieces of the puzzle to state the stochastic model predictive control problem in Section 4.4.

4.2.1 Evaluation of predictive ability

Predictive ability. Models, regardless of their form and structure, need to be appraised for their predictive ability. For this purpose we introduced and used a new performance indicator which we present in this section. At time k, using past information (past observations of the system state and input values) and given the current control action u_k, the model predicts a value $\hat{y}_{k+1|k}$. Recursively, and given a sequence of future (predicted) control actions $u_{k+1}, u_{k+2}, \ldots, u_{k+N-1}$, the model is used to predict $\hat{y}_{k+i|k}$ for $i = 2, \ldots, N$. Once the actual output realisations have occurred, that is, at the end of the horizon, we may compute the errors

$$\epsilon_{k+j|k} = \hat{y}_{k+j|k} - y_{k+j} \tag{4.1}$$

Thus, at every k we may create a vector $\epsilon_k := (\epsilon_{k+1|k}, \ldots, \epsilon_{k+N|k}) \in \mathbb{R}^N$ which is, essentially, a random variable. We quantify the predictive ability of the model *at time* k using the *predicted mean square error* defined as

$$\mathrm{PMSE}_k = \frac{1}{N} \sum_{j=1}^{N} \|\epsilon_{k+j|k}\|^2. \tag{4.2}$$

We then define the *predicted root mean square error at time* k which we denote by PRMSE_k as the square root of PMSE_k,

$$\mathrm{PRMSE}_k = \sqrt{\mathrm{PMSE}_k}. \tag{4.3}$$

This is a random variable. It should be clear that performing a single-shoot prediction and computing PRMSE_k at a particular k is not indicative of the model's predictive ability at other time instants k. But computing the expected value of PRMSE_k is again not really indicative because we will be disregarding all other higher-order moments of PRMSE_k.

Quantifying risk. In order to extract a meaningful characteristic value out of the random variable PRMSE_k we need to employ an appropriate *risk measure*. In particular, since we need to minimize the prediction error – rather than to maximise it – we are looking for a convex risk measure. The theory of risk measures has been well established the last decade and certain well-posedness axioms have been postulated.

Rigorously speaking, a risk measure is a mapping from a space of measurable integrable functions $\mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$ over a probability space $(\Omega, \mathcal{F}, \mathrm{P})$ to the set of extended-real numbers $\bar{\mathbb{R}} = \mathbb{R} \sup\{\pm\infty\}$ which is proper (i.e., for all random variables $Z \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$, a proper risk measure $\rho : \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P}) \to \bar{\mathbb{R}}$ satisfies $\rho(Z) > -\infty$ and $\mathrm{dom}\, \rho = \{Z : \rho(Z) < +\infty\} \neq \varnothing$. A risk measure is called *coherent* if it satisfies the four *coherency axioms* which are stated as follows

1. Subadditivity: $\rho(Z + Y) \leq \rho(Z) + \rho(Y)$ for all $Z, Y \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$
2. Positive homogeneity: $\rho(\alpha Z) \leq \alpha \rho(Z)$ for all $Z \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$ and $\alpha \geq 0$
3. Monotonicity: $\rho(Z) \leq \rho(Y)$ whenever $Z \leq Y$ for $Z, Y \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$ and the relation $Z \leq Y$ is meant in the almost-sure sense, i.e., $\mathrm{P}[\{\omega : Z(\omega) > Y(\omega)\}] = 0$
4. Translation invariance: $\rho(Z + c) = c + \rho(Z)$ for al $Z \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathrm{P})$

Notice that because of the *subadditivity* and *positive homogeneity* requirements, coherent risk measures are convex.

A risk measure which complies to these axioms is termed a *coherent* risk measure. The most popular risk measure which enjoys a series of favourable properties is the *average value-at-risk* of an (integrable) random variable Z which is defined as

$$\mathrm{AV@R}_\alpha[Z] = \inf_{t \in \mathbb{R}} \{t + \alpha^{-1} \mathbb{E}[Z - t]_+\}, \tag{4.4}$$

where $[X]_+ = \max\{0, X\}$ and α is the significance level at which $\mathrm{AV@R}_\alpha$ is estimated.

In order to understand the meaning and practical significance of this operator, we need first to define the *value-at-risk* of a real-valued random variable Z. This is

$$\mathrm{V@R}_\alpha[Z] = F_Z^{-1}(1 - \alpha), \tag{4.5}$$

that is, it a value above which there is probability at least α that Z is, and in fact it is the smallest such value. Formally,

$$\mathrm{V@R}_\alpha[Z] = \inf\{z : \mathrm{P}[Z \leq z] \geq \alpha\}. \tag{4.6}$$

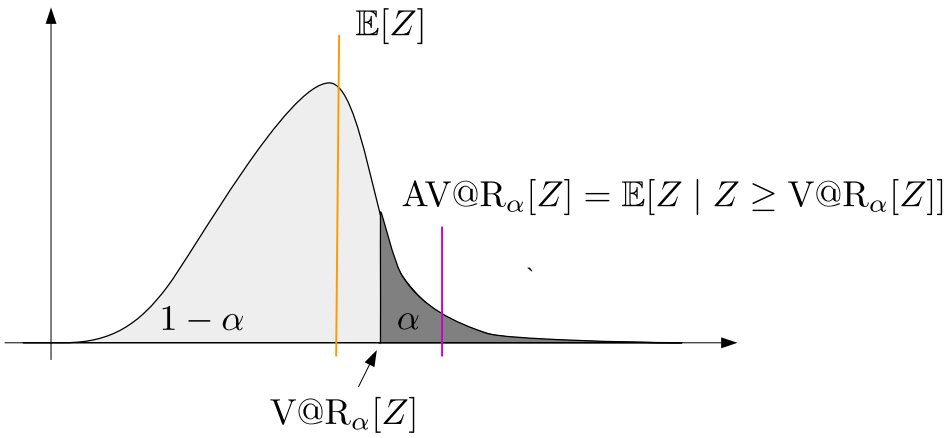

Figure 4.6: Understanding $\mathrm{AV@R}_\alpha$: $\mathrm{V@R}_\alpha$ splits the probability density function in two parts where the right tail has probability α. $\mathrm{AV@R}_\alpha$ is the probability of the part of the PDF which is above $\mathrm{V@R}_\alpha$.

It can be easily shown that $\mathrm{AV@R}_\alpha[Z]$ is the expectation of Z conditioned by $Z \geq \mathrm{V@R}_\alpha[Z]$, i.e.,

$$\mathrm{AV@R}_\alpha[Z] = \mathbb{E}[Z \mid Z \geq \mathrm{V@R}_\alpha[Z]]. \tag{4.7}$$

We need to highlight that the minimisation of $\mathrm{AV@R}_\alpha[Z]$ leads to different choices than the minimisation of $\mathbb{E}[Z]$. It should be neither assumed that $\mathrm{AV@R}_\alpha[Z_1] \leq \mathrm{AV@R}_\alpha[Z_2]$ implies $\mathbb{E}[Z_1] < \mathbb{E}[Z_2]$, nor the converse.

In case the underlying probability space is $\Omega = \{\omega_1, \ldots, \omega_K\}$ (as in our approach) and $p_i := \mathrm{P}[\omega = \omega_i]$ then $\mathrm{AV@R}_\alpha$ is computed via the following LP

$$\mathrm{AV@R}_\alpha[Z] = \max_{Y \in \mathbb{R}^K} \underbrace{\sum p_i Y_i Z_i}_{\langle Y, Z \rangle} \tag{4.8a}$$

$$\text{s.t.} \quad \sum p_i Y_i = 1 \tag{4.8b}$$

$$0 \leq Y_i \leq \alpha^{-1} \tag{4.8c}$$

Equivalently, $\mathrm{AV@R}_\alpha$ can be computed by

$$\mathrm{AV@R}_\alpha[Z] = \max_{\mu \in \mathbb{R}^K} \underbrace{\sum_i \mu_i Z_i}_{\mathbb{E}_\mu[Z]} \tag{4.9a}$$

$$\text{s.t.} \quad \sum \mu_i = 1 \tag{4.9b}$$

$$0 \leq p_i^{-1} \mu_i \leq \alpha^{-1} \tag{4.9c}$$

We may also compute $\mathrm{AV@R}_\alpha[Z]$ using its definition as follows

$$\mathrm{AV@R}_\alpha[Z] = \min_{t\in\mathbb{R}} \left\{ t + \alpha^{-1}\mathbb{E}[Z-t]_+ \right\} \tag{4.10a}$$

$$= \min_{t\in\mathbb{R}} \left\{ t + \alpha^{-1} \sum_i p_i [Z_i - t]_+ \right\} \tag{4.10b}$$

$$= \min_{\substack{t\in\mathbb{R},\xi\in\mathbb{R}^n \\ \xi\geq 0, Z_i - t \leq \xi_i}} \left\{ t + \alpha^{-1} \sum_i p_i \xi_i \right\} \tag{4.10c}$$

We now give two equivalent implementations of $\mathrm{AV@R}_\alpha$ in MATLAB for finite-value risk measures. The first one relies on the dual representation of $\mathrm{AV@R}_\alpha$ and returns a subgradient of $\mathrm{AV@R}_\alpha$ at Z.

```
function [a, mu] = avar(Z, p, alpha)
% Z     : Discrete values of RV
% p     : probabilities
% a     : Average value at risk (level alpha)
% mu    : A subgradient of AVAR_alpha at Z
[mu, a, exitflag] = linprog(-Z',[],[], ...
   ones(1, n), 1, zeros(n,1), p/alpha);
assert(exitflag == 1, 'numerical problems');
a = -a;
```

```
function a = avar(Z, p, alpha)
% Computation using the definition
% This code does not return a subgradient
n = length(Z);
f = [1 p/alpha];
H = -[ones(n,1) eye(n)];
[~, a]=linprog(f, H, -Z',[],[],[-Inf;zeros(n,1)]);
```

Quantifying the predictive ability. In our analysis, in order to select a reliable dynamical model, we assess its predictive ability on a set of data which has not been used for training; this we call the *test dataset* and thereon we compute $\mathrm{AV@R}_\alpha[\mathrm{PRMSE}_k]$. Note that the models we produce may not be suitable for long-term open-loop predictions, they are suitable for control applications where a predictive ability of a few time instants is required, notwithstanding, but the availability of feedback obviates the need for very accurate models.

4.2.2 Feature selection for dimensionality reduction

One of the problems faced by machine learning and data driven approaches is the well known "curse of dimensionality". Most methods cannot handle a large input space, leading to reduced generalization performance as well as very large training times. Therefore before the training of the model, a preprocessing step takes place that aims at reducing the dimensionality of the data set that is fed to the learning model. There are two major families of methods that can be used for that. The first one is feature selection, which tries to pick the most informative/relevant inputs

(also known as attributes of features for the case of classification and regressors in the case of regression) and the second one is input transformation/projection to lower dimensional space through manipulation of the original input vector. In the next subsection we describe some of the most prominent methods for both approaches.

Feature selection is a search problem to find a subset of l features from a given set of d measurements/inputs/variables/features, such that $l < d$ "near optimal" features for the learning task (either classification or regression) are obtained. In other words, for a set of d features, the algorithm selects a subset of size $l < d$ features, which contain the greatest ability to discriminate between classes (classification) or predict the target variable's value (regression – time series prediction).

The *goodness* of a particular feature subset is evaluated by using an objective function, $J(Y_m)$, where Y_m is a feature subset of size m. Two types of approaches are the most widely used for the calculation of this kind of objective function: filters and wrappers [86, 87]. Filters rate the goodness of features based on general characteristics, such as interclass distance, statistical independence or degree of correlation between input and output variables, without employing any learning algorithm. Wrappers, on the other hand, evaluate feature subsets based on their predictive accuracy at detecting specific events when employing a particular learning algorithm. The two approaches are not mutually exclusive. In fact it is very common to use a filtering/ranking approach to select a subset of the original feature set and then employ a wrapper to select an even better subset. This hybrid approach is presented in the following section.

Finally we should note that there is a third family of methods, which is called embedded methods that try to select the most relevant features while building the learning algorithm. Tree induction methods belong to that third category of methods.

4.2.2.1 Relief

Relief is a popular algorithm for feature selection originally developed for binary classification problems, but subsequently expanded to tackle both multiclass (ReliefF) as well as regression problems (RReliefF). The successfulness of Relief algorithm is based on its simple and effective evaluation of the features' quality. It is a feature weight based algorithm inspired by instance-based learning [88]. The main difference between Relief and other ranking methods is that it is not a "myopic" algorithm. In other words the contexts of other features is taken into account with the Relief algorithm.

The Relief algorithm searches for the features that are statistically relevant to the target concept using training data D, a sample size m and a τ-threshold ($\tau \in [0, 1]$). The algorithm calculates a weight for each individual feature taking however into consideration its interaction with other features. For the binary classification case the algorithm can be summarized in Algorithm 8.

In Algorithm 8, x_H (near hit) and x_M (near miss) denote the nearest point to x_i in D that belongs to the other and the same class respecively. For each feature F_i the function diff returns the difference of feature values of the two instances.

Algorithm 8 Relief algorithm

Require: A set of data $D = \{(x_i, y_i)\}$ with $x_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$, τ (relevance cut-off value), number of iterations (T)
$z \leftarrow 0$ initialize the weight vector to zero
for $t = 1, \ldots, T$ **do**
Pick random x_k
for $i = 1, \ldots, N$ **do**
$w_i \leftarrow w_i + \frac{\text{diff}(i, x_k, x_M)}{T} - \frac{\text{diff}(i, x_k, x_H)}{T}$
end for
end for
select the feature set whose members exceed the given relevancy cut-off (threshold) τ, that is $S = \{i; w_i > \tau\}$.
return Selected features

In Regression problems, the notions of near hit and near miss do not apply directly. Therefore a revised formulation proposed by Sikonja and Kononeko is used to calculate again weight vectors for each feature [89–91]. In Algorithm 9, N_{dY} is the sum of "probabilities" that two nearest instances belong to different classes, $N^i_{dY \cap dF}$ is the sum of "probabilities" that two nearest instances have different feature values and N^i_{dF} is the sum of "probabilities" that two nearest instances belong to different classes and have different feature values.

Algorithm 9 Relief algorithm tailored for regression

Require: Regression dataset D with M instances and N features, a sampling parameter m, a number n of nearest neighbours
$N_{dY} \leftarrow 0$, $N^i_{dF} \leftarrow 0$, $N^i_{dY \cap dF} \leftarrow 0$, $w^i \leftarrow 0$
for $l = 1, \ldots, m$ **do**
Pick random x_k
Find indices k_j of n nearest instance-neighbours; $j = 1, \ldots, n$
for $j = 1, \ldots, n$ **do**
$N_{dY} \leftarrow N_{dY} + \frac{1}{n} \text{diff}(0, x_{k_j}, x_k)$; index 0 in diff corresponds to target (regression) variable
for $i = 1, \ldots, N$ **do**
$N^i_{dF} \leftarrow N^i_{dF} + \frac{1}{n} \text{diff}(i, x_{k_j}, x_k)$
$N^i_{dY \cap dF} \leftarrow N^i_{dY \cap dF} + \frac{1}{n} \text{diff}(0, x_{k_j}, x_k) \text{diff}(i, x_{k_j}, x_k)$
end for
end for
end for
for $i = 1, \ldots, N$ **do**
$w^i \leftarrow \frac{N^i_{dY \cap dF}}{N^i_{dY}} - \frac{N^i_{dF} - N^i_{dY \cap dF}}{m - N_{dY}}$
end for
return Selected features

One way to tackle the zone temperature prediction is to treat it as a standard regression problem using as regressors delayed values of the input variables and probably of the output variable. In this case the input variables are

```
WBF_Z01_OilControl_FICHSIMV (In1)
WBF_Z01_CombAir_FICHSIMV (In2)
WBF_Z02_OilControl_FICHSIMV (In3)
WBF_Z02_CombAir_FICHSIMV (In4)
```

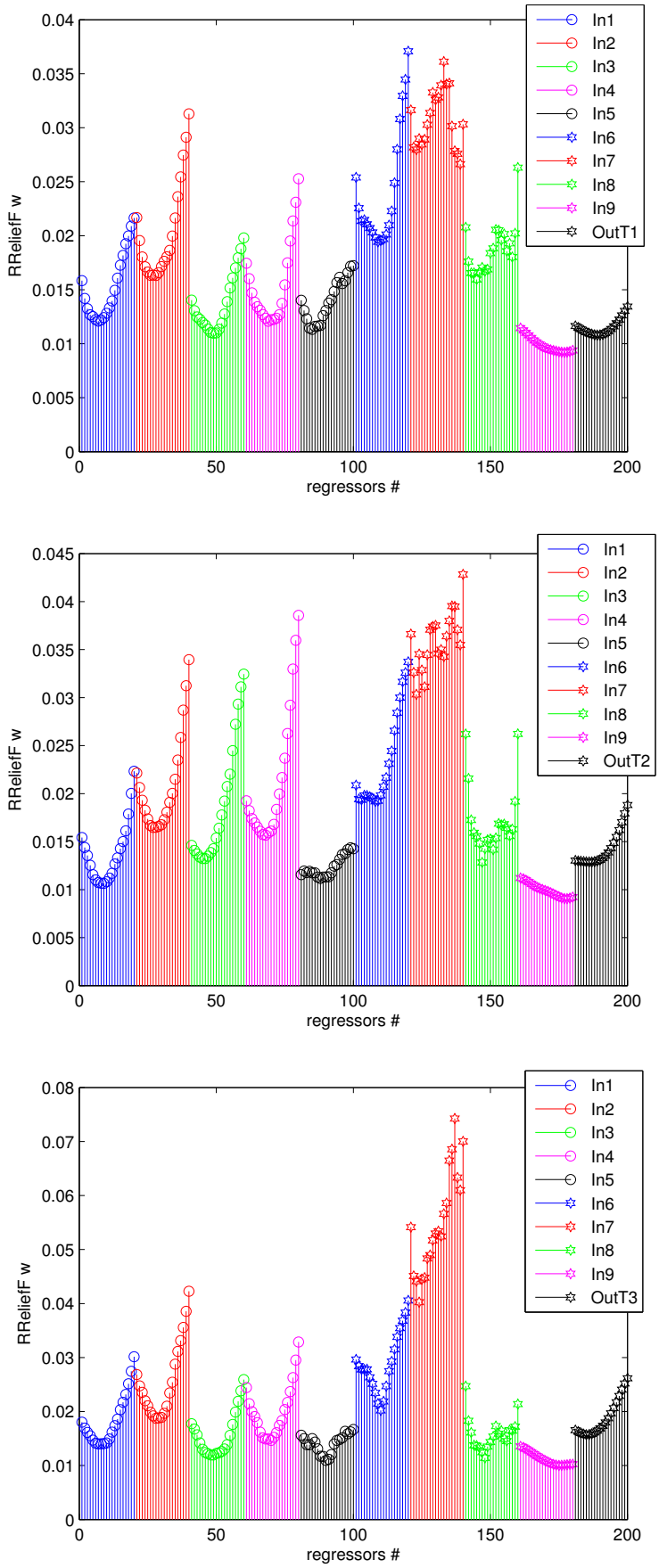

Figure 4.7: Relief weights for the feature selection at three zones of MEFOS's walking beam furnace.

```
WBF_Z03_OilControl_FICHSIMV (In5)
WBF_Z04_CombAir_FICHSIMV (In6)
SU_UML_GB30_SGNHSIValue (In7)
SU_IML_GB6_SGNHSIValue (In8)
WBF_MainExhaust_ExhaustTemperatureHSIMV (In9)
```

while the output/target variable is the temperature at each zone measured over the next time instance

```
WBF_Z01_ZoneTempHSIMV (OutT1)
WBF_Z02_ZoneTempHSIMV (OutT2)
WBF_Z03_ZoneTempHSIMV (OutT3)
```

Using 20 time lags for each of the aforementioned variables leads to an input vector of size 200 (9 input variables by 20 + 20 past values of the output variable), which is quite a high input space. Applying RReliefF the following weights are assigned to each one for the aforementioned regressors for each of the three output/target variables as depicted in Figure 4.7. From the figures it can be observed that the Exhaust variable is correlated with all three target values. Moreover the Oil control variable is more important predictor than the Combustion air variable.

4.2.2.2 Ranking using the Area Under the ROC

Exploring each feature independently for its discriminating capability is the first step in feature selection procedure [92,93]. Along this path, many quantities/methods have been proposed and used for assessing the goodness of a feature: the Fisher's ratio, the correlation coefficient, the correlation ratio etc. Among them a method that is also insensitive to a class imbalance relies on the use of the Area Under the Receiver Operating Characteristic (ROC) curve (AUC) [94]. Wasikowski and Chen [94] proposed the use of the trapezoidal rule with a small number of trapezoids to estimate the AUC, trading accuracy for speed. However for the case of binary classification problems, a more accurate estimation is given using the equation below [95]:

$$\mathrm{AUC} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{i=1}^{n} I(r_i^-, r_j^+), \tag{4.11}$$

where I is given by

$$I(r,s) = \begin{cases} 1, & \text{if } r > s \\ \frac{1}{2}, & \text{if } r = s, \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

and m is the number of negative cases r^-, r_i^- is the value of the feature of the i-th negative case and n is the number of positive cases r^+ and r_j^+ the value (of the feature) of the j-th positive case.

ROC curves are dominantly used in binary classification problems but its use can be also accommodated for multiclass settings [96], where the AUC is averaged across all class pairs.

The use of the average measure of AUC is, however, an approach that neglects possible correlations with other features and might lead to a list of features, which have little to add once put together for a classification problem. A simple solution is to rank features based on AUC value and also take correlation with other features into account. A possible implementation given in Algorithm 10 [92, 93, 97].

Algorithm 10 Ranking algorithm using the Area Under the ROC and the correlation between variables

Require: ...

Rank the features $f_i; i = 1, \ldots, K$ in descending order (where K is the number of features) and select the one f_{i_1} with the highest average AUC value,

$$i_1 \leftarrow \arg\max_j \{\mathrm{AUCa}_j\}_{j=1,\ldots,K}$$

with AUCa_j being the average AUC value of j-th feature.
Select the second feature f_{i_2} for which

$$i_2 \leftarrow \arg\max_j \{w_1 \mathrm{AUCa}_j - w_2 |\rho_{i_1,j}|\}_{j=1,\ldots,K; j \neq i_1},$$

where $\rho_{i_1,j}$ is the cross-correlation coefficient between f_{i_1} and f_j and the weights w_1 and w_2 control the relative importance of AUC value and cross-correlation respectively.
for $k = 3, \ldots, K$ **do**
$i_k \leftarrow \arg\max_j \left\{ w_1 \mathrm{AUCa}_j - \frac{w_2}{k-1} \sum_{r=1}^{k-1} |\rho_{i_r,j}| \right\}_{j \neq i_r}$
end for
return Ranked features

In this algorithm the first feature is ranked based on the average AUC value, the second is ranked taking into account its average AUC value and its correlation with the best feature, and the rest is ranked taking into account their average AUC value and their average correlation with the already "ranked-higher" features. Note that the parameters w_1 and w_2 are selected either by considering expert knowledge or through a cross-validation procedure.

The ranking using the AUC of ROC will be used in Section 4.6 to predict the combustion quality.

4.2.3 Grey-box models based on first principles

In this section we present results using a grey-box modelling approach. The model structure is derived from first principles, which in our case are the heat and mass balance equations, but with unknown parameters. The model parameters are then reconstructed from measurements.

Air, which is used to effectuate the combustion in the three zones of the furnace, is provided by inlets at each zone. This air draft flows upstream from zone 3 to zone 2 and down to zone 1 and, finally, it is let out to the environment through an exhaust pipe. Let F_i be the mass flow of air into zone i. Let us assume that the furnace is firmly closed. Then, the flow of air from zone 3 to zone 2 will be F_3. The flow from zone 2 to zone 1 will be $F_2 + F_3$ and the outflow from zone 1 will be $F_1 + F_2 + F_3$.

Heat flows, roughly speaking, are conditioned by the temperature gradient, the intensity of

the convection and the free area through which the heat may be transferred.

The heat balance equations have the following form:

$$c\frac{\mathrm{d}m_i T_i}{\mathrm{d}t} = \dot{Q}_i - \dot{L}_i - \sum_{j\in C^-(i)} \dot{H}_{ij} + \sum_{j\in C^+(i)} \dot{H}_{ji}, \tag{4.13}$$

where c is a constant to be identified, T_i is the temperature at zone i, m_i is the mass of air at zone i, $\dot{L}_i$ is the rate of heat losses from zone i to the environment, $\dot{H}_{ij}$ is the rate of heat exchange from zone i to zone j. Since, however, $T_i > T_j$ for $i > j$, heat flow upstream, from zone 3 to zone 1. This is further promoted by the air draft which flows in this direction, therefore, this simplifies the above model as: Heat accumulation = heat supply – heat losses to the environment – heat convection upstream + heat convection from downstream, or,

$$c\frac{\mathrm{d}m_i T_i}{\mathrm{d}t} = \dot{Q}_i - \dot{L}_i - \sum_{j<i} \dot{H}_{ij} + \sum_{j>i} \dot{H}_{ji}, \tag{4.14}$$

We will assume that the heat supply is proportional to the fuel flow, that is

$$\dot{Q}_i = \kappa \dot{f}_i, \tag{4.15}$$

for some positive constant $\kappa > 0$. We assume that the heat losses depend linearly on the temperature gradient between the zone temperature T_i and the ambient temperature T_∞, that is

$$\dot{L}_i = \lambda(T_i - T_\infty) + \dot{L}_i^d, \tag{4.16}$$

for some $\lambda > 0$, where, of course $T_i > T_\infty$ and $\dot{L}_i^d$, for $i = 1, 3$, are the heat losses because of the doors of the furnace not being closed. The latter can be modelled as

$$\dot{L}_i^d = \lambda^d \delta_i F(T_i - T_\infty), \tag{4.17}$$

where $\delta_i \in [0, 1]$ is the door status at zone i with $\delta_i = 0$ meaning that the door is firmly closed and $\delta_i = 1$ meaning that the door is fully open and F is the flow of the air draft which promotes the convection of heat.

Finally, the heat convection rate between zones will be given by

$$\dot{H}_{ij} = \beta_i F(T_i - T_j) \tag{4.18}$$

for $i > j$. Overall, the system model becomes

$$c\frac{\mathrm{d}m_i T_i}{\mathrm{d}t} = \kappa \dot{f}_i - \lambda(T_i - T_\infty) - \lambda^d \delta_i F(T_i - T_\infty) - \sum_{j<i} \beta_i F(T_i - T_j) + \sum_{j>i} \beta_j F(T_j - T_i). \tag{4.19}$$

The derivative in the left hand side of (4.19) can be written as

$$\frac{\mathrm{d}m_i(t)T_i(t)}{\mathrm{d}t} = m_i(t)\frac{\mathrm{d}T_i(t)}{\mathrm{d}t} + \frac{\mathrm{d}m_i(t)}{\mathrm{d}t}T_i(t). \tag{4.20}$$

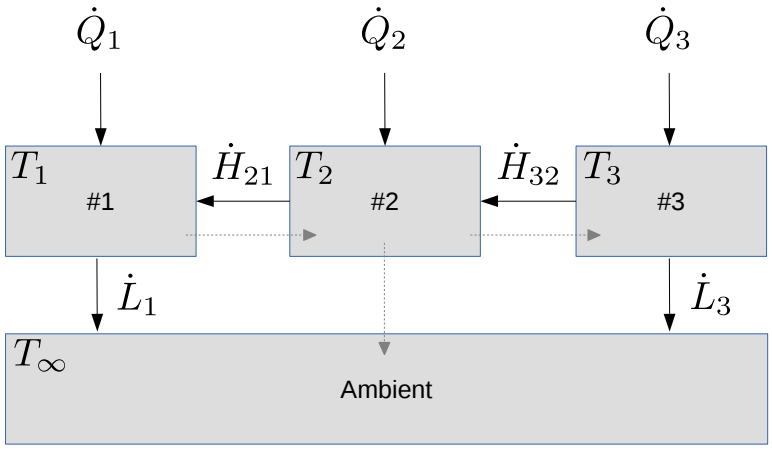

Figure 4.8: Sketch of the heat flows through the walking beam furnace. Heat flows $\dot{Q}_i$ come from the burning of fuel. Heat losses to the environment are denoted as $\dot{L}_i$ and depend on the doors' status. Heat flow from zone i to zone j is denoted as $\dot{H}_{ij}$.

In Figure 4.8 we present a conceptual illustration of the physical model of the system. The light-grey arrow indicate terms which are neglected: we assume that there is no downstream heat flow and the losses from zone 2 are negligible. This assumption is meaningful because zone 2 is insulated by zones 1 and 3, so the only heat losses are through the furnace walls.

By the mass balance in the furnace we have that

$$\frac{\mathrm{d}m_i(t)}{\mathrm{d}t} = F(t). \tag{4.21}$$

The above analysis has not taken into account a series of complex phenomena such as turbulence, heat transfer by radiation, but still introduces a structure for the WBF as it is described in Figure 4.8. What is most important is the fact that even with this elementary analysis, we can see that the physical model of the system turns out to be nonlinear. Therefore, linear models can only be used as approximate models in the context of control and not for open-loop simulations. The above analysis is indicative of a model structure which determines which input and output variables affect which output ones which will be later used in Section 4.2.5 (see for example Tables 4.3 and 4.4).

4.2.4 Subspace identification of linear models

Discrete time state-space formulation of a linear dynamical system is given as

$$x_{k+1} = Ax_k + Bu_k + Ke_k, \tag{4.22a}$$
$$y_k = Cx_k + Du_k, \tag{4.22b}$$

Subspace state-space system identification methods, often abbreviated as 4SID or subspace identification methods, aim to identify system matrices A, B, C, and D directly from input-output data. Hence, we do not assume any kind of structure of the model, except in our case

Variable name
WBF_Z0[1-3]_ZoneTempHS:IMV
WBF_Z0[1-3]_OilControl_FICHSI:MV
WBF_Z0[1-3]_CombAir_FICHSI:MV
SU_IML_GB6_SGNHSIValue
SU_UML_GB30_SGNHSIValue
WBF_MainExhaust_ExhaustFlow_FICHSI:MV
WBF_Z0[1-3]_ColdCAirOutputHSI:MV
WBF_Z0[1-3]_O2_QICHSI:MV

Table 4.2: Variables used in subspace identification

that $D = 0$. These methods provide an alternative to classical methods, such as ARX, but are founded on completely different principles. They draw heavily on geometric projections and linear algebra methods, such as QR decomposition and SVD, which gives them favorable numeric properties. More specifically, they exploit the relation of state-space realization and the convolution sum of a discrete-time dynamic system though so-called Hankel matrix. In turn, this matrix can be factored as a product observability and controllability matrix, which depend on system matrices A,B and C. By using linear algebra methods these system matrices can be reliably estimated.

On the other hand they lead to linear time-invariant systems with a state-to-output relationship of the form (4.22b), which calls for the design of a state observer if the model is to be used for control purposes.

Furthermore, the only parameter to be decided is the model order. This avoids need for complicated parametrization even when dealing with complex models [98]. However, the theory behind them is somewhat involved and they typically need a large number of data to estimate accurate models.A good overview of subspace identification methods for linear systems is given in [99]. Another obstacle is a need for special care when dealing with data obtained in closed loop which is our case. Luckily, today these methods have mature and reliable implementations already exist, such as Matlab's `n4sid` algorithm.

Informally speaking, unlike ARX or ARMAX methods which optimize for one-step ahead predictions, subspace methods effectively optimize for j-steps-ahead prediction. This means that multiple steps ahead prediction is accounted for directly and this is what is typically desired for models that used in model predictive control. As we will see in next chapter, when dealing with ARX model K-steps ahead prediction, we actually perform K one-step ahead predictions. Doing this can introduce a bias in our predictions, because different steps ahead predictions may not behave similarly.

Predictive performance obtained with Matlab's N4SID algorithm implementation is depicted in 4.9. Variables used in subspace identification are listed in Table 4.2.

4.2.5 Linear auto-regressive models

ARX model description In this section we present an overview of efforts in identifying an ARX type system models. In short, we are interested in finding a linear mapping of past values of

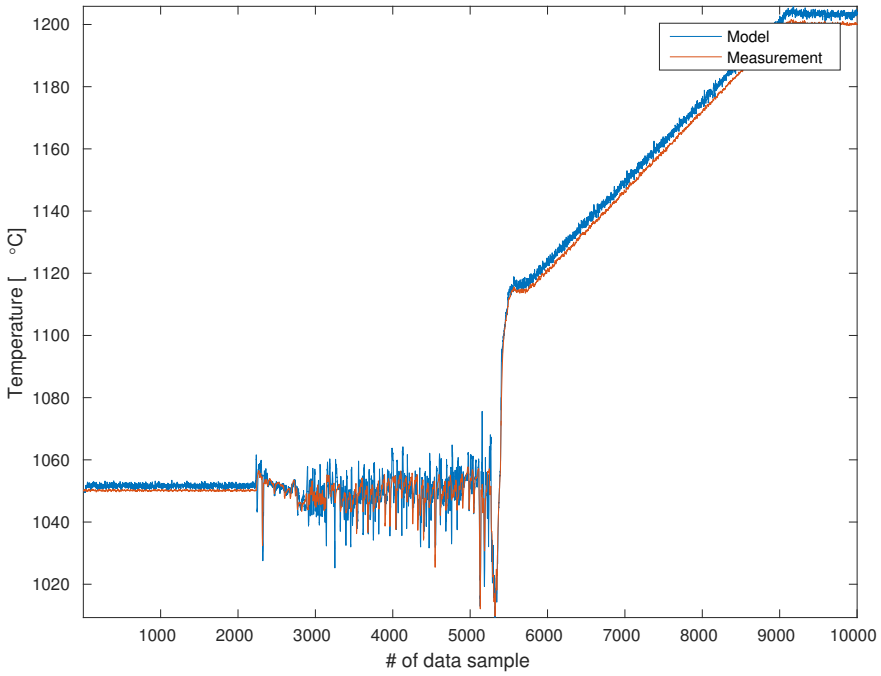

Figure 4.9: Results obtained with subspace identification method

both inputs and outputs of a process to predict it's future behaviour. Thus, model of the furnace has the following form:

$$y_m(k) = \sum_{i=1}^{N_y} \sum_{j=1}^{N_y^i} a_j^i y^i(k-j) + \sum_{i=1}^{N_u} \sum_{j=1}^{N_u^i} b_j^i u^i(k-j), \tag{4.23}$$

where N_y is the number of process outputs and $m \in \{1, \ldots, N_y\}$. N_y^i is the number of values of output $y_i(k)$ we will use. Total number of inputs is N_u, while N_u^i is the number of past values of input data for input i. Coefficients a_j^i and b_j^i are to be estimated to provide best possible fit according to a specified criterion. Notice that we can rewrite (4.23) as

$$y_m(k) = \phi(k)^\top p. \tag{4.24}$$

Column vector $\phi(k)$ is called a vector of regressors at time k and is a collection of $y^i(k-j)$ and $u^i(k-j)$ values, while p is an unique vector of corresponding coefficients as defined above.For convenience we will define the following

$$A = \begin{bmatrix} \phi(H_m+1)^\top \\ \phi(H_m+2)^\top \\ \vdots \\ \phi(H_m+N_D)^\top \end{bmatrix}, \tag{4.25}$$

here N_D is number of data points in our training data set. The standard approach to system identification suggests finding a vector of parameters p which solves the following optimization

problem

$$\text{minimize } \frac{1}{2}\|Ap - y_m\|_2^2. \tag{4.26}$$

This is well known least squares problem and it's solution can be expressed in closed form. We also refer to formulation as minimizing L2 penalty. Similar problem can be formulated with L1 penalty which would read

$$\text{minimize } \frac{1}{2}\|Ap - y_m\|_1. \tag{4.27}$$

Both problems are convex and can be solved easily for reasonable dimensions using standard techniques like the ones given in [100]. Because of oscillatory predictions , we will introduce a new optimization problem which will take into account the overall fit of the data but will account for oscillations in the prediction vector which, in general, are not desirable by additional penalty term. Now, the optimization problem reads

$$\text{minimize } \frac{1}{2}\|Ap - y_m\|_2^2 + \lambda_{SR}\|\Delta(y_p)\|_2^2 \tag{4.28}$$

The vector in the second term Δy_p is defined as $\Delta y_p(k) = y_p(k+1) - y_p(k)$, $\forall k \in \{1, \ldots, \text{len}(y_p) - 1\}$ and is usually called slew rate of the signal (hence λ_{SR}), where $\lambda_{SR} > 0$ is a user defined parameter that accounts for trade-off between model fit and smoothness of predicted values. Larger values of λ_{SR} will enforce a very smooth response at the cost of accuracy, and vice versa.

Model training and validation The dataset given by our industrial partners is partitioned into two separate sets; training and testing set, with 25000 and 10000 data points respecively. As the names suggest, the model will be identified on a training set and then validated on a separate set of data in order to test it's predictive abilities outside the training region. This standard technique is used to avoid over-fitting of the data on the training set. For example, an overly complicated model will be able to achieve superior performance on training set by adapting to measurement noise of the provided data, but will have poor performance on testing set. Striking the balance between model complexity that captures significant characteristics of a process and is simple enough not to capture noise, is precisely why we use two separate sets. Partition of the data is shown in the Figure 4.10; notice that the mid part of the data in this figure is omitted. This is both because these data correspond to low temperatures and because during that period the furnace experienced certain technical problems according to MEFOS.

We believe that testing data captures all of the important operating conditions of the plant. It shows a region where steady set point is required, it shows a region where there is a lot of movement of material and door openings and it shows a region where there is a ramp like increase of the temperature.

However, by pure visual inspection, we are able to observer a significant similarly between two input variables, namely, oil expenditure and combustion air (draft). This is depicted for test and current training data in Figures 4.11 and 4.12 where these two vectors are plotted against

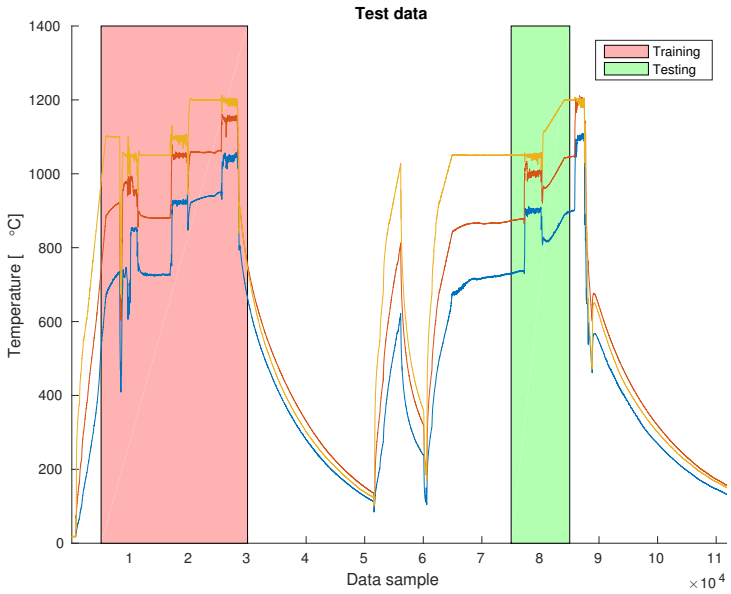

Figure 4.10: Partitioning of the data provided by industrial partner. Red patch in the image is the data used to train a model, whereas green patch indicates a testing set. Note that we have selected regions whose operating temperature is above 500 degrees Celsius which indicates normal operating regime.

each other for all three zones.

All of the predictive performance regarded thus far has been concerned with one step ahead prediction, i.e., the model is required to predict only one time step into the future (one sampling time). However, model predictive control requires N steps ahead prediction because one step ahead prediction is usually not informative enough. Ideally, we would like to have N very large, but more often than not this is not feasible. Hence, we introduce the following performance measure to evaluate K steps ahead predictive ability of a model

$$\mathrm{PMSE}_k = \frac{1}{N}\sum_{j=1}^{N} \epsilon_{k+j|k}^2. \tag{4.29}$$

In order to have only one measure for the entire model and not one indicator for each one of the predicted temperatures (*states*, in general) we will define $\epsilon_{k+j|k}$ as a norm of the vector of prediction errors

$$\epsilon_{k+j|k} = \|\hat{y}_{k+j|k} - y_{k+j}\|, \tag{4.30}$$

where $\hat{y}_{k+j|k}$ and y_{k+j} are vectors of predictions and measurements of size $N_y = 3$ in our case (walking beam furnace has three different zones).

Results Here we present predictive K-steps-ahead performance of an identified model. From a computational stand point this is a series of K successive one-step predictions, where an one-step-ahead prediction at time $k+1$ from the previous time step k is used instead of measured value at time $k+1$. This is true only for output variables. Input values are assumed to be known

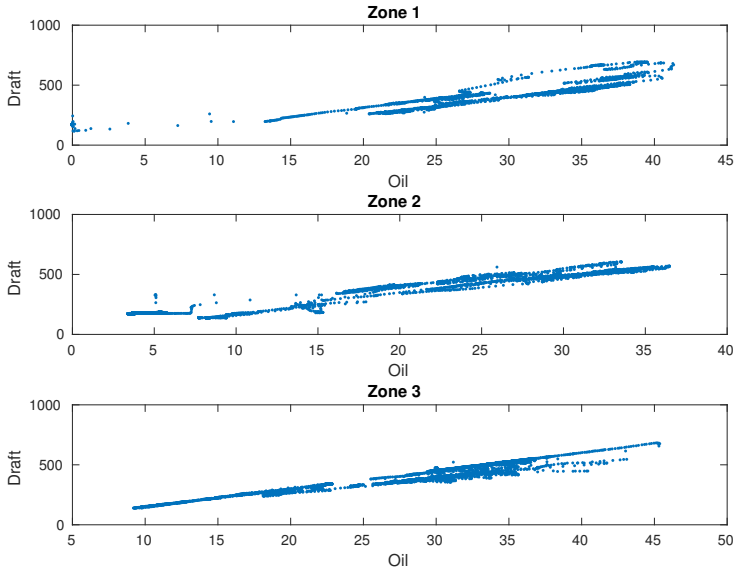

Figure 4.11: Correlation between burned oil and draft of combustion air inside the furnace on the set used for testing

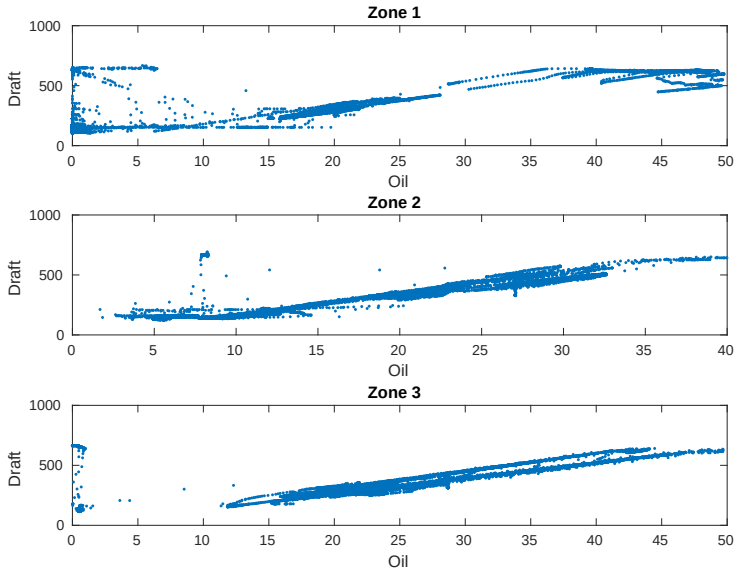

Figure 4.12: Correlation between burned oil and draft of combustion air inside the furnace on a training set

along the prediction horizon which is a realistic assumption and suitable for control applications where input variables are decided by the controller. This way, we obtain a realistic assessment of performance in a real world scenario.

We provide the list of input variables used in this model in Table 4.7 and the model structure is presented in Tables 4.3 and 4.3 where we show the time lags considered for each variable. For example, $T_1(k+1)$ is predicted using q past values of T_1[1], k past values of T_2 and, according to Table 4.4, k past values of the oil flow and combustion air at zone 1 as well as k past value of the load hatch-door status. We have associated the load door status with T_1 alone and the unload door status with the temperature at the third zone. We assume that the temperature at zone 2 is indirectly affected by the doors' statuses via T_1 and T_2 respectively.

	T_1	T_2	T_3
T_1	q	k	
T_2	k	q	k
T_3		k	q

Table 4.3: Output-output correspondence of variables and the lags used for each variables. We use the parametrisation $q = \lfloor 1.67k \rfloor$.

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
T_1	k	k					k	
T_2			k	k				
T_3					k	k		k

Table 4.4: Output-input correspondence of variables and the lags used for each variables. *Input varialbes.* u_1, u_3, u_5: Oil consumption rate at zones 1, 2 and 3 respectively. u_2, u_4, u_6: Combustion air flow at zones 1, 2 and 3 respectively. u_7: load hatch-door, and u_8: unload hatch-door.

Figures 4.13 and 4.14 show predictions for different values of K and the $\mathrm{AV@R}_\alpha$ values of the PRMSE of various models are given in Tables 4.5 and 4.6. In Figure 4.15 we see an example predicted trajectory of an ARX model along with the actual trajectory.

It can be seen that prediction becomes worse as we increase parameter K, which is not surprising and is to be expected. However, there is a noticeable degradation of predictive performance in the region where there is a lot of material movement and door opening. It is observed that the predictive performance index does not change significantly when we disregard information on furnace door openings. This could be due to system operating in closed loop and by doing so successfully compensates for the disturbance introduced by door openings.

4.2.6 Sparse models — LASSO and Elastic net

In this section we will describe efforts taken to identify sparse models of the underlying physical process. Our working assumption is that only some past values of system variables are important to obtain a good predictor model. We aim at finding *simple* models that still explain the data with sufficient accuracy. By doing so we hope to obtain a model that will have superior generalization properties, based on the reasoning that simple model is not able to capture noise. This

[1] that is, $T_1(k-j)$ for $j = 0, \ldots, k-1$.

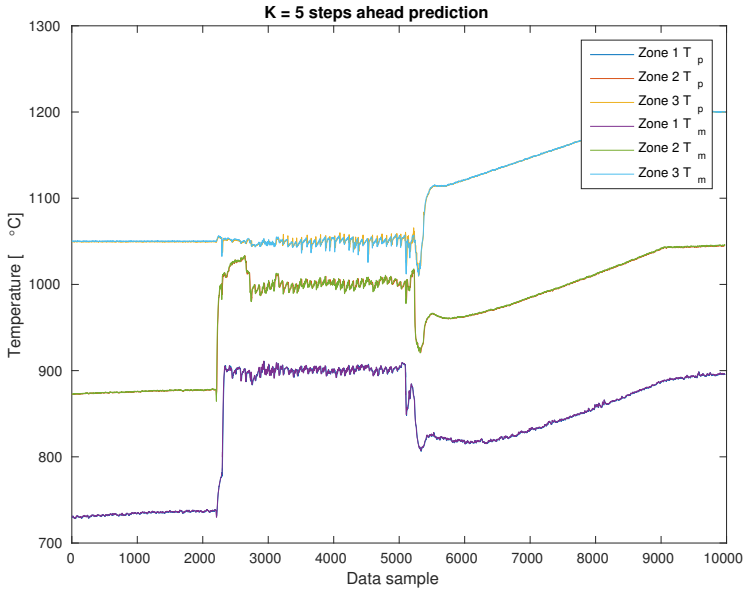

Figure 4.13: Prediction performance on all zones for 5 steps ahead prediction.

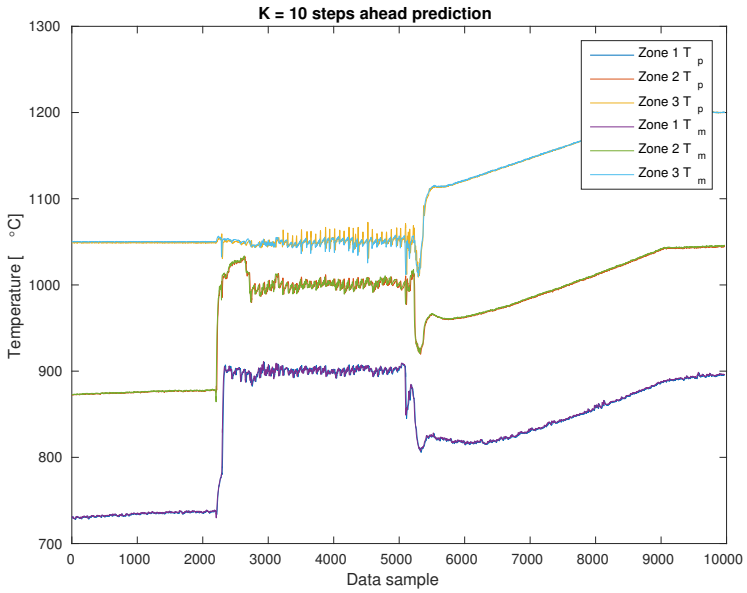

Figure 4.14: Prediction performance on all zones for 10 steps ahead prediction

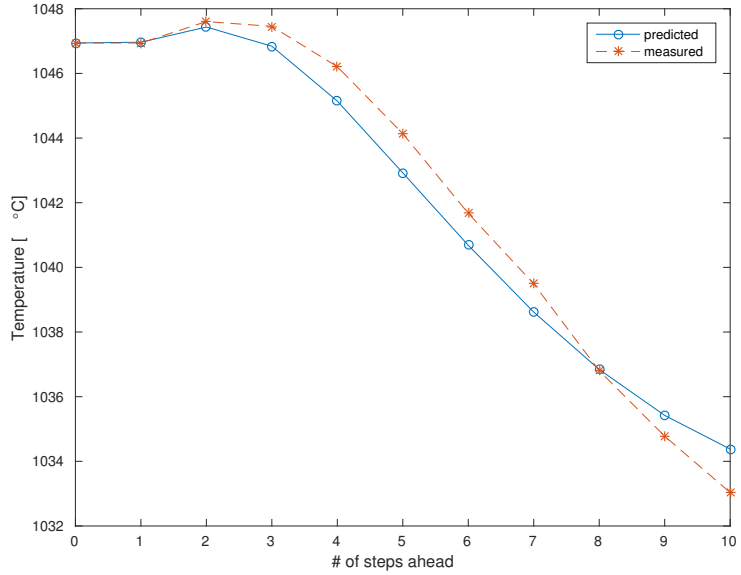

Figure 4.15: An illustration of predictive ability of trained model. Predicted trajectory is compared to measured temperature in Zone 3 at some time instant.

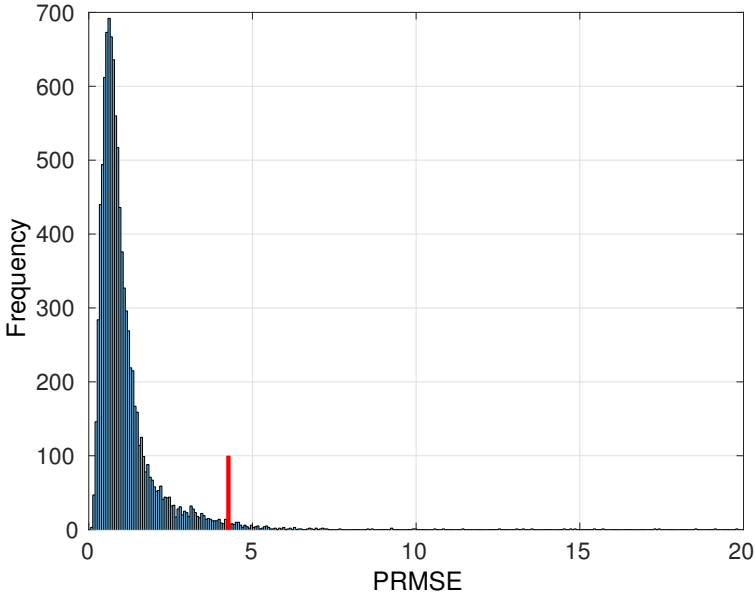

Figure 4.16: Depiction of distribution of PRMSE error measure. Red vertical line indicates $\mathrm{AV@R}_\alpha$ for $\alpha = 0.05$. Roughly speaking this is a quantification of the importance/impact of the 5%-right tail.

q	k	Penalty	Smooth	$\mathrm{AV@R}_{0.05}$	$\mathrm{AV@R}_{0.1}$
8	5	L2	No	4.2336	3.2574
8	5	L1	No	4.4440	3.4156
20	12	L2	No	4.2331	3.2562
20	12	L2	Yes	17.0589	12.4930
20	12	L1	No	4.4466	3.4169
33	20	L2	No	4.2794	3.2936
33	20	L1	No	4.4617	3.4260
50	30	L2	No	4.3007	3.3132
50	30	L1	No	4.4299	3.4044

Table 4.5: Summary of system identification results for $K = 5$ steps-ahead prediction. Column named *Smooth* stands for standard least squares problem with additional smoothing parameter as defined in (4.28).

q	k	Penalty	Smooth	$\mathrm{AV@R}_{0.05}$	$\mathrm{AV@R}_{0.1}$
8	5	L2	No	8.7980	6.8428
8	5	L1	No	9.1353	7.0752
20	12	L2	No	8.8208	6.8744
20	12	L2	Yes	20.9694	15.4784
20	12	L1	No	9.1512	7.0864
33	20	L2	No	8.8798	6.9127
33	20	L1	No	9.1265	7.0690
50	30	L2	No	8.9429	6.9885
50	30	L1	No	9.0637	7.0211

Table 4.6: Summary of system identification results for $K = 10$ steps-ahead prediction. Column named *Smooth* stands for standard least squares problem with additional smoothing parameter as defined in 4.28.

is nor unreasonable assumption but the problem is how to select those inputs. A well known technique is to solve ℓ_1 regularized least squares problem, where penalization is imposed on models parameters. This technique is otherwise known as Least Absolute Shrinkage and Selection Operator (LASSO) [12, 101] and enables us to select model structure based on the data only. Furthermore, results of LASSO can help us interpret better interpret the model. LASSO is used in various fields, perhaps most notably in analysis of biological data which comprises of great number of factors, of which only some contribute to predictive ability [101]. LASSO problem batch formulation is shown below

$$\text{minimize } \tfrac{1}{2}\|Ap - y\|_2^2 + \lambda\|p\|_1. \tag{4.31}$$

Parameter λ enforces sparsity in the model, meaning that higher λ will tend to force more values of p towards zeros and is usually chosen by cross-validation. For WBF process, we solve three different LASSO problems to obtain three different models which we will use for prediction. Hence, we will replace appropriate terms in formulation 4.32 with A_i, p_i, y_i, where $i \in \{1, 2, 3\}$ Afterwards, we solve standard least squares problem for reduced order system which is usually called *debiasing*.

Similar to LASSO, elastic net is another technique used in model selection. The associated

Variable name
WBF_Z0[1-3]_ZoneTempHS:IMV
WBF_Z0[1-3]_OilControl_FICHSI:MV
WBF_Z0[1-3]_CombAir_FICHSI:MV
SU_IML_GB6_SGNHSIValue
SU_UML_GB30_SGNHSIValue

Table 4.7: Variables used in ARX identification

optimization problem is

$$\text{minimize } \tfrac{1}{2}\|Ap - y\|_2^2 + \lambda\|p\|_1 + \frac{\lambda^2}{2}\|p\|_2^2. \tag{4.32}$$

Results are reported in table 4.8. It can be seen that Elastic net method produces models that give lower $\mathrm{AV@R}_\alpha$ score, at least on the test data.

Method	λ	$\mathrm{AV@R}_\alpha$
LASSO	30	8.9751
LASSO	60	9.3528
LASSO	90	8.9414
LASSO	120	8.2951
LASSO	150	9.1920
Elastic net	30	8.0293
Elastic net	60	7.9545
Elastic net	90	8.3693
Elastic net	120	8.2951
Elastic net	150	8.4822

Table 4.8: Summary of model selection algorithms. Performance is measured by $\mathrm{AV@R}_\alpha$, with $\alpha = 0.1$. In all of the models history of $k = 80$ was used with different λ parameter.

4.2.7 Adaptive sparse models

Having a lower complexity model for which we believe that it can properly capture behaviour of the system well enough, doesn't mean that it will perform well in all regimes. This could be due to some unforeseen changes in the process or due to an operation in a new set point that was not captured by the training data. One way of dealing with this issues is model reconfiguration or adaptive modelling. Usually, in system identification, the whole set of data is collected and batch regression is performed. This approach is usually not adopted for adaptive modelling , because of the computational burden. What is preferred is the recursive approach, in which after each sampling instant, we update our model with the latest measurement. If we also include a forgetting factor in our model, we can put more importance on recent measurement rendering our model up-to-date with latest operating conditions.

To motivate this idea, below we compare results using batch least squares estimation against model selection with LASSO and least squares re-estimation after each measurement. Forgetting factor was used not used in this particular instance. Results are presented in Figures 4.18

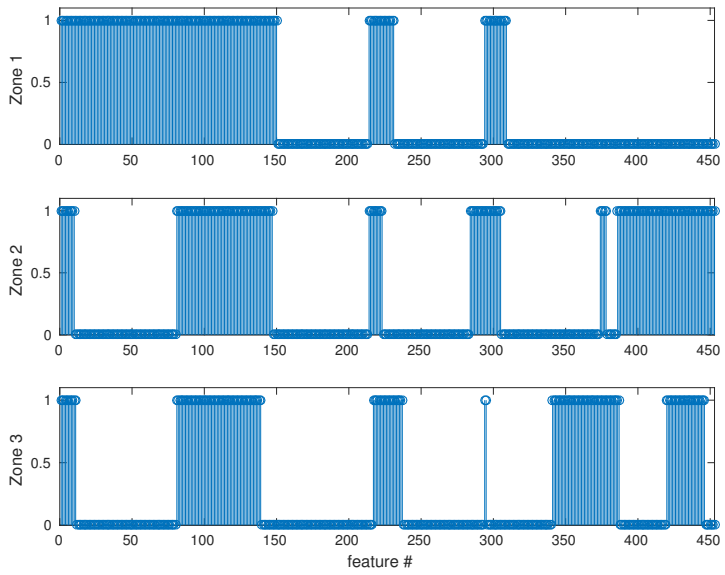

Figure 4.17: Sparsity pattern of all three zones as chosen by elastic net algorithm with $\lambda = 60$. Zeros in the figure signal variables that are discarded from the model. Model is reduced to about 40% variables out of possible 450.

and 4.19.

Optimization problem has the following form

$$\text{minimize } \frac{1}{2}\|Ap - y_m\|_W^2 + \lambda\|p\|_1 \tag{4.33}$$

Here, W is a matrix of weight parameters

$$\begin{pmatrix} 1 & & & & \\ & \lambda & & & \\ & & \lambda^2 & & \\ & & & \ddots & \\ & & & & \lambda^{N-1} \end{pmatrix} \tag{4.34}$$

where N is a number of measurements. The sum in (4.33) expands to

$$\text{minimize } \frac{1}{2}(x_1^\top p - y_m^1) + \lambda(x_2^\top p - y_m^2) + \cdots + \lambda^{N-1}(x_N^\top p - y_m^N). \tag{4.35}$$

If the forgetting factor is chosen like $\lambda \in (0, 1)$, then older measurements are given lower weight. This accounts for our assumption that process may change and newer measurements should bear more relevant data. In our case we chose $\lambda \approx 0.99$.

An interesting concept which can be employed to accelerate the online training of models is that of *online censoring* [102] which was very recently (2016) used for regression over a stream of data. The concept is based on the adaptive training approach we described above and the same principles we used in Section 2.1.2, but certain data points can be omitted if the existing model can predict them with adequate accuracy.

Effectively, when the system dynamics is described by $y_{k+1} = a^\top x_k + b^\top v_k + e_k$, where $y_{k+1} \in \mathbb{R}$, $x_k = (y_k, \ldots, y_{k-H_y})$ and $v_k = (u_k, \ldots, u_{k-H_u})$ and $e_k \sim \mathcal{N}(0, \sigma^2)$, then when a new

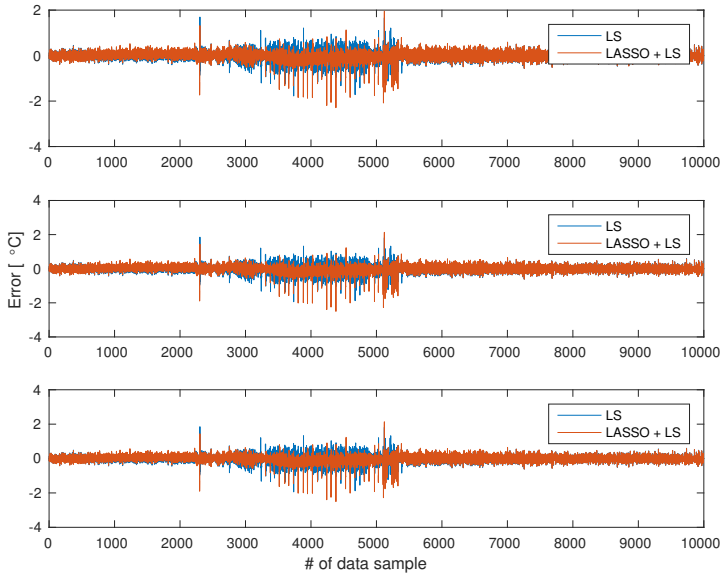

Figure 4.18: One step ahead prediction error for standard least squares estimation vs LASSO with recursive least squares. Forgetting factor was not used in this experiment.

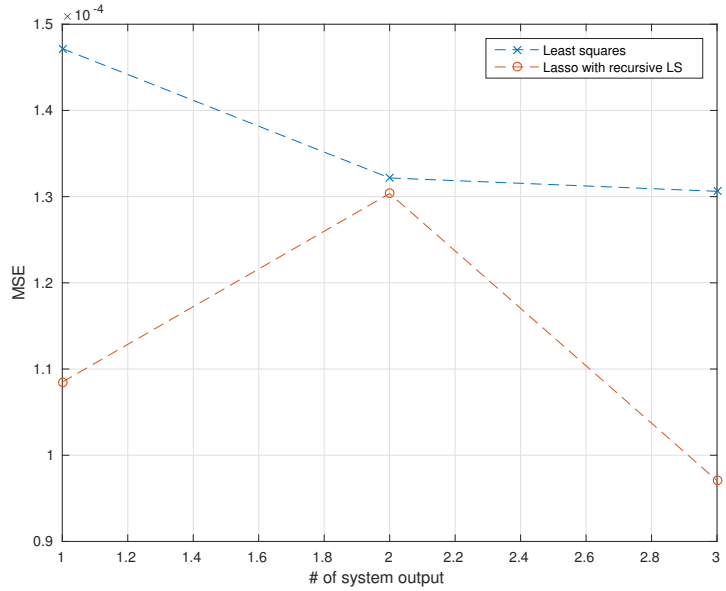

Figure 4.19: Mean squared error of one step ahead prediction for standard least squares estimation vs LASSO with recursive least squares. Forgetting factor was not used.

instance (x_k, v_k, y_{k+1}) arrives, it is used to update the model only if

$$\left| \frac{a^\top x_k + b^\top v_k - y_k}{\sigma} \right| > \tau_k, \tag{4.36}$$

where τ_k are threshold parameters which are determined as explained in [102]. This can further accelerate the process of model building, especially when this is performed online.

4.2.8 Nonlinear dynamical models

4.2.8.1 Support vector machines for regression

Supposed a set of n observations is provided. In the general case, each observation, $i = 1, \ldots, n$ lies in m-dimensional space $x_i \in \mathbb{R}^m$. Along with each x_i there is an observed target variable y_i which we need to predict. The regression problem consists in determining a function f which predicts y given x.

A good answer to this problem is given in the support vector context. Support Vector Machines (SVM) is a notion introduced in the exceptional work of Vapnik [103]. As an extension to this theory, SVR cover the corresponding regression task. The problem, in its simplest formulation is stated as follows. Given a parametrization

$$f(x) = w^\top x + b \tag{4.37}$$

find w which minimizes the functional

$$\frac{1}{2} w^\top w + C \frac{1}{n} \sum_{i=1}^{N} \max(|y_i - f(x_i)| - \epsilon, 0). \tag{4.38}$$

The second term of the functional is an ϵ-sensitive cost function and that is where the ϵ-SVR name comes from. To add generality and flexibility to the method, slack variables ξ_i and ξ_i^* are incorporated. The optimization problem is modified as follows

$$\min \frac{1}{2} w' w + C \sum_{i=1}^{N} \xi_i + \xi_i^*, \tag{4.39}$$

subject to the constraints

$$y_i - w' x_i - b \leq \epsilon + \xi_i \tag{4.40}$$

$$w' x_i + b - y_i \leq \epsilon + \xi_i^* \tag{4.41}$$

$$\xi_i, \xi_i^* \geq 0 \tag{4.42}$$

which is a quadratic optimization problem with linear constraints. The term $\sum_{i=1}^{N} \xi_i + \xi_i^*$ corresponds to an upper bound to the number of points not belonging to function f where N is the number of points that reside outside the ϵ margin. Constant C regulates the tradeoff between a smoother or a more accurate interpolation function and it is a parameter provided by the user.

A large C reduces the training error but increases the model complexity, i.e., the number of support vectors, and, therefore, the training time and possibly generalization ability. A smaller C results in a less complex model with higher mean square error but possibly resulting in better generalization. The final, dual optimization problem can be formulated as follows

4.2.8.2 Assessment of nonlinear dynamical models

For the prediction of the temperature in the three zones, first RReliefF was used to rank the 200 regressors. Then the nested subset selection approach was used along with a linear regression method to estimate the necessary number of regressors. That original analysis revealed that more than of the 120 top ranked regressors are needed to achieve a substantial drop in $\mathrm{AV@R}_\alpha$. Based on that observation, an SVR was tested achieving $\mathrm{AV@R}_{0.05} = 12.5765$ $\mathrm{AV@R}_{0.1} = 10.0856$.

4.3 Uncertainty propagation

We reviously presented various predictive dynamical models and proposed a method to evaluate their predictive ability. When, however, it comes to using these models to make predictions it is expedient to know, not only how good we expect the prediction to be, but what are some probable realisation of the future evolution of the system trajectory. Such a collection of future state trajectories, $\{x^i_{k+j|k}\}_{i,j}$ is known as a *scenario fan* as discussed in Chapter 3. Scenario fans and scenario trees, in a way, describe how uncertainty propagates throughout the prediction horizon and it typically gives rise to representations as in Figure 4.20. The three plots in the first row of Figure 4.20 are the original scenario fans which have been constructed using actual process data (these are sequences of temperature prediction errors in $^\circ C$) and count 10^4 (equiprobable) trajectories. Using the scenario reduction methodology we described in Section 3.1.2 we then derived the plots of the second row.

The scenario tree structure is better illustrated in the conceptual Figure 4.21 where we see how $10,000$ scenarios are compressed into a scenario tree of no more than 100 scenarios. Notice that scenario reduction not only reduces the complexity of the original scenario fan, but it also removes scenarios which are very unlikely to happen; indeed, notice the scale of the y-axis in the plots of the first compared to the second row in Figure 4.21.

4.4 Model predictive control

For the model predictive control problem formulation we identify two main objectives: (i) to retain the temperatures at each zone as close as possible to the desired temperature set-points with higher importance on the tracking error at zones 2 and 3, (ii) to penalise the consumption of fuel at each zone. For this purpose, we use the stage cost function

$$\ell(x, u; x^{sp}) = \tfrac{1}{2}\|x - x^{sp}\|^2_Q + \tfrac{1}{2}\|u\|^2_R, \tag{4.43}$$

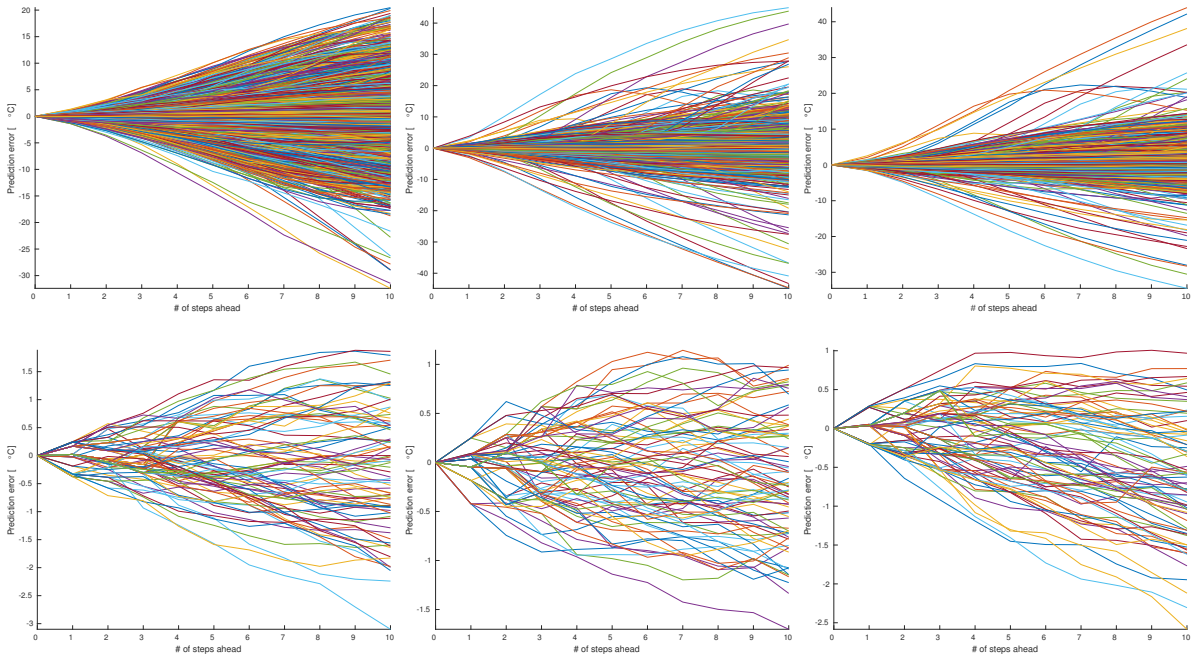

Figure 4.20: Error plots for all three furnace zones. Upper row shows all of the error trajectories calculated at every time instant in the test data. Lower row shows reduced error trajectories by using scenario tree with at most 100 scenarios.

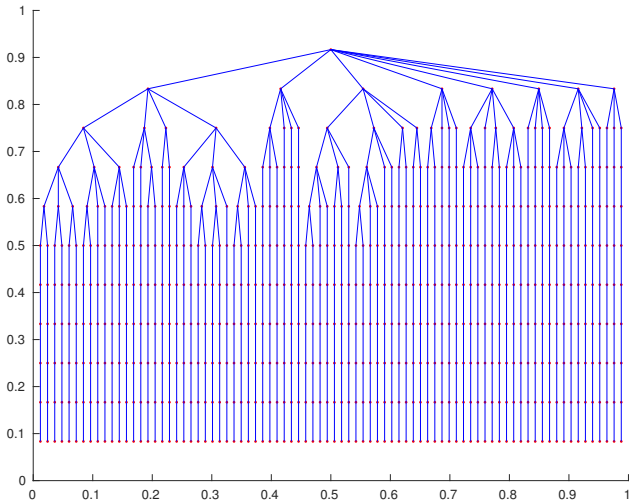

Figure 4.21: Scenario tree structure describing error propagation in the model

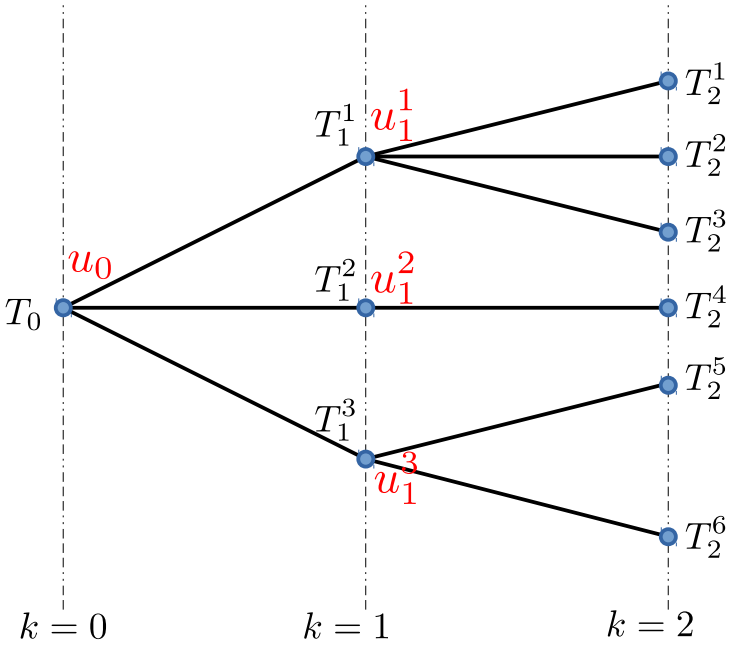

Figure 4.22: Scenario tree structure for the predicted state sequence and causal control actions along the scenario tree.

where $Q = \mathrm{diag}(Q_1, Q_2, Q_3)$ is a diagonal matrix where $Q_i > 0$ is the tracking error weight for zone i with $Q_1 < Q_2 < Q_3$, x^{sp} is the temperature set-point which is specified by the operator and R is a digonal matrix with zeros on the diagonal at the positions which don't correspond to the fuel consumption. The notation $\|\cdot\|_Q^2$ is meant as $\|x\|_Q^2 = x^\top Q x$. Matrices are Q and R and we will discuss later about how they can be chosen and updated online using PAT data (see Figure 3.1).

Additionally, the SMPC controller needs to take into account the input constraints. Let $u = (u_1, u_2, u_3)$ where $u_1 \in \mathbb{R}^3$ are the fuel supply rates, u_2 is the exhaust flow and $u_3 \in \mathbb{R}^3$ are the combustion air flows. Then

$$0 \leq u_1 \leq 50, \tag{4.44}$$

$$0 \leq u_2 \leq 2400, \tag{4.45}$$

$$0 \leq u_3 \leq \begin{bmatrix} 1000 \\ 900 \\ 1000 \end{bmatrix}. \tag{4.46}$$

In this case study, causality is to be interpreted as in Figure 4.22. Indeed, at some time instant k we observe the current temperature $T_k = T_{k|k}$, but we don't have accurate knowledge of the upcoming temperature $T_{k+1|k}$. Temperature $T_{k+1|k}$ is estimated as a function of past observed outputs and inputs of the system, but its actual future realisation is unknown. We assume that it is an instance of a finite-dimensional probability space with sample space $\{T^i_{k+1|k}\}_i$. At k the decision variable $u_{k|k}$ cannot be evaluated as a function of T_{k+1} since this is unknown and has not occured yet, so $u_{k|k} = \psi_{k|k}(T_{k|k}, \text{other observed info})$. Instead, $u_{k+1|k}$ is a function of T_{k+1}. Recursively, this flow of information (formally known as a filtration) is reflected on the tree

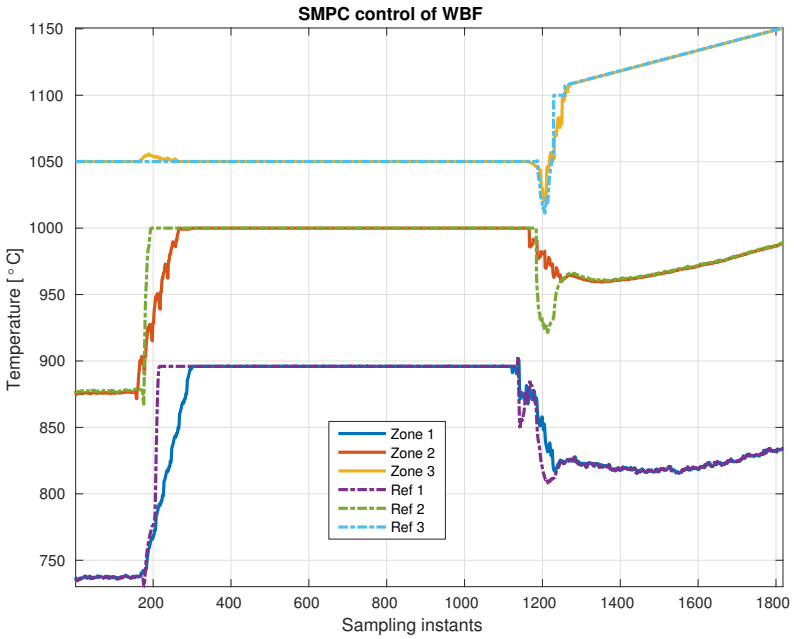

Figure 4.23: Closed-loop simulations of the walking beam furnace with a stochastic model predictive controller.

structure of Figure 4.22.

We present closed-loop simulations of the WBF with the aforementioned stochastic MPC controller in Figure 4.23 where we see that the three controlled temperatures track the prescribed set-points with adequate accuracy. We selected these set-points so as to operate the furnace in a similar manner and around the same operating points as in the experimental setting.

Then, the stochastic MPC problem will be formulated as detailed in Chapter 3 using the above stage cost and input constraints. The tuning parameters Q and R will affect the closed-loop behaviour of the system as well as the unmodelled dynamics of the oxygen and pressure inside the furnace. The available data have not allowed us to derive reliable dynamical models for these variables. Regarding the oxygen, one possible reason is the inhomogeneity of the distribution of oxygen in the furnace combined with the low accuracy of the oxygen sensor. However, since these variables are not at the main focus of the control problem formulation, i.e., we do not need to steer them to specified set-point, the availability of such models is not first-priority. Instead, we will try to discover how the values of the tuning parameters Q and R affect the availability of oxygen and the pressure in the WBF.

What is easier to obtain, is a machine learning model of *predicted oxygen availability* which may be a nominal value. This is a nominal variable which is defined as

$$\langle [O_2] \rangle_{k,H} = \begin{cases} 0, & \text{if the } H\text{-step-ahead predicted average } [O_2] \text{ is in } [0,2]\%, \\ 1, & \text{if it is in } (2,5]\%, \\ 2, & \text{otherwise} \end{cases} \tag{4.47}$$

Then, the goal is to obtain a reliable classification model of the form

$$\langle [O_2] \rangle_{k,H} = \Phi(y_k, y_{k-1}, \ldots, y_{k-H_y}, u_k, \ldots, u_{k-H_u}, [O_2]_k, \ldots, [O_2]_{k-H_o}). \tag{4.48}$$

This relation can be interpreted as a very coarse dynamical model, which is, however, suitable for our purposes here.

When the penalty parameter R on fuel consumption is low, then the controller may behave more aggressively at the corresponding zone sacrificing more fuel to obtain a lower tracking error. This is then likely to drain the chamber of the furnace off oxygen. When R is instead too high, then less fuel will be consumed, but then it is likely to not reach the desired tracking accuracy.

Ultimately, what we need to achieve is a mapping of the form

$$(Q, R) \mapsto \text{performance indicator}, \tag{4.49}$$

i.e., a mapping from the tuning parameters of SMPC to the expected closed-loop performance of the controlled system. In Section 4.5 we introduce appropriate performance indicators which will effectuate the construction of such a mapping and allow for the online recalibration of the controller. A more detailed analysis will be provided in deliverable reports D2.2 and D2.3.

4.5 Key performance indicators

Key performance indicators (KPIs) need to be introduced so as to be able to assess the closed-loop performance of the walking beam furnace. Such a performance assessment can be done either *in silico*, or in practice where the system should be continuously monitored and assessed, so that corrective actions can be taken to avoid the perpetuation of an operation with poor performance — potentially detrimental either to the plant or to the environment.

We will distinguish between *level A* and *level B* KPIs. Level A KPIs will be defined at a higher frequency resolution and will be used by the process operator to monitor the instantaneous performance of the process. For example, we shall define the *running fuel consumption* which will quantify the instantaneous fuel consumption of the furnace using an average of a few past measurements (spanning a few minutes back in time). These, level A, KPIs will be used by the operators to predict poor performance which might be resulting from a technical error or human mistake. Moreover, level A KPIs will be used to recalibrate the controller so that, overall, the controlled system performs as expected.

On the other hand, level B KPIs will be used to assess the overall performance of the furnace throughout the course of the whole campaign. Taking into account the dynamics of the WBF, for all level-A KPIs we shall use a window of $5min$, whereas for all level-B KPIs, we will use a length of 1 day (or part of the day for shorter campaigns). During ordinary industrial operation, level-B KPIs will use a window of 1 day.

KPIs are used to evaluate the performance of the operation of the walking beam furnace and, being scalars, they allow comparisons between different configurations. They are, however, not the only tools with which one can evaluate the performance of the process. Visualisations of the distribution of various variables such as *fuel consumption*, *tracking error*, *excess of oxygen* and *concentrations of pollutants* can also be employed.

The *average fuel consumption* of the furnace over a horizon H is given by

$$\mathrm{KPI}_f = \frac{1}{H}\sum_{j=0}^{H-1} f_j, \tag{4.50}$$

where f_j is the consumption of fuel at time j.

The *root mean squared tracking capability* of the controlled furnace will be defined as

$$\mathrm{KPI}_t = \sqrt{\frac{1}{H}\sum_{j=0}^{H-1} \|T_j - T_j^{sp}\|^2}, \tag{4.51}$$

where T_j is the 3-vector of zone temperatures at time j and T_j^{sp} is the corresponding set-point at time j.

The quality of combustion is quantified by the availability of excess of Oxygen. For this purpose we introduce the following KPI:

$$\mathrm{KPI}_c = \mathrm{AV@R}_\alpha \left\{\kappa_1 \|[-[O_2]_j]_+\|_\infty + \kappa_2 \|[[O_2]_j - [O_2]_{\max}]_+\|_\infty\right\}_{j\in\mathbb{N}_{[0,H-1]}}, \tag{4.52}$$

where the operator $[\cdot]_+$ is defined as $[x]_+ = x$ whenever $x \geq 0$ and $[x]_+ = 0$ otherwise, $[O_2]_j$ is the 3-vector of the concentration of Oxygen at the 3 zones of the furnace at time j, $[O_2]_{\max}$ is the 3-vector of the maximum allowed concentration of Oxygen at each zone and κ_1, κ_2 are constant weights. With this particular KPI, we are interested in quantifying the extent of violation of certain bounds — in particular $[0, [O_2]_{\max}]$. For this reason, taking the *average violation* will not be indicative; sparse but significant and extreme violations (which are of course undesirable) would likely lead to a small value of KPI_c. On the other hand, taking the maximum is again not very indicative since a single and almost instantaneous violations (which could be the result of a faulty measurement or local inconsistency of the concentration of Oxygen) would lead to an unreasonably high value of KPI_c. To bridge the gap between these two extremes, the expectation and the supremum, we use the *average value at risk* with confidence level α as we have already done in Section 4.2.1.

Last, we need to introduce a KPI to reflect the extent of air pollution produced by the furnace. Typical pollutants are the CO_2, CO, NO_x, SO_x and PM (particulate matter) and especially PM10 and PM2.5 (fine particles). These pollutants are measured in the exhaust of the furnace. According to MEFOS, the pollutants which are currently measured in production are the CO, CO_2, $H2_xC$, NO_x and SO_2. The ultimate objective should be that the air around the furnace and especially at places where people can be should comply with the EU air quality standards[2] and special emphasis should be put towards complying with the EU Industrial Emissions Directive[3]

At an industry-wide level, A KPI which is typically used and can also be employed here is the *air quality health index* (AQHI) which quantifies the impact of the *current* and *predicted* quality of air (taking into account a series of factors and pollutants) to public health.

[2] See http://ec.europa.eu/environment/air/quality/standards.htm.

[3] See http://ec.europa.eu/environment/industry/stationary/ied/legislation.htm and http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32010L0075.

4.6 Modelling the combustion quality

4.6.1 Problem configuration

The combustion quality is related to the discussion in the previous section; using machine learning techniques we derived models which allow us to predict the expected average oxygen availability and estimate a running (level-A) value of KPI_c.

As it was described in Section 4.5, the Oxygen state prediction can be cast as a classification problem, involving three classes

C1 if the H-step-ahead predicted average [O_2] is in $[0; 1.5]\%$

C2 if the H-step-ahead predicted average [O_2] is in $(1.5; 5.0]\%$

C3 Otherwise

The prediction is again based on the following inputs and their time lagged versions up to a lag equal to 20 and also delayed outputs (up to 20) of the value of oxygen in the targeted zone

```
WBF_Z01_OilControl_FICHSIMV (In1)
WBF_Z01_CombAir_FICHSIMV (In2)
WBF_Z02_OilControl_FICHSIMV (In3)
WBF_Z02_CombAir_FICHSIMV (In4)
WBF_Z03_OilControl_FICHSIMV (In5)
WBF_Z04_CombAir_FICHSIMV (In6)
WBF_MainExhaust_ExhaustTemperatureHSIMV (In7)
SU_UML_GB30_SGNHSIValue (In8)
SU_IML_GB6_SGNHSIValue (In9)
```

Figure 4.24 depicts the transformation from raw data to classification labels for zone 1. From these figures it is obvious that C1 is heavily under-represented. Therefore, a *SMOTE* stage was deemed necessary to compensate for that imbalance. Moreover the imbalance makes overall accuracy not a suitable measure for assessing performance. In classification problems the most illustrative way to assess the performance of a model is through a confusion matrix

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	M_{11}	M_{12}	M_{13}
	C2	M_{21}	M_{22}	M_{23}
	C3	M_{31}	M_{32}	M_{33}

Table 4.9: Confusion matrix definition.

The (individual) recall for class i is given by the following equation:

$$\mathrm{recall}_i = \frac{M_{ii}}{\sum_{j=1}^{C} M_{ij}} \tag{4.53}$$

For imbalanced data sets and in the absence of a cost-matrix, the average recall, the cubic

(geometric) mean are better options compared to overall accuracy:

$$\text{recall}_{av} = \frac{1}{C}\sum_{i=1}^{C=3} \text{recall}_i, \tag{4.54a}$$

$$\text{c} - \text{mean} = \sqrt[3]{\prod_{i=1}^{C=3} \text{recall}_i}, \tag{4.54b}$$

$$\text{accuracy} = \frac{\sum_{i=1}^{C} M_{ii}}{\sum_{i=1}^{C}\sum_{j=1}^{C} M_{i,j}}. \tag{4.54c}$$

From an initial assessment of the predictive capabilities of the different input variables it can be seen that the past history of the O_2 measurement is a very good predictor for the average 10-step ahead state of the O_2 for each zone (Figure 4.25). What is is also interesting, is that the variables that hold the information regarding the opening and closing of the doors are ranked very low. Moreover as it can be seen from Figure 4.26 which depicts the normalized eigenvalues coming from the PCA of the training data for the three zones, it is obvious that the input variables are highly correlated.

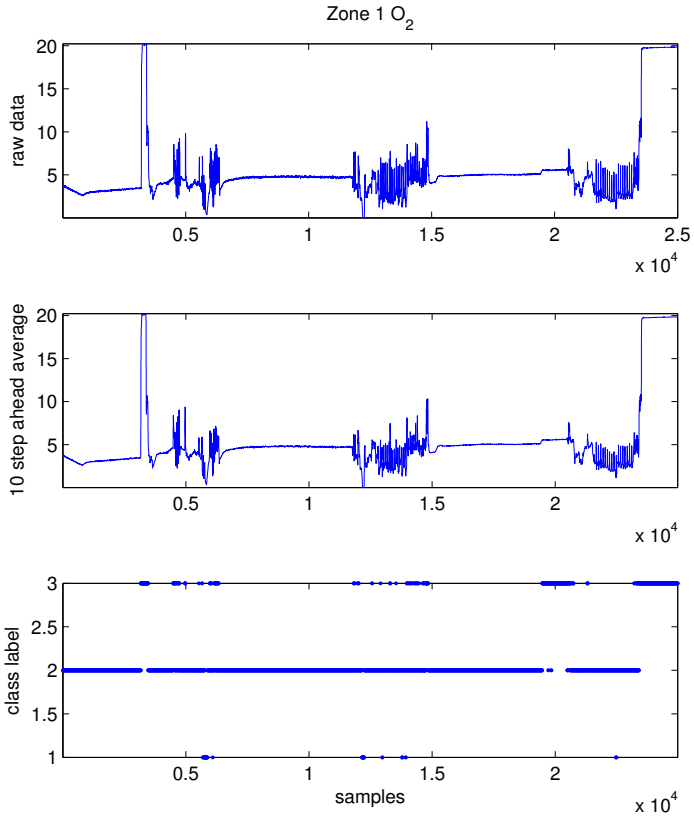

Figure 4.24: Oxygen level, 10-step-ahead average and Oxygen level as a nominal variable (low/medium/high) – data from the experimental campaign of MEFOS.

4.6.2 Dealing with imbalanced data

Synthetic Minority Oversampling TEchnique (SMOTE) procedure is a method to tackle imbalance [104]. This procedure creates synthetic instances for the minority class. The algorithm for creating a new instance of the minority class introduces a synthetic example along any/all of the lines joining that particular instance with its k nearest neighbors that belong to the minority class. Therefore during the training process SMOTE is employed to increase the training instances of C1.

For the oxygen state prediction we tested four different approaches. The first two are based on *feature ranking*, the third one based on *PCA* (principal components analysis) [105] for dimensionality reduction, and the last one uses the *ordinal classification approach* proposed by Frank and Hall [106].

4.6.3 AUC-based approach

As it was presented in the previous section, in order to reduce the number of input variables, we ranked the variables based on their average AUC and then used a kNN classifier to find the optimal number of features which we subsequently fed on an SVM. Figure 4.27 shows the performance of different features sets using as a performance measure the geometric mean, the average recall and the overall performance for the case of zone 2. As it can be seen for this case the "optimal" number of features is 6. The Following tables show the results of the classification process for the three zones using the "optimal" number of features and a nearest neighbor classifier. As it can be seen some overlap/confusion exists between consecutive classes but no overlap between classes 1 and 3.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**350**	16	0
	C2	13	**1908**	199
	C3	0	138	**7347**

Table 4.10: Results for Zone 1 using simple AUC for ranking.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**564**	8	0
	C2	10	**2100**	77
	C3	0	64	**148**

Table 4.11: Results for Zone 2 using simple AUC for ranking.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**42**	4	0
	C2	49	**5410**	699
	C3	0	388	**3379**

Table 4.12: Results for Zone 3 using simple AUC for ranking.

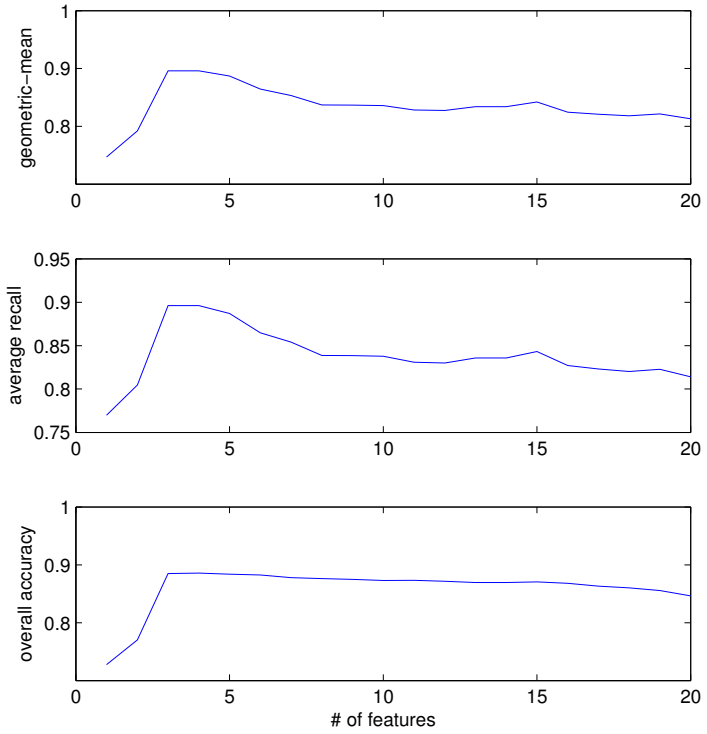

Figure 4.27: (AUC of ROC) Performance for different number of input variables using different performance measures (from top to bottom: geometric mean, average recall, overall accuracy)

4.6.4 AUC and correlation metric

The combination of AUC with the correlation measure ($w_1 = 0.7$) for ranking has as a result the rearrangement of the top ranked features as it can be observed in Table 4.13 for zone 1.

Ranking	1	2	3	4	5	6	7	8	9	10
AUC	200	199	198	197	196	195	194	193	192	191
AUC+corr.	200	40	41	199	198	197	42	196	195	39

Table 4.13: Feature ranking using AUC and AUC+correlation (results for zone 1).

Figure 11 depicts the performance of the classification scheme for $w_1 = 0.8$, which yielded the best results. The following tables also show that this scheme actually harms the classification performance indicating that good, correlated, predictors can under some circumstances improve the performance of the classifier.

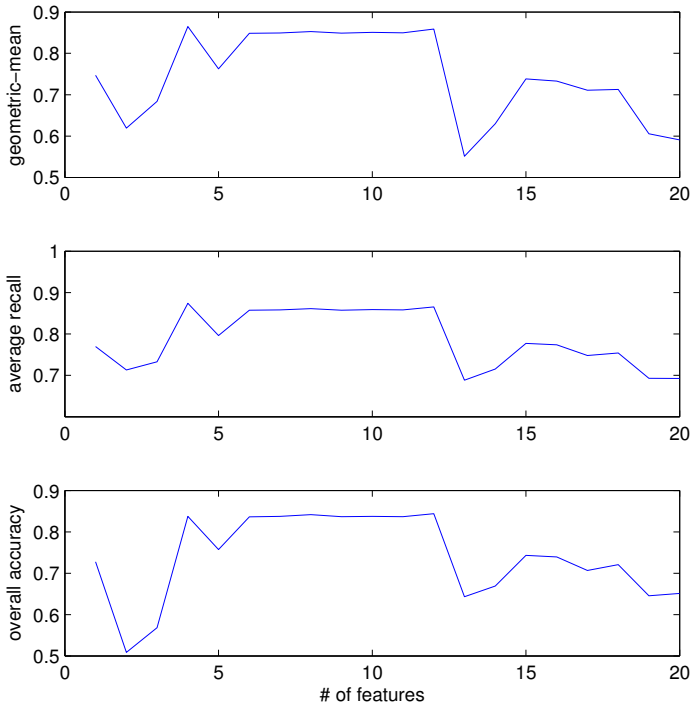

Figure 4.28: (AUC and correlation) Performance of the classification scheme using AUC+correlation for ranking.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**75**	291	0
	C2	0	**1908**	212
	C3	0	179	**7306**

Table 4.14: Results for Zone 1 using simple AUC+correlation for ranking.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**504**	68	0
	C2	19	**2009**	159
	C3	1	133	**7078**

Table 4.15: Results for Zone 2 using simple AUC+correlation for ranking.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**46**	0	0
	C2	296	**5644**	218
	C3	1	1108	**2659**

Table 4.16: Results for Zone 3 using simple AUC+correlation for ranking.

4.6.5 PCA Dimensionality Reduction

The behavior of the prediction algorithm using PCA instead of feature selection are quite different. As it can be observed from Figure 4.29 and the following confusion matrices the results are much worse compared to the previous approaches especially for class C1 of zone 1.

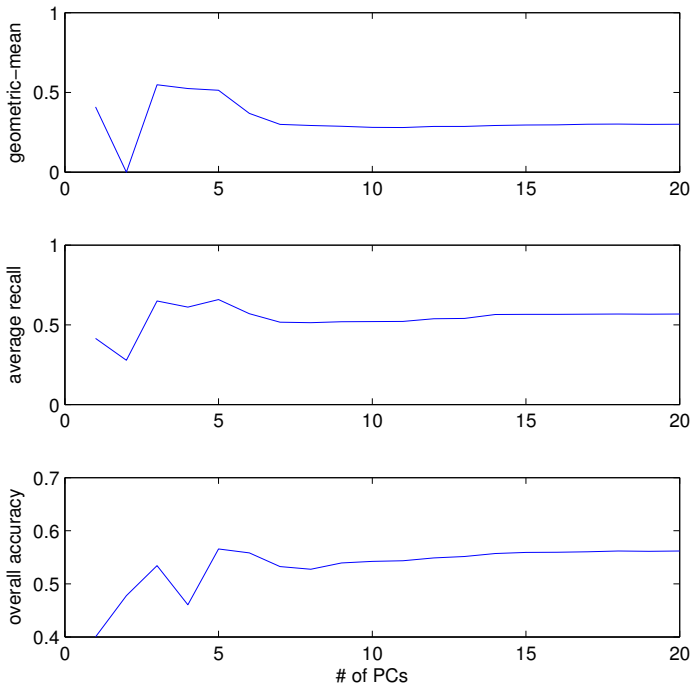

Figure 4.29: (PCA for dimensionality reduction) Performance for different number of input variables using different performance measures (from top to bottom: geometric mean, average recall, overall accuracy)

Confusion Matrix		Predicted		
		C1	C2	C3
	C1	**4**	362	0
True	C2	1	**1537**	218
	C3	0	371	**7114**

Table 4.17: Results for Zone 1 using simple PCA.

Confusion Matrix		Predicted		
		C1	C2	C3
	C1	**510**	57	5
True	C2	6	**1281**	900
	C3	0	127	**7085**

Table 4.18: Results for Zone 2 using simple PCA.

Confusion Matrix		Predicted		
		C1	C2	C3
True	C1	**46**	0	0
	C2	388	**4415**	1355
	C3	36	2865	**866**

Table 4.19: Results for Zone 3 using simple PCA.

4.6.6 Ordinal Classification

Regression is one of the most fundamental problems concerning machine learning. Many techniques such as Support Vector Machines (SVMs) [103] have been proposed in order to deal with this kind of problems. However, less attention has been paid to ordinal regression also called ordinal classification or ranking learning problem [107], where the labels of the target variable exhibit a natural ordering.

Ordinal problems are all around us. For example, student satisfaction surveys usually involve rating teachers based on an ordinal scale $\{$poor, average, good, very good, excellent$\}$. Hence, class labels contain order information, e.g. a sample vector associated with class label "average" has a higher rating (or better) than another from the "poor" class, but "good" class is better than both.

Formally speaking, the ordinal regression problem consists of predicting the label y of an input vector $x \in X \subseteq \mathbb{R}^K$ and a set of labels $Y = \{C_1, C_2, \ldots, C_Q\}$. In other words, x lies in a K-dimensional input space and y is a label space of different labels/classes. The objective of ordinal regression/classification is to find a function $r : X \to Y$ able to predict the labels of new patterns, given a training set of N points, $D = \{(x_i, y_i)\}_{i=1,\ldots,N}$ and considering an ordering of the labels e.g. $C_1 \leq C_2 \leq \ldots \leq C_Q$ where $\leq$ defines the ordering relation between the labels.

A very simple approach for ordinal classification was proposed in [106], is chosen since it only requires the preprocessing of the involved dataset without any change of the learning algorithm with the only provision that the learned model can produce a probabilistic output (a-posteriori estimates). The method transforms the original Q class ordinal problem into $Q-1$ binary class problems and uses the probabilistic values of a classifier to predict the class value. Sequentially a model is built to predict what is the probability of a given instance to belong at any of the classes that are located higher than C_1 higher than C_2 and so on up to the probability of the instance belonging to the "highest" class C_Q. After that it is simple matter to find the class with the highest probability using the following set of equations:

$$\mathrm{P}[C_1 \mid x] = 1 - \mathrm{P}[\text{class} \geq C_1 \mid x] \tag{4.55a}$$

$$\mathrm{P}[C_n \mid x] = \mathrm{P}[\text{class} \geq C_{n-1} \mid x] - \mathrm{P}[\text{class} \geq C_n \mid x], \text{ for } 1 < i < Q, \tag{4.55b}$$

$$\mathrm{P}[C_Q \mid x] = 1 - \mathrm{P}[\text{class} \geq C_{Q-1} \mid x] \tag{4.55c}$$

$$\tag{4.55d}$$

In the original work of Frank and Hall, C4.5 was the base learning method [106] for which we used the Weka free software [108]. The approach works very well for zone 1 and zone

3, with minimum overlap for the two “extreme” classes (class 1 and class 3). For zone 2 the classification scheme is not so effective, even though again the overlap between classes 1 and 3 is very small, for class 3 which is misclassified as belonging to class 2. Therefore further investigation is need for zone 2.

Confusion Matrix		Predicted		
		C1	C2	C3
	C1	**346**	20	0
True	C2	14	**1914**	192
	C3	0	148	**7337**

Table 4.20: Results for Zone 1 using orginal classification.

Confusion Matrix		Predicted		
		C1	C2	C3
	C1	**400**	172	0
True	C2	14	**2066**	107
	C3	1	4502	**2709**

Table 4.21: Results for Zone 2 using orginal classification.

Confusion Matrix		Predicted		
		C1	C2	C3
	C1	**40**	4	2
True	C2	21	**5173**	964
	C3	0	859	**2908**

Table 4.22: Results for Zone 1 using orginal classification.

4.7 Application of Control Configuration Selection Methods

A LASSO model which was trained as described in 4.2.6 has been used for applying control configuration selection methods. This method leads to linear autoregressive models as described by (4.23). For the calculation of the RGA and the PEIA described in 2.2, we first obtain the DC-gains of the system and the Frequency Response Function (FRF) formulated e.g., as a discrete transfer function. By simply rearranging the coefficients in (4.23), we can reformulate the model as:

$$y(t+1) = \sum_{r=0}^{N} A_r \cdot y(t-r) + \sum_{r=0}^{N} B_r \cdot u(t-r) \tag{4.56}$$

where $y(t)$ and $u(t)$ are the output and input vectors. To do this rearrangement, for each triplet of indexes $\{i, j, m\}$, the coefficient $a_i^j(m)$ in (4.23) is placed as the element $[A_j(t)]_{mi}$, and the coefficient $b_i^j(m)$ is placed as the element $[B_j(t)]_{mi}$. After this rearrangement, the multivariable

discrete transfer function representing the FRF is calculated as:

$$G_z(z) = \left(I - \sum_{r=0}^{N} A_r(t) \cdot z^{-r} \right)^{-1} \sum_{r=0}^{\cdot N} B_r(t) \cdot z^{-r} \tag{4.57}$$

The calculation of the RGA using (2.107) has to be adapted to discrete time systems. Meaning that for the system in (4.57), the RGA is calculated as:

$$RGA = G_z(1) \otimes G_z(1)^{-T} \tag{4.58}$$

Which for the system composed by the Oil actuators and the section temperatures results in:

$$RGA = \begin{array}{c} \begin{array}{ccc} \text{Oil}_1 & \text{Oil}_2 & \text{Oil}_3 \end{array} \\ \left(\begin{array}{ccc} \boxed{0.6199} & 0.3801 & 0 \\ 0.3801 & \boxed{1.7116} & -1.0917 \\ -0.0000 & -1.0917 & \boxed{2.0917} \end{array} \right) \end{array} \begin{array}{c} T_1 \\ T_2 \\ T_3 \end{array} \tag{4.59}$$

And the DRGA is calculated as:

$$DRGA(\omega) = G_z(e^{\mathbf{j}\omega}) \otimes G_z(e^{\mathbf{j}\omega})^{-T} \tag{4.60}$$

which for the subsystem formed by the oil combustion and the temperatures results in the magnitude in Fig. 4.30

Using the pairing rules stated in (2.2) we can conclude the following:

- The most appropriate decentralized controller is a diagonal controller with the following pairings $Oil_1 - T_1$, $Oil_2 - T_2$, $Oil_3 - T_3$. This is validated by the analysis at of DC-gains and also at higher frequencies.
- However, this controller involves the selection of a number of 2.0917, and values significantly larger than 1 are related to ill-conditioned systems. So it is expected that the system will be hard to control with a decentralized controller.

The decentralized controller configuration which has been in use at the plant prior to the initiation of the studies in DISIRE is actually the same configuration than the one suggested by the RGA as the most adequate decentralized configuration. The large values of the RGA indicate that while decentralized control is possible, the performance degradation would be large, which encourages the exploration of more complex configurations. This exploration will be performed with the use of the PEIA which was created in DISIRE and is introduced in 2.2.4.

The calculation of the PEIA for the complete model results in

$$PEIA = \begin{array}{c} \begin{array}{cccccc} \text{Oil}_1 & \text{Oil}_2 & \text{Oil}_3 & \text{In Gate} & \text{Out Gate} & \text{Exhaust} \end{array} \\ \left(\begin{array}{cccccc} \boxed{0.5346} & \boxed{8.1865} & 0.1323 & 0.0319 & 0.0604 & \boxed{7.6501} \\ 0.0001 & \boxed{54.6701} & 0.5133 & 0.0002 & 0.1892 & 0.0478 \\ 0.0000 & 1.7563 & \boxed{23.6595} & 0.0000 & 2.5628 & 0.0049 \end{array} \right) \end{array} \begin{array}{c} T_1 \\ T_2 \\ T_3 \end{array} \tag{4.61}$$

which suggests that, in addition to the connections used by the previously discussed decentralized control, an adequate controller should also consider at least the exhaust flow and the second combustion oil to control the temperature at the first chamber T_1. Therefore, a potential has been identified to improve controller performance by using a more complex control configuration like e.g. with the use of an MPC controller.

4.8 Discussion of the results

Overall, we have presented an in-depth discussion and we have presented results on data-driven modelling and predictive control for MEFOS' walking beam furnace. Our analysis involved an initial screening of the data in Section 4.1.2 with a discussion about data management and filtering, a proposal for a meaningful evaluation of the predictive capacity of models in Section 4.2.1 followed by a methodological approach to selection of variables in Section 4.2.2 using Relief and the AUC of ROC approach. We then explored the possibility of deriving models based on the physical heat-flow equations of the system which, however, did not yield fruit because of the many assumption we had to draw and the luck of appropriate experiments to build such models. This analysis was useful for the derivation of grey-box models in Section 4.2.3, notwithstanding. Then, we built and evaluated various other types of linear models in Sections 4.2.4, 4.2.5 and sparse linear models in 4.2.6 and, finally, nonlinear models in Section 4.2.8. For all these models we evaluated their predictive ability using the proposed AV@R-based approach and, more imporantly, we reconstructed the multistage probability distribution of their prediction errors (at each time stage) using the scenario tree structure which was introduced in the previous Chapter in Sections 3.1.1 and 3.1.2. We used all the above — models and scenario trees — to build an advanced scenario-based stochastic MPC controller as detailed in Chapter 3. Finally, we did not model the dynamics of the Oxygen using dynamical models — instead we proposed a machine-learning based approach which is decoupled with the modelling and control scheme wherein we associate how well a temperature-tracking controller behaves in terms of combustion quality. This required meta-dynamical-models which predict the quality of combustion in terms of the predicted average excess of Oxygen using machine learning classification models. We also introduced key performance indicators, which will be detailed in report D2.3, and are the basis for the evaluation of the closed-loop system.

As a footnote, a model based on data can be no better than the available data which were used to construct it, therefore it should be noted that the models we present here are accompanied by a *domain of applicability* which is dictated by the coverage of the training data. It is well known that closed-loop systems are much harder to identify because the system dynamics needs to be distinguished from the controlled dynamics which tends to be at steady state.

It is recommended that the furnace operates in dual mode: in *controlled mode* a SMPC controller tries to keep the temperatures at the three zones at desired set-points and the furnace operates almost at steady state and in *calibration mode*, where the controller deliberately creates the necessary variability (by administering random control actions) so that an adaptive modelling module improves the system model.

From the above analysis, it seems that the ARX models presented in Section 4.2.5 are

mostly suitable for the walking beam furnace and combine low complexity (which deems them suitable for MPC implementations) with a good predictive ability. Nevertheless, the adaptive modelling approach presented in Section 4.2.7 would be a lot more suitable for an online long-term operation of the furnace where there are more data available.

Future trials with MEFOS's walking beam furnace should be performed either in an *open-loop* fashion where the controller is completely disconnected from the system[4], or a *random error* should be added to the controller's actions or the measurements (so that the controller receives false and perturbed measurements). An alternative, if possible from a practical point of view, would be that during some periods of time, to short-circuit the controller's set-point signal with the controlled output, so that the deviation becomes zero and the controller provides no control action.

[4]However, MEFOS would not like to perform such trials because of safety considerations.

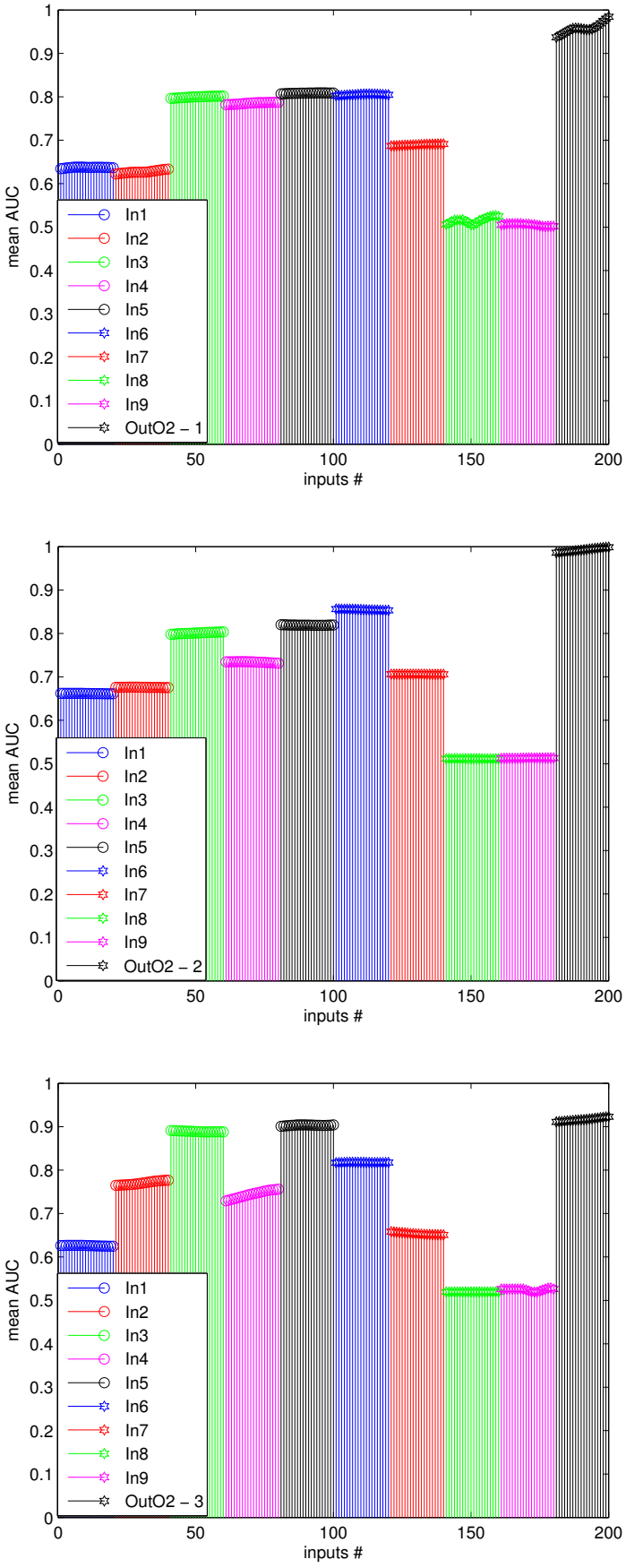

Figure 4.25: AUC of ROC for ranked feature selection at the three zones of MEFOS's walking beam furnace.

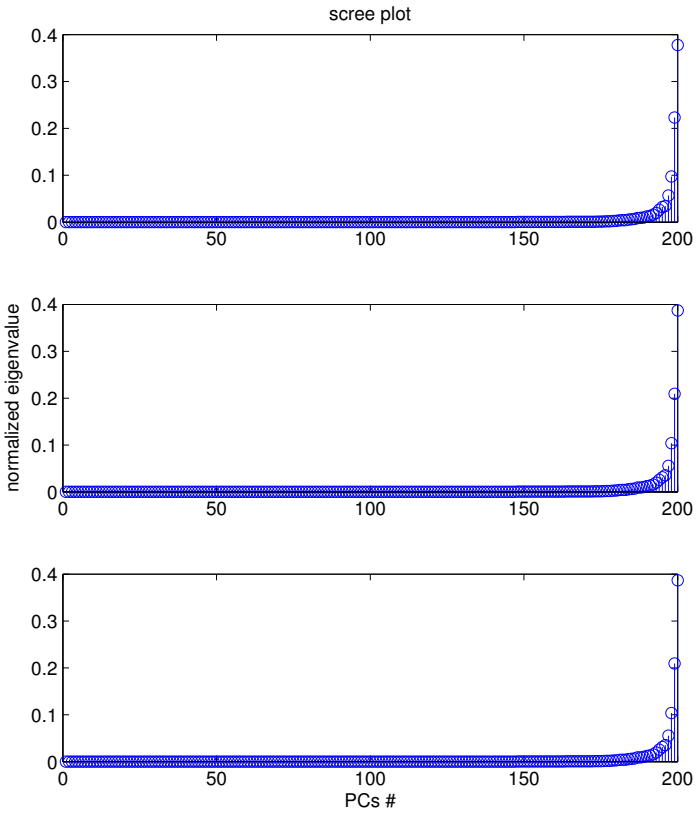

Figure 4.26: Normalized eigenvalues for the training data from the three zones.

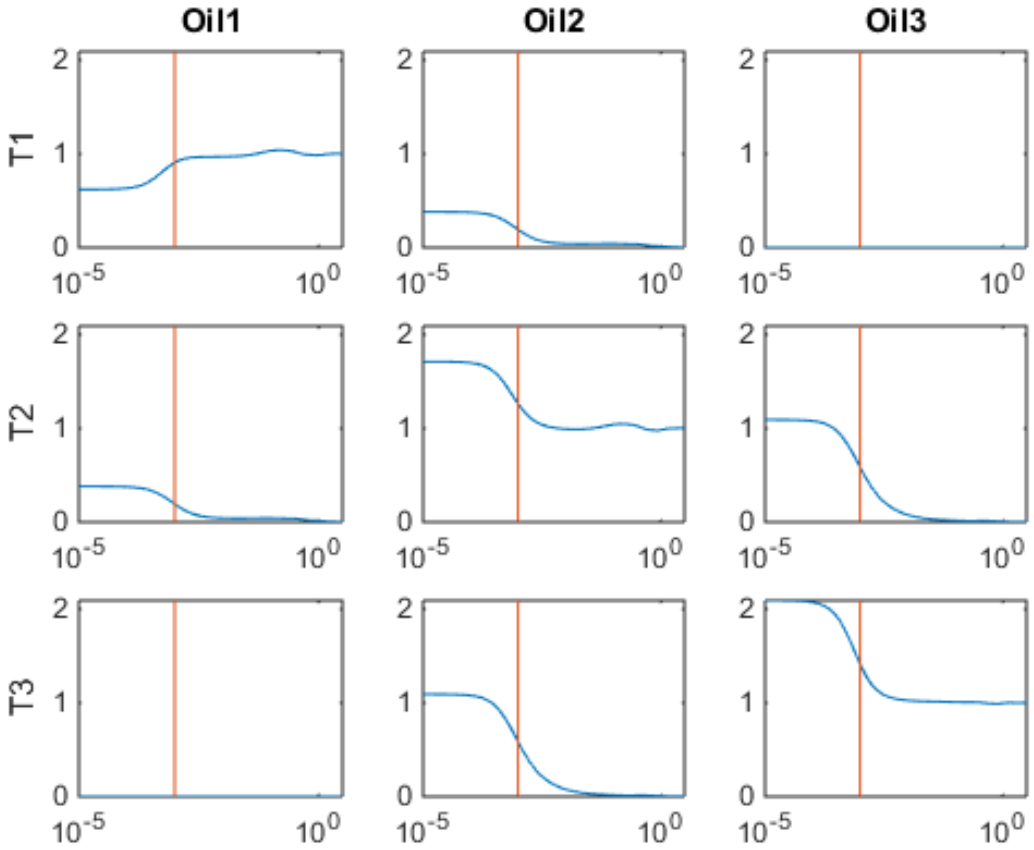

Figure 4.30: Magnitude DRGA of the subsystem formed by the oil flow rate and the temperatures.

Chapter 5

Case study: Naphtha/LPG cracking furnace

5.1 Introduction and problem statement

5.1.1 The process

We will first provide a technical description of the naphtha/LPG cracking furnace which will lead to the proper statement of a control problem. A schematic of DOW Chemical's naphtha/LPG cracking furnace (henceforth, NLPGCF) is shown in Figure 5.1.

Following deliverable reports D1.X, we have identified that the *manipulated variables* of the NLPGCF are

1. The damper positions at the exhaust of the furnace
2. The infeed valves (FG valves) which directly control the fuel supply rate

Some technical specifications as an addition to the specifications provided in deliverables D1.X are provided in this paragraph. The naphtha infeed under normal operating conditions varies between 20000 and $27000kg/h$, while, when at start-up or shut-down this value is between 1000 to $20000kg/h$. For normal operation, both limits should be treated as hard limits. The LPG

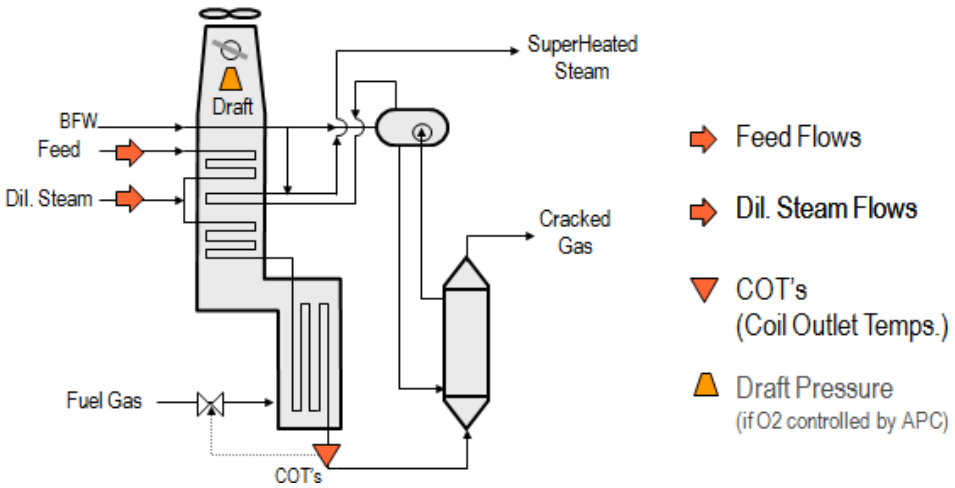

Figure 5.1: Schematic of the naphtha/LPG cracking furnace of DOW Chemical

Variable	*Units*
COT[1-8]	$^{\circ}C$
Draft[1-3]	$\mathrm{mmH_2O}$
FGflowlowrange	kg/h
FGflowhighrange	kg/h
O2[A/B]	$\%$
CO[A/B]	ppm
FGValve[A/B]	$\%$
Damper[A/B]	$\%$
WobbeIndex	$kcal/m^3$
Propanefeedflow	kg/h
Naphthafeedflow	kg/h
FGpressure[A/B][1-3]	kg/cm^2

Table 5.1: Variables of the NLPGCF and their units of measurement

(propane or butane) flow is between 16000 to $32000kg/h$ at normal operation while drops down to $1000kg/h$ during start-up and shut-down modes. When the furnace is not working, or is in off for maintenance, the naptha and the LGP flows are below $1000kg/h$ and either of the FG valves are at a position which is below 20% (shut down). When the furnace is in decoking mode, both napth and LPG flows are low ($< 1000kg/h$) and either of the FG valves is almost fully open ($> 80\%$). All other cases, correspond to unknown and likely faulty operation.

What is interesting for this application is to control the operation of the NLPGCF at normal operation, so data which correspond to shut-down or maintenance phases, can and should be neglected. The controller will be active only during normal operation. In our subsequent analysis, we have ruled out all operating phases which are not classified as “normal operation”.

Typical *coil outlet temperature* (COT) values are about $840^{\circ}C$ to $860^{\circ}C$ for LPG and $830^{\circ}C$ to $850^{\circ}C$ with naphtha (depending on the desired *severity* and *selectivity*). The COT values may vary in order to adjust the *propylene/ethylene ratio* (PER) of the furnace. The COT can be modified by changing the set-point of the PER, so, the PER control system, effectively, controls the COT.

The maximum allowed carbon monoxide (CO) concentration is $100ppm$ and the maximum allowed excess of O_2 is 2%. The excess of Oxygen should under no circumstances drop below 0% as this would inhibit the combustion and lead to incomplete burning. Values of O_2 above 5% cannot be tolerated for energy efficiency considerations, whereas, between 2% and 5% the limit can be treated as a *soft constraint* in the parlance of control theory.

Following the above discussion, we have compiled Table 5.2. For the control of the cracking furnace we may use two sets of manipulated variables: the fuel valve positions (there are two such values, namely FGValveA and FGValveB) and the damper positions (there are two dampers, DamperA and DamperB).

The control objectives for this process can also be inferred from Table 5.2: Variables COT[1-8] need to be kept as close as possible to the cracking temperature set-point. Measured variables such as Draft[1-3], O2[A/B] and CO[A/B] need to be kept within certain limits, while other measured variables such as WobbeIndex are measured and provided to the controller.

Since most of the variables of the system are constrained — on some there are imposed *soft* constraints which are not critical for the smooth operation of the process, such as the

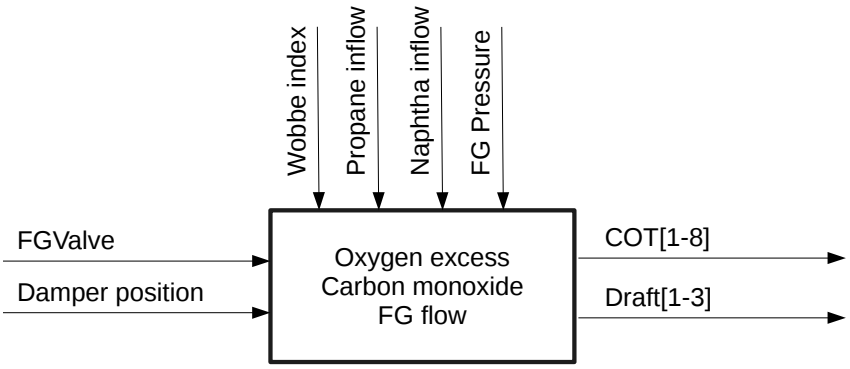

Figure 5.2: Inputs (Fuel gas valve and damper position), states (oxygen excess, carbon monoxide and flow of gaseous fuel) and disturbances (Wobbe index, propane/naphtha feed flow and pressure of gaseous fuel) of the naphtha/LPG cracking furnace.

requirement that the excess of oxygen be lower than 2% while other constraints are *hard* such as the operating range of valves. All at the same time, the plant operates under uncertainty: for example, although the Wobbe index is measured and known at every time instant k, its future evolution is ambiguous. The system dynamics is modelled by an approximate dynamical model which introduces further uncertainty into the model and finally measurements are subject to measurement noise. For these reasons, stochastic model predictive control will be the method of choice in our approach.

The control objectives which will be reflected into the choice of the MPC cost function are the following

1. Tracking of a specified COT set-point. Set-points are computed off-line. External skin tube temperature (also called *TMT* – Tube Metal Temperature) is related with COT. Due to coking process on the inner radiant coil surface, heat transfer resistance increases with time in a relation such as $T_{\text{ext coil}} = \text{COT} + (100 + 2.5\text{day} - \text{of} - \text{run})$. In order to cope with such phenomena and control the desired COT (set point) an increase in the fuel gas mass flow is required.
2. Reduction of fuel consumption to achieve the desired goal
3. Retain O_2 and CO within desired levels. The current practice is that O_2 is assigned a fixed desired set-point. This may be too restrictive as it creates a very stiff control loop. In fact, there is no desired set-point for oxygen; just an operating window
4. Keep the draft pressure (`Draft[1-3]`) within desired limits

5.1.2 Data curation

A dataset of 2.52GB and $7.1 \cdot 10^6$ measurements was obtained from DOW Chemical's DCS system and was compiled into a CSV file by G-Stat which was then uploaded to Amazon's S3 storage service. An initial screening procedure was applied to the dataset to remove data points which do not correspond to normal operating conditions and filter out entries with out-of-scope measurements (e.g., negative Wobbe index) and NaN values. The sampling time is $10s$, so the above data correspond to 821 days of operation of NLPGCF. This data was further

Variable	*Description*	*Limits*	*Role*
`COT[1-8]`	COT stands for "Coil outlet temperature". See report D1.2. There are 8 coils in which steam flows together with a stream of naphtha/propane. This is very well explained in Figure 37, p. 67 of D1.2.	As close to the set-point as possible. Typical temperatures are around $840 - 860°C$.	T
`Draft[1-3]`	Draft pressure (negative)	-9 to -4 mmH_2O; the draft pressure should not drop below -9 mmH_2O. Values above -4 mmH_2O most likely mean that the NLPGCF is in maintenance mode.	T
`FGflowlowrange`	FG low-flow flowmeter	Measures flows up to $1100kg/h$	M
`FGflowhighrange`	FG high-flow flowmeter	Measures flows higher than $1100kg/h$. Typical FG flows are roughly in the range $2000 - 2500kg/h$ with propane feedstock and $3000-3500kg/h$ with naphtha.	M
`O2[A/B]`	Excess of oxygen in $\%$	Below 2%, always above 0%. It is not acceptable at all if it is above 5%.	M
`CO[A/B]`	Carbon monoxide can be seen as a *by-product* of the combustion. See Figure 5.3.	Maximum allowed value is $100ppm$	M
`FGValve[A/B]`	Position of the HC valve in $\%$.	$0 - 20\%$: the valve is totally closed, $80 - 100\%$ the valve is completely open, $20 - 80\%$ the valve is actually used to control the flow of the incoming feedstock. Limits: $0 - 100\%$.	MV
`Damper[A/B]`	Damper position in $\%$	Position of the damper. Important note: Unlike FGValve[A/B], for this variable: $100 - 110\%$: damper is totally shut, $0 - 20\%$: damper is fully open and $20 - 100\%$: the damper is at some intermediate position. Limits: $0 - 110\%$.	MV
`WobbeIndex`	The Wobbe index is an indicator of the Higher Heating Value of the FG. Using the Higher Heating Value and the FG flow we can calculate the amount of energy introduced to the furnace. The Wobbe index changes with the FG composition.	–	M
`Propanefeedflow`	Inflow of propane in kg/h. The NLPGCF is operated either with naphtha or with propane (two operating modes), but not with both simultaneously.	During normal operation with propane infeed, there is a minimum flow of $16000kg/h$ and the maximum capacity of the plant is $32000kg/h$	M
`Naphthafeedflow`	Inflow of naphtha in kg/h.	Between $16000kg/h$ and $32000kg/h$	M
`FGpressure[A/B][1-3]`	Fuel pressure	–	M

Table 5.2: Variables of the NLPGCF. T: target, MV: Manipulated variable, M: Measured variable (e.g., output).

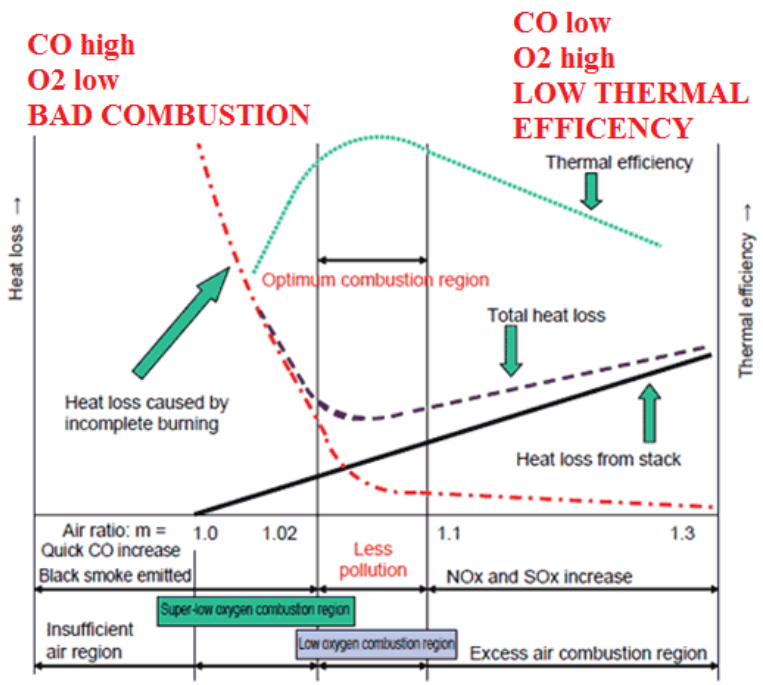

Figure 5.3: Combustion quality and the optimal combustion region.

processed to eliminate samples corresponding to maintenance mode, which is indicated by draft pressure values greater than -4 mmH_2O. On removal of such samples, the dataset was then split into multiple experiments of normal uninterrupted operation (minimum 48 hours) with reliable measurements. After curation, the dataset contained 24.34% samples of the original $7.1 \cdot 10^6$ measurements and was split into 30 experiments. For the purpose of model identification, we use data from the experiment which has longest uninterrupted operation. Figure X shows the time profiles of variables for the same experiment that has a duration of 30 days.

5.2 Data-based model identification of process dynamics

In this section we describe the approach to identify data-driven black-box models for the process described by Figure 5.2. Since for control it is desirable to have low complexity models, we use a similar approach as described in the previous chapter (Section 4.2) for identification of linear black-box models. Section 5.2.1 briefly describes the modeling methodology. In Section 5.2.2 the results obtained with the identification approach are discussed in detail.

5.2.1 Linear auto-regressive models

Approach The main idea is to construct a linear model of the form (4.23). By observing the output versus input plots, it was concluded that the dynamics is not linear and that a single LTI model cannot describe the process. However, for a short period of time the process dynamics can be approximated by a linear model. Hence, the strategy is to use a linear model identified using a certain number of past statistical data of the process and then adapt it online using

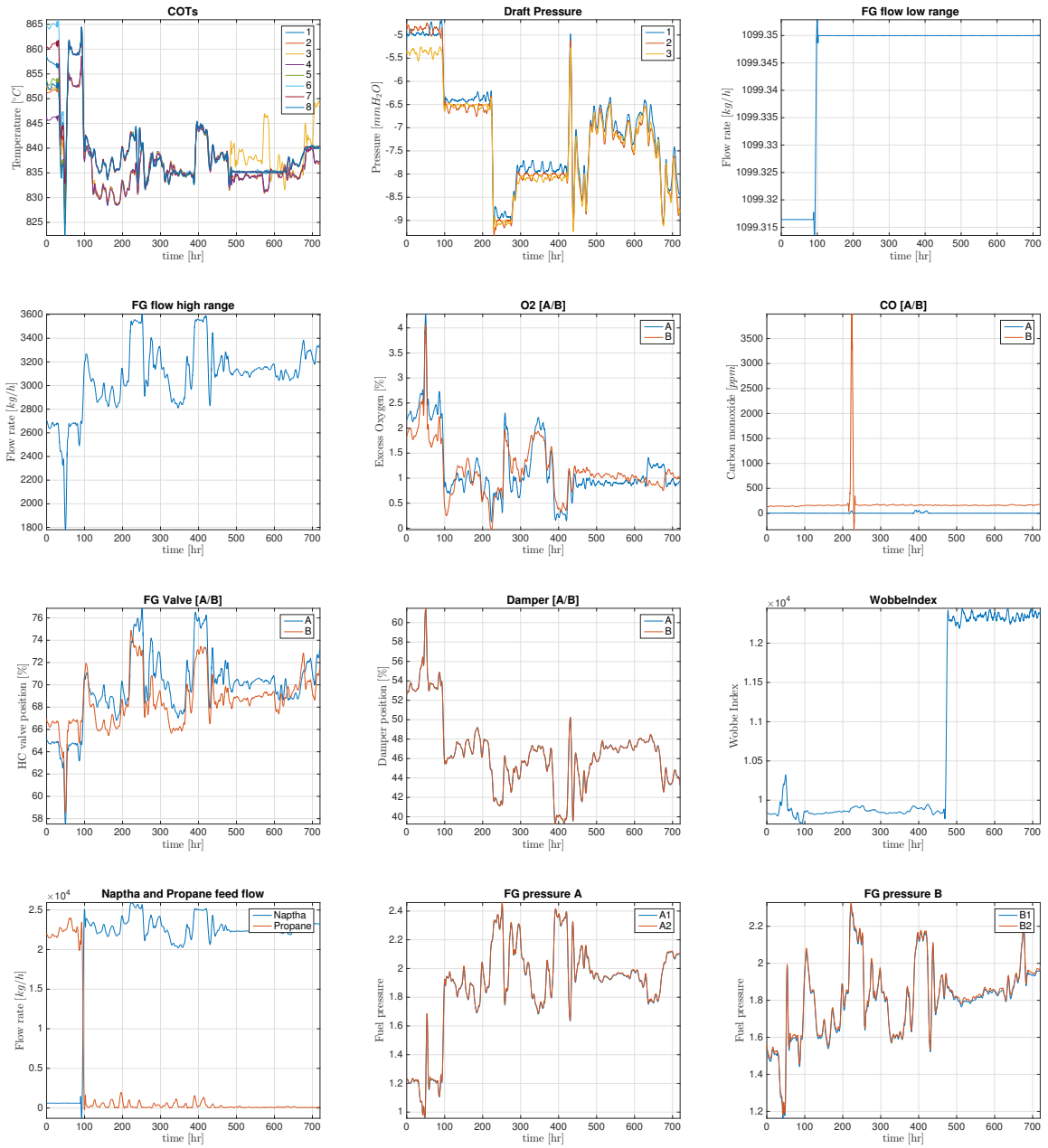

Figure 5.4: Time profiles of the measured variables for experiment having longest normal operation (30 days). For better visualization of the noisy measurements, a Savitzky-Golay FIR smoothing filter of 5^{th} order was applied to the data with frame size of 5001.

new information. From the time profiles (see Figure 5.4) of measured variables such as Wobbe index, fuel feed flow, and FG pressures; it is apparent that the variation of these parameters is slow considering the sampling time of 10 seconds. We assume that the influence of these parameters which do not vary much for a short period of time can be neglected.

Moreover, we assume that the COTs are not influenced by damper positions, and the draft pressures do not vary w.r.t. the FG valve position. This implies that we can identify two models, one for COTs as outputs and valve positions as inputs whereas, the second model describes the draft pressure dynamics with respect to damper positions. These two models being dynamically decoupled, they can be cascaded into a single prediction model which would include all the desired input-output variables described in Figure 5.2.

Identification technique For identification of the model described above, we use the least-squares algorithm developed by ODYS which identifies an ARX model from input-output data. The identification algorithms developed by ODYS also include the option to identify sparse models (Section 4.2.6), particularly suitable for large-scale processes. A key feature of the algorithm is that it can process large sized data for identification using batch approach efficiently through a recursive least-squares implementation based on linear algebra techniques. Furthermore, the recursive algorithm developed can be used for efficient real-time model adaptation by using new information without storing the past values of input-output data.

5.2.2 Results

In this section we present preliminary results based on the approach discussed above. We first identify a model using past one hour data (360 samples) and then re-identify it with a batch approach by using 180 new data samples and forgetting initial 180 samples from the previous batch. These values (360, 180) are chosen rather arbitrarily for the purpose of demonstration. In general, the number of samples used to initialize the recursive modeling and quantity of data to be used in the new batch are flexible values which need to be tuned while considering the quality of predictions. For both the ARX models (model 1 – COTs, model 2 – Draft pressures), we tune the structure to have dependency on 5 past output and 3 past input values. The quality of the adaptive model is evaluated by the PRMSE criterion described in Section 4.2.1. Using first 350 hours of data (700 batches) shown in Figure 5.4 (without smoothing), we obtain mean PRMSE values shown in Table 5.3 for model 1 and model 2. Figures 5.5 and 5.6 show the PRMSE values with respect to time for different values of K (number steps-ahead prediction).

Table 5.3: Summary of system identification results for K steps-ahead prediction

K	Mean PRMSE_1 ($^\circ$ C)	Mean PRMSE_2 (mmH_2O)
10	11.6222	0.4843
20	9.4924	0.4796
50	7.6161	0.4858
100	6.7868	0.5030

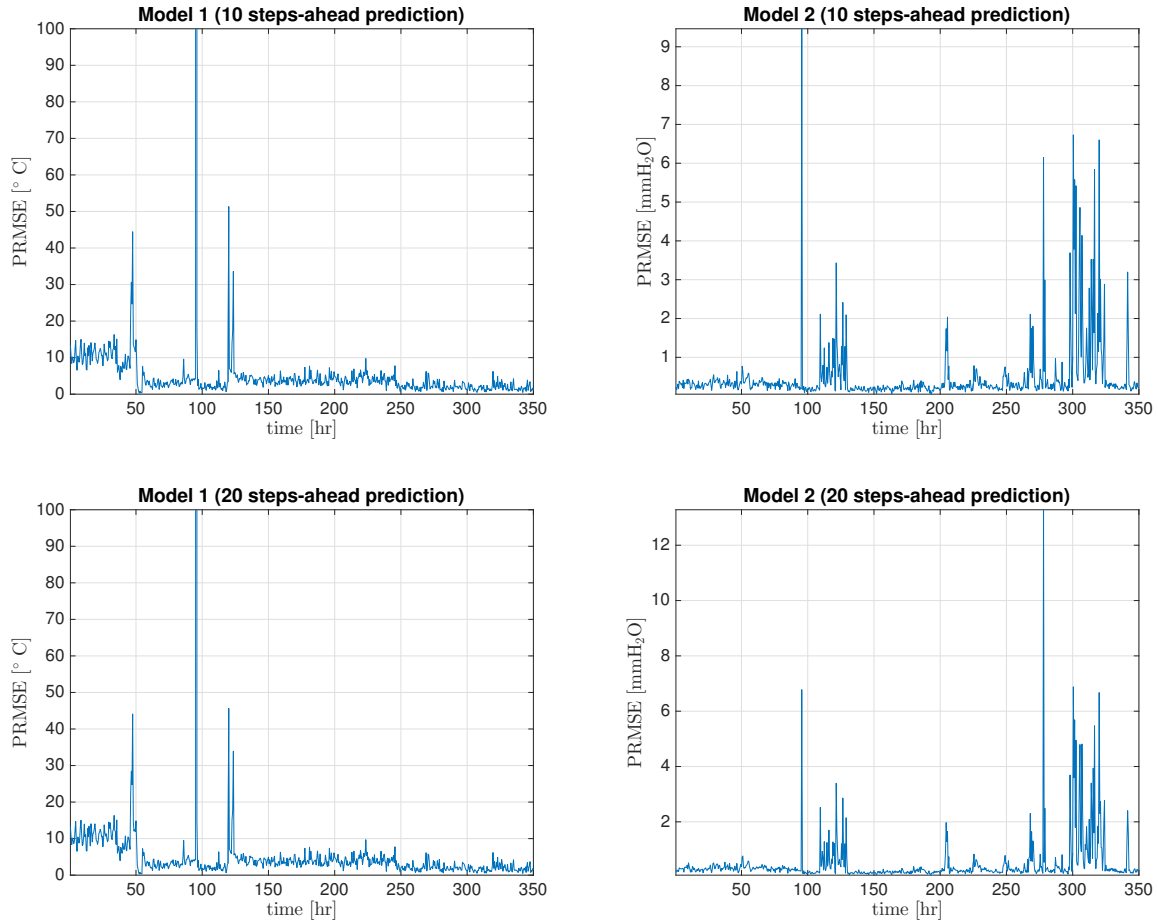

Figure 5.5: PRMSE values with batch approach for K steps-ahead prediction

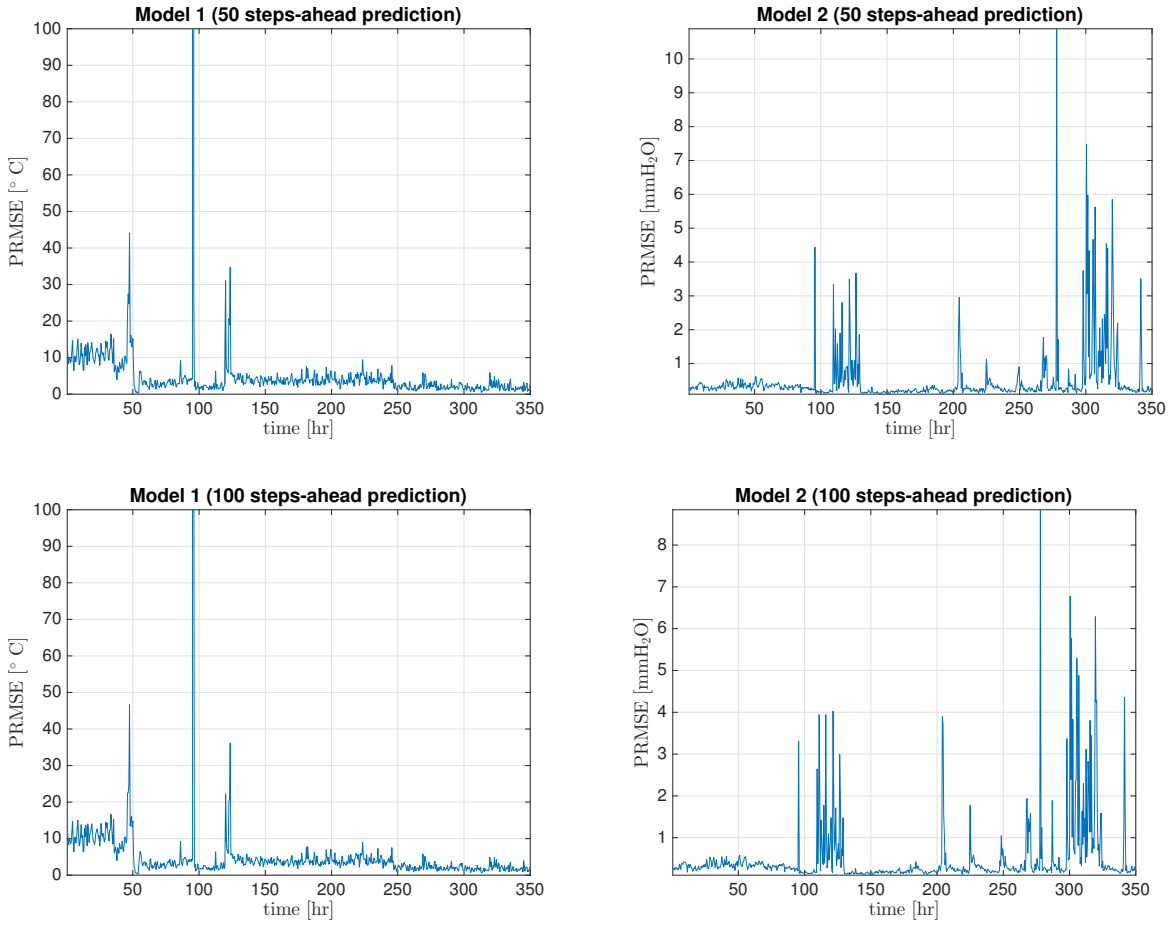

Figure 5.6: PRMSE values with batch approach for K steps-ahead prediction

Discussion The mean PRMSE values in Table 5.3 indicate that the identified models can predict as well for 100 steps-ahead without loss of quality as compared to 10 steps-ahead predictions. This is rather surprising, especially for the case of COTs as we notice better predictions with increase in K. This is clarified on observing Figure 5.7 where we notice that the values of COT do not vary appreciably even for 100 time steps and the resulting predictions are not highly accurate due to the noticeable offset. The same was observed for models identified with other batches of data. One reason for this is that the given data corresponds to measurements of a closed-loop controlled process so the output variables track the set-point. This means that the data is not rich in information about the process dynamics even for time periods as long as 1000 seconds. However, the data used to identify draft pressure model was better in quality (more variation) resulting in a comparatively better model as illustrated in Figure 5.8. The plots of PRMSE values shown in Figures 5.5 and 5.6 indicate that a linear model suffices to capture the dynamics at majority of time instants. However, the presence of some spikes in the plots indicates that there is scope of improvement. The adaptive modeling strategy with cascaded structure and neglection of some measured variables discussed earlier can be concluded to be suitable as the prediction errors were in acceptable range considering a stochastic control strategy. This results in simple linear models for control as desired. In general it can also be concluded that the closed-loop data needs to be sufficiently exciting besides having a suitable model structure in order to obtain linear models of good quality.

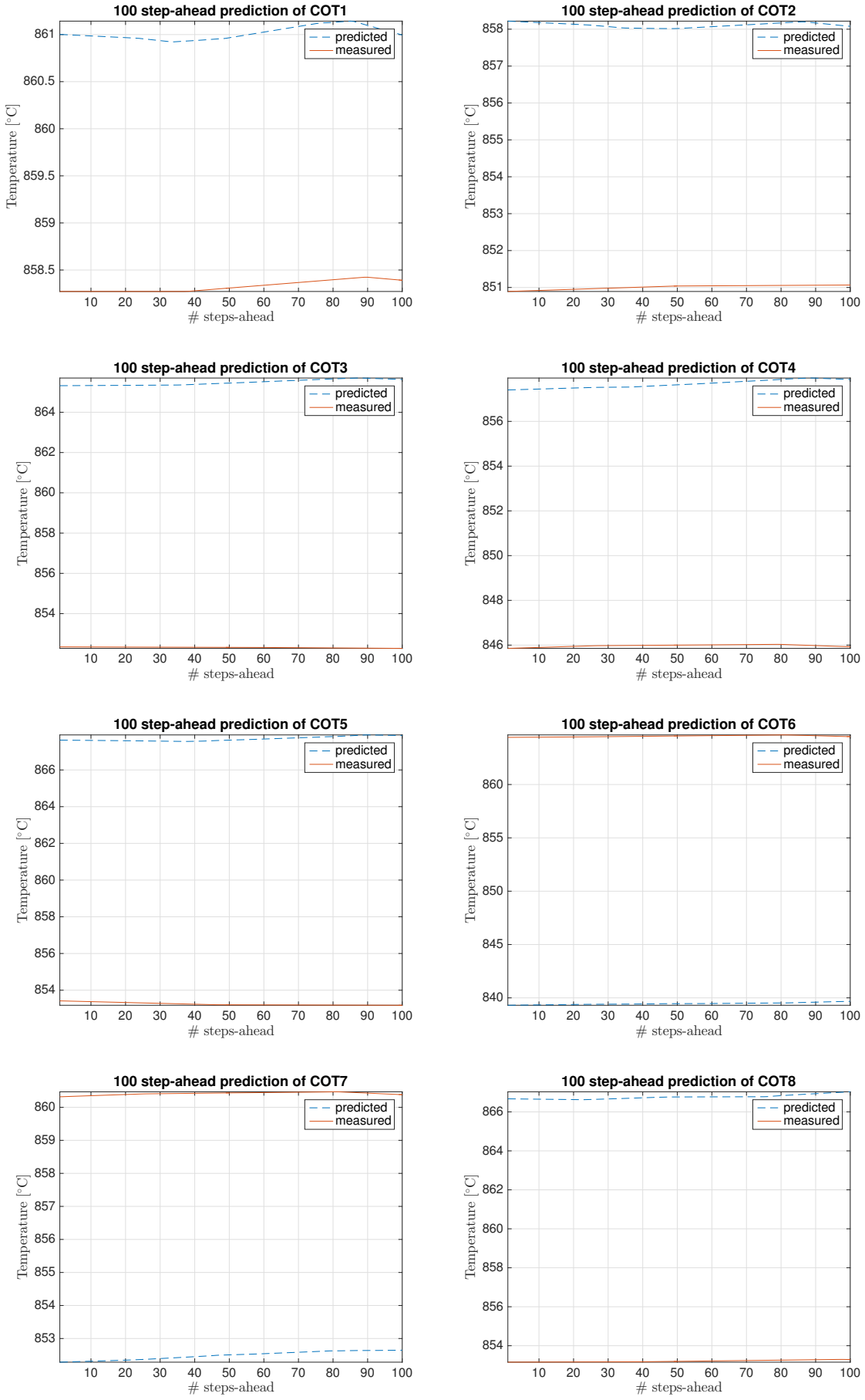

Figure 5.7: Validation of model 1 identified with first batch of data (PRMSE = 13.4687)

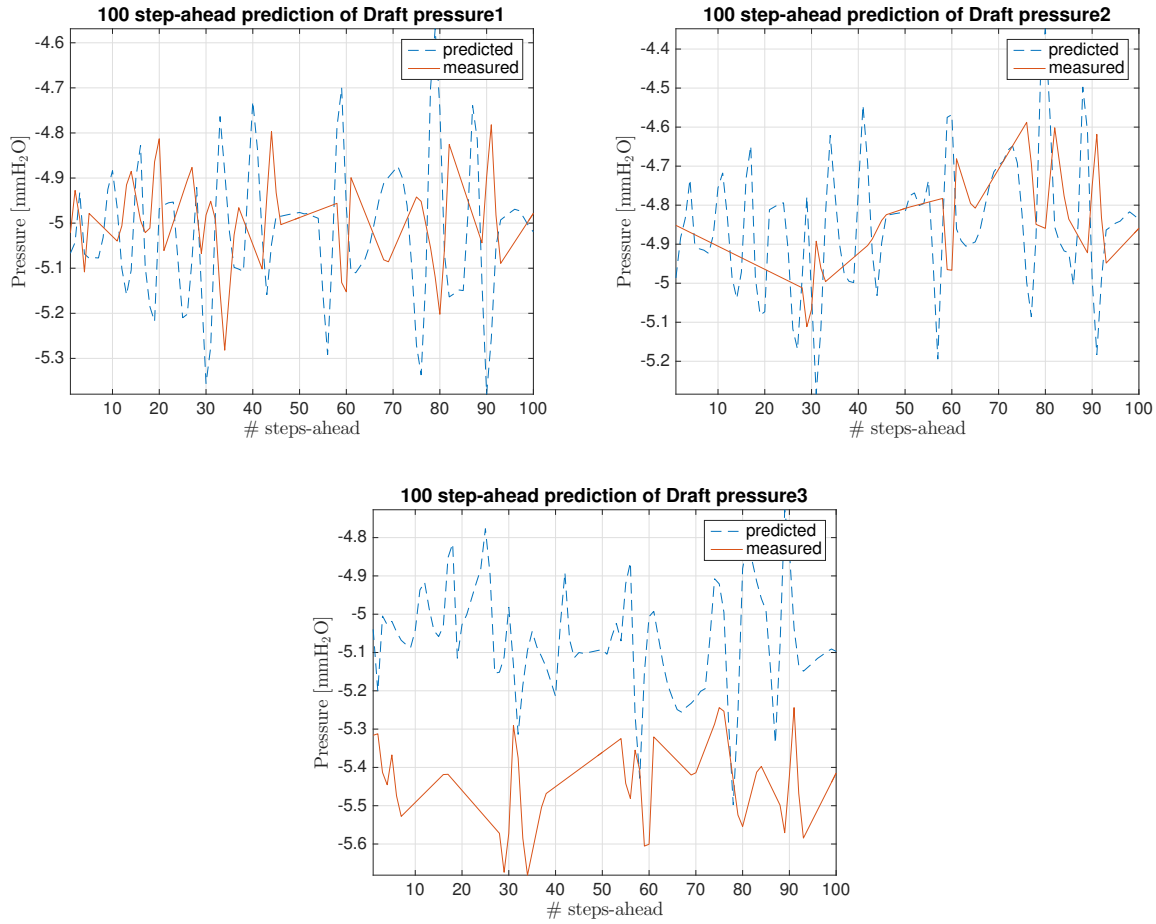

Figure 5.8: Validation of model 2 identified with first batch of data (PRMSE = 0.2847)

Recursive approach The cracking furnace data is subject to noise and changes in environmental or operating conditions with time. In order to tackle the influence of such factors, it is essential to recursively identify the model at each step with higher weight on recent data. We consider this approach for future work in order to obtain better results where the plan is to implement the newly developed recursive algorithm.

5.3 Conclusions

The modelling and control of this process is more challenging than the walking beam furnace, mainly for two reasons. First, because of the size of the dataset, batch processing methods on the whole dataset are prohibitive as they require a tremendous computational power. Second, the data are real process data and, therefore, are of lower quality with missing values that have to be filtered out and worn out, faulty or generally more fluctuating sensors creating the need to preprocess the data to filter out outliers and faulty measurements. The process is operated normally in two operating modes: with naphtha and with LPG as input feedstock. At the same time, the furnace may be either working, or in operating mode, so we cannot arbitrarily choose some data (or even all the data) to build a process model. Last, and most important, the available data are closed-loop data and the system dynamics it to a great extent concealed by the (MPC) controller; as a result, the derived models cannot be considered reliable. WP2's proposal is

to run open-loop experiments and other experiments suitable for system identification (e.g., to keep constant some inputs for some period of time – see the discussion in Section 4.8.

Chapter 6

Case study: Network of conveyor belts

6.1 Introduction

6.1.1 General

The customer order to customer delivery of the mining is in theory a long, but in many cases not too complicated process to model and control, although there are certain obstacles which still make at least part of it challenging to control. The main reason for this is that the automation level (limited integration between control systems, availability of sensors and a single access point for data etc.) in the mine is rather low compared to other process industries including the mining flotation process which has a normal standard of automaton maturity. In this chapter the focus is only on the mine process from the *face*[1] where the ore is mined to the mill just before the flotation. The flotation might be excluded, but it is nevertheless the first customer of the ore produced in the mine.

Coming back to the mine and the control of the material flow from the face to the mill, there are semi-continuous ongoing ore producing blast cycles at the face or rather in the drifts of the KGHM Lubin mine. Face location, time and ore mass from the blasts are considered as *inputs* to the modelled process, but the control of these variables is not within the scope of the current study. The reason is that the KGHM Lubin mine is a complex and big mine with many faces, equipment, resources and activities ongoing at this stage, and it is too much work to take scheduling of the blast cycles into account.

Also, it is a very large project to replace current *blast scheduling*[2] in KGHM with an automated one in a mine of this size. Therefore, it is not possible to be able to take over the blast scheduling of KGHM within the scope of this project, although how the schedule influences the material flow in mines in general will be studied. ABB have made some initial simulations based on a smaller mine in a previous project, where the blasts were scheduled together with the ore transport out of the mine, and could hence be considered as free parameters.

The ore grade is varying in the mine at KGHM, and it can be divided into three different lithological structures (sandstone, dolomite and shale). We will come back to this in the control

[1] In mining, the surface *face* is where the mining work is advancing. In surface mining it is commonly called pit face, in underground mining a common term is mine face. *Face equipment* is the mining equipment used immediately at the mine face used for removal and near-face transportation of the material: cutting machines, loaders, etc.

[2] Which typically consists in drilling, blasting, ventilation, scaling after blasting, bolting, concrete sparying, etc.

problem once the ore is mined at the face, it is transported with *LHD* (load, haul and dump) to a *screen*[3] at the dump location above the conveyor belt. The screen is used to stop the ore chunks above $40cm$ from falling into the conveyor, and to enable hydraulic hammers to crush over-sized ore chunks into smaller ones that can pass the screen into the conveyor belt. It is hard to know the origin of the ore on the conveyor belts; there are many dump locations at the conveyor belts, the conveyor belts are running continuously and many locations are mined simultaneously. We will come back to what is needed to track or know this in tests or in future automation systems. The ore is transported over several conveyor belts on its way to the mine hoist where it is skipped to the surface.

The conveyor belt system transports the ore from the hoist to the main separation screen where the ore it is divided in two flows, one with ore finer than $25mm$ that can go directly into the ball mill, and one with larger pieces that needs to be crushed first.

It is possible to measure the electric current on the electric motors that are driving the conveyor belts when the ore is transported on the conveyor belt system. The electric motors are operated at known constant speed. The mass of the passing ore is measured at some locations along the conveyor belt system. The ore flow could be quite easy to predict and follow through the system, if there were no places where the ore could be stored. There are several places like bunkers and other storage facilities along the transportation line where it is difficult to predict how much ore is stored, how the incoming ore is mixed and when it will leave the storage. The Lubin mine does not seem to have any level measurements at the bunkers. The bunkers are just storage buffers between the conveyors that act as buffers for the material flow. If there is a stop before or after the bunkers, they can be used to make the rest of the system less sensitive to disturbances until technical problems are solved.

The bunkers pose the main challenge towards modelling and controlling the material flow. This is because there is a limited number of sensors in the bunkers so it are not often — if at all — any level measurements at the bunker and since the ore flow before and after will include some error there will be an accumulated error in the determination of storage levels that will cause the modelled level and the true level to drift apart resulting in overflowing or empty storages. This in turn could lead to shut-downs affecting the entire material flow in the mine. The ore level in a bunker has a maximum level for obvious reason. At some bunkers, the ore level should not be too low, to avoid the bunker output feeder from getting damaged by incoming ore. Therefore, some minimum ore level has to be maintained, to maintain a layer of ore acting as a damper from the incoming ore.

Tracking the ore levels at the bunkers is a general control problem although now we are still considering the ore as homogeneous, i.e., every piece of ore is of the same quality, which is not realistic. Each ore piece is different and has different content, but this granularity level is not possible to track in the mine. In the mine realistically each blast could get an individual identity, due to the mixing at the blast and since the *block model* probably will not have higher level of detail. Thus if this level of detail is needed in predicting the ore through the conveyor system, the models need to keep track of the ore in the bunkers. This would be easy if the bunkers would

[3]Mechanical screening, often just called *screening*, is the practice of taking granulated ore material and separating it into multiple grades by particle size.

behave as a first-in first-out container, but they do not. We will come back to this later in this chapter when we describe the more advanced storage models developed. In our first modelling approach we have used FIFO models to get the system up and running.

The *block model* is a model that describes the ore content in the mine and the mine has a prediction of the ore content in every blast based on drill holes made in the prospection phase and with better material tracking these models will improve. The hoist is the simplest, yet the most expensive part in the transportation, and it takes long time to increase the number of shafts, so this is the bottleneck of most mines, and this is the main reason why the bunker levels as well keeping the conveyors running is very important. Since the mine hoist is operating continuously, it is crucial that the ore flow is constantly coming to the mine hoist. Lost production in the hoist cannot be gained back.

To make sure that the models developed in the project are as generic as possible there have been and will be examples of mines with trucks and trains in the modelling, simulation and control evaluation.

The idea has been to first make a basic mine model for the simulation framework and then during the project these models will be extended and improve based on real data and measurements from the Lubin mine as well as simulations results from the developed advanced models. In the second half of the project different cases and mines will be simulated and evaluated. Based on this in the end of the project there will exist control concepts that could be possible to implement in a real mine. The goal is to reach as close as possible to some MPC solution.

6.1.2 Underground transportation system in KGHM s.a. mines

The transport systems used in the copper mines were chosen and designed for the room and pillar mining system. Taken into consideration were factors such as the methods of mining and loading, the properties of mined material, the ore flow characteristics, and the required capacity and length of transport routes.

The characteristic feature of the horizontal transport in the three KGHM mines is the use, at the various stages and on a different scale, of both cyclic transport (LHDs, trucks and rail transport) and the high-capacity continuous transport (belt conveyors). The transport function within mining panels in all the copper ore mines is carried out by LHDs feeding the ore onto trucks. The trucks transport the ore to mining panel reloading points where it is dumped directly onto a screen functioning as a classifier. Oversize lumps left on the screen are crushed mechanically by hydraulic hammers controlled by an operator who monitors the dumping proceeds. The screen is situated over a chute that feeds the belt conveyor via a vibrating feeder. The feeder discharge panel is positioned $60cm$ above the belt so the impact of material falling on the belt is mitigated. Up to several loading points with either fixed or vibrating screens can be set up along the route of a single conveyor.

Ore leaving mine panels is usually stored in panel storage bunkers that have a form of blind shafts. The panels can have either one storage bunker, as is the case of Lubin mine, or one bunker may be servicing two or more panels (Rudna mine). In the Lubin and the older part of the Polkowice-Sieroszowice mine, the primary transport from the panel bunker to ore loading pockets that adjoin the shafts is provided by conveyors or by rail. Rail car loading

stations are used for conveyors to load the trains and car dumps are used to unload trains in the shaft area. The latter are either of tipple type (Lubin and Polkowice-Sieroszowice mines) or bottom dump cars are used (Polkowice-Sieroszowice mine). Rudna mine uses only belt conveyor transport. The newer part of the Polkowice-Sieroszowice mine has a mixed primary transport system where it is possible to transport ore from panels to shaft both by rail or a series of belt conveyors. The mine also has a system of belt conveyors that allows to transport ore to the adjacent Rudna mine.

Surge stockpiles, whose purpose is to provide an efficient interface between the continuous and cyclical transport system play a major role in transportation systems. There is a need to store output in stockpile to average the erratic flow of the ore coming from mine panels and to match the transportation capacities of various links in the transportation system, many of which operate at different output levels and at different times. Stockpiling material in panels and by the shaft loading stations makes the panel operation independent of that of shaft, and both of these independent of the main transport system. As a result, it is easier to organize the work of all the links in the ore transport system and to utilize the equipment to its full capacity.

Short breakdowns of individual links of the transportation system do not interfere with operation of other links and do not restrict system capacity. Construction of bunkers and stockpiles, although costly and time-consuming, is justified by benefits it brings to organization of work and to utilization of workforce and equipment. As a result it improves the overall economics of mining operations.

6.1.3 Technical specifications

The whole analysed underground BC system of the Lubin mine is presented in Figure 6.1. The system consists of 3 main branches:

1. Left branch (conveyors: N465 – T1, L44 – L41 and the railroad leading to the T2 conveyor). The T1 conveyor supplies the ore to the one 1000 t shaft bunker that feeds the skip. Notes: At the moment only the conveyor line N465 – T1 operates. The second transportation line consisting of belt conveyors line L44 – L41 and the railroad has not operated for the last 6 months. This means that the ore is supplied to the shaft R1 only by conveyors.
2. Right branch (conveyors: L1031 – P1 and its subbranches: C1701 – C106, L910b – L142, M41a – L52). The P1 conveyor supplies the ore to the second 1000 t shaft bunker that feeds the skip. There are no switches between the shaft bunkers nor the possibility of enlarging their capacity.
3. Bottom branch (conveyors AS34/7 – A34/1, A15, S2A – S2, S310, S320 – M21, E3 and the railroad to the shafts).

The work of the conveyor belt system is managed with regard to the following constraints and rules:

1. There is a serial connection between an ore bunker and belt conveyors – a conveyor belt supplies a bunker which then supplies the consecutive conveyor belts, there is no bypass of this connection,
2. An ore bunker can be either opened to supply the next conveyor belt or closed; no partial opening is feasible,

Belt ID	*Length (m)*	*Speed (m/s)*	*Head pulley to drive station*	*Weigth to head pulley*	*Loading points to head pulley*	*Bunker to head pulley*
P–1	300	2.5	39.5	X	X	X
P–2	820	2.5	25	X	X	X
P–3	990	2.5	11,5	X	P-36a – 948.5	X
P–4	700	2.5	13	X	X	b1 682.5 b2 691
P5a	350	2.5	250	X	X	X
P6	1020	2.5	225	175	970	X
P7	1250	2.5	50	X	X	X
P8	1350	2.5	15	X	I. 465 II. 770 III. 1225	1340
L142	950	1.94	150	30	I. 400 II. 675 III. 980	X
L910b	750	2	165	X	I. 400 II. 750	X
P9	1350	2.5	150	800	755	X
L52	1300	2	170	X	I. 340 II. 590 III. 1300	b1 870 b2 1160
M41a	1150	2.5	225	X	I. 375 II. 675 III. 875 IV. 1150	X
L413	830	2	175	275	830	X
A183	710	2.5	100	55	710	X
L161	1250	1.94	50	X	I. 650 II. 850 III. 1250	X
L1031	800	2	75	X	I. 410 II. 800	X

Table 6.1: Key data of the belt conveyors of the right branch of the Lubin mine BC system.

3. The shaft ore bunkers are controlled by shaft operators, the bunkers inside the conveyor belt system are controlled by a foreman on the spot,
4. A division ore bunker is usually open (not filled up) to maintain the continuous flow of ore and closed only in case of stoppage the conveyors succeding the bunker. This allows mining divisions not to stop supplying the ore onto the loading points of preceding BCs,
5. The above rule does not apply to the large “East” bunker (4500t) that supplies the P4 BC. This bunker is used for systematic storing the ore since Monday till Friday which is then released during the “non-mining” shifts in a weekend (when mining divisions do not mine the ore from mining panels).
6. The actual level of ore in a bunker is controlled by a gauge which automatically stops the preceding BCs when the ore reaches the top crest of the bunker. Practically (including the “dead” margins in the bottom and the conical shape on the top formed due to the angle of repose) some 80% of the theoretical bunker capacity can be used,
7. The division BCs usually operate since the first loader or truck discharges ore onto a screen in the loading point up to the end of the shift. There were undertaken tests of temporary switching off the BCs but the savings were considered to be smaller than the decrease of ore supply involved.
8. Moreover any stoppage of not fully emptied BC means that it has to be then started when loaded which raises the start-up, dynamic belt tensions and can cause premature fatigue of belt splices. Therefore the staff responsible on BC maintenance was reluctant to implementing the strategy of switching off and on partially loaded conveyors.
9. BCs are switched on and of in cascades. A cascade consists of all BCs between ore bunkers. All BCs in one cascade have to be switched off. The preceding cascade can operate until the bunker between them is fully filled.
10. The start-up procedur of a BC takes approximately 30 seconds.
11. Conveyors that are supplied by a bunker and another BC (like P4) can be supplied simultaneously by both sources. In case of the risk of overloading, the priority is given to the preceding conveyor over the preceding bunker.
12. The shaft bunkers are considered to be too small and provide the limited storing capacity.

6.2 Data collection

Ideally all sensors and data needed are available for identifying and tracking the material flow, but this is not the case in practice. The first basic simulation framework and models are up and running in Matlab and the next step is to obtain actual data from KGHM to tune and improve the model to fit the real KGHM Lubin mine. A complementary to the existing real data in the mine could be to conduct RFID tests which would improve the success probability of material flow modeling. At this stage the tests presented below are not confirmed.

There is a number of tests using RFID technology to track the ore that could be performed to get valuable input data to the evaluation of the material flow in the mine. These measurements need to be performed if one needs to validate the models and the material flow in an accurate and detailed way.

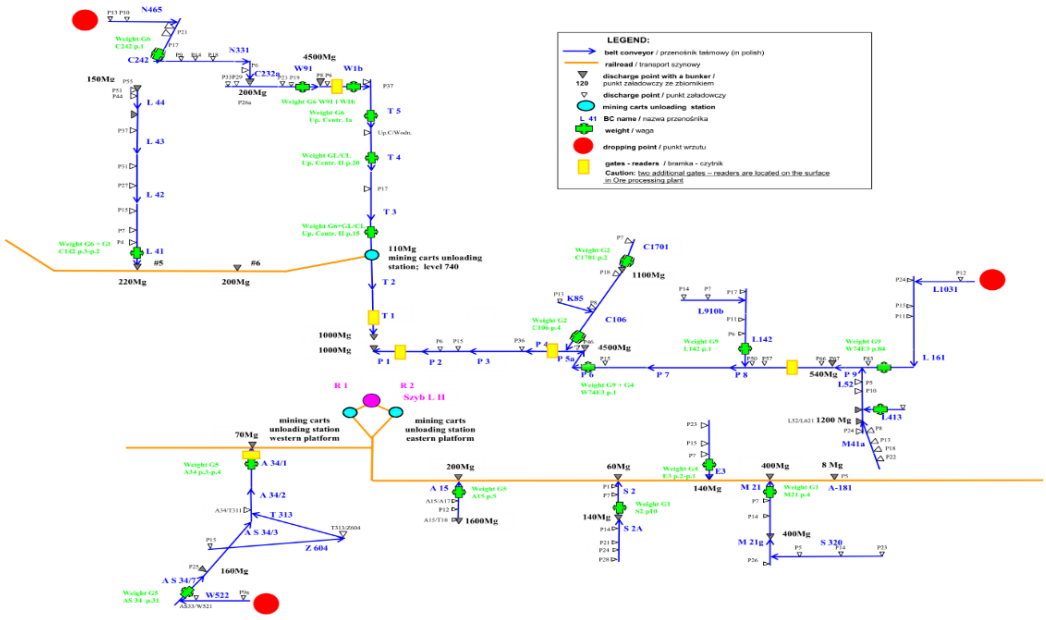

Figure 6.1: Schematic overview of the Lubin mine with planned RFID drop positions as red circles and planned RFID identification antennas as yellow rectangles.

1. RFID tags will be sent along the conveyor belt system from a few selected positions (at least one), together with the ore when it enters conveyor belts at the dump places, and track by antennas their way to the flotation. One RFID sensor/antenna after each storage would give a good picture of the entire material flow as well as the duration in the bunkers and some rough estimation of the bunker levels.
2. (Alternative to 1) RFID tags will be sent along the conveyor belt system from a few selected positions (at least one), together with the ore when it enters the conveyor belts at the dump positions, and tracked by one or more antennas overground close to the flotation.
3. One test to validate only the storage models in the bunkers above ground
4. (Extension of 3) A second step would be to perform more extensive tests on a single storage where RFID could be placed in many locations on the surface of feed into the storage and then the antenna would be just outside the storage, this would give better input of the behaviour of the mixing in the individual storage and this could be valuable in identifying storage mode and the mixing properties.

Tests above ground are more likely since this requires less resources and permits. The basic models are already in place, but it is expected that the models need to be tuned and improved once more data are available. Later it could be possible to make a test or two per week to calibrate the material flow and storage levels etc. A schematic overview of the Lubin mine with planned RFID drop positions as red circles and planned RFID identification antennas as yellow rectangles is presented in Figure 6.1. Figure 6.2 shows a breakdown of the schematic overview of the Lubin mine, in order to fit the simulation framework that has been developed for this project.

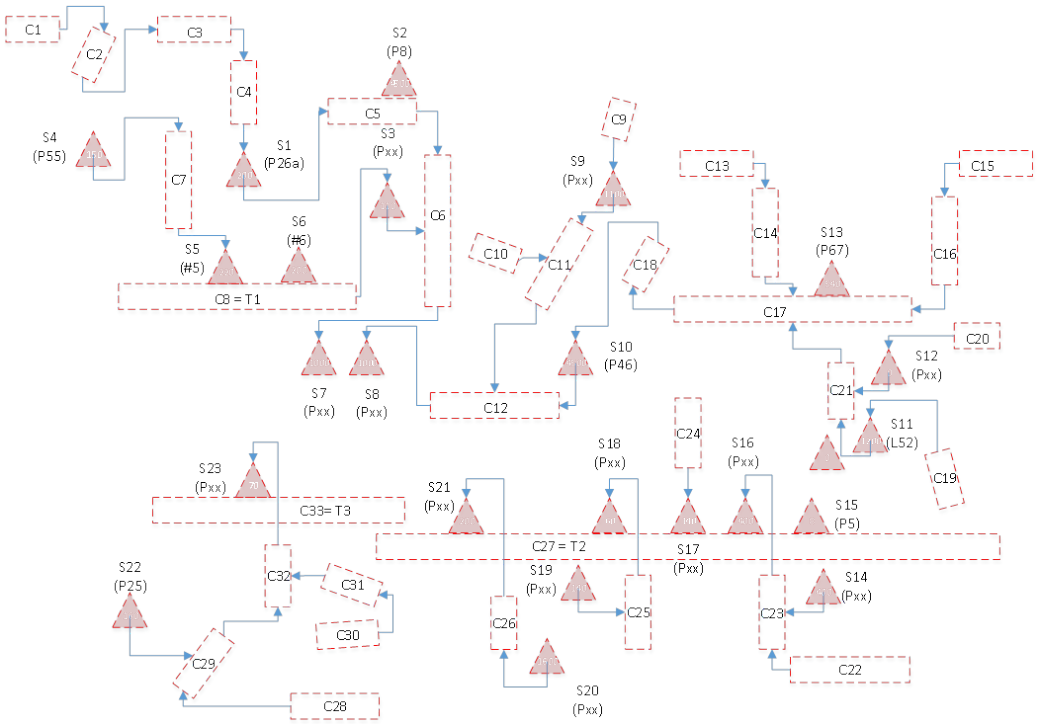

Figure 6.2: Break down of the schematic overview of the Lubin mine, in order to fit the simulation framework that is developed for this project.

6.3 Modelling and Simulation Framework

A basic framework is in place, where ore transport out of a mine can be set up and simulated. The ore transport in a mine can be constructed using a library of Matlab functions that has been developed for this purpose in this project[4]. The library contains functions for initiation, connection, simulation and plotting the mine. The simulation model is in discrete time and it is sampled using a fixed time step. The ore dynamic is discretised by volume along the transportation. Some equipment contains more than one sample. Incoming ore from different samples and from different locations are can be mixed when it enters the samples of an equipment.

6.3.1 Ore handling equipment models

Below are the models implemented in the Matlab simulation framework. These models will be improved and extended to fit the Lubin mine in greater detail once more real data becomes available.

6.3.1.1 LHD

LHDs move ore from face to other ore-handling equipment periodically. The simulation can be performed with noise added to the time period.

[4]The IPR status of this library is *confidential*.

6.3.1.2 Trucks

There are no models for truck transports in the mine implemented yet.

6.3.1.3 Grid

The grid is limiting the ore flow out. This is a simple simulation of the operation of a hydraulic hammer poking around and hammering larger rocks into smaller ones at the grid.

6.3.1.4 Conveyor belt

Ore can be fed at inputs at one or several positions along the conveyor belt. The belt is assumed to run at fixed speed. The ore leaves the belt at its end position. The conveyor belt is sampled in time domain.

6.3.1.5 Train

A train can collect ore from several places and leave it at other places. The ore is mixed on the train.

6.3.1.6 Store

The type of store that implemented so far is a FIFO store and it has a fixed number of positions, or samples. Each sample has an upper volume limit. The first sample is filled up before next sample is filled. Ore is taken from the last sample until it is empty. Ore will then be taken from previous sample.

6.3.1.7 Hoist

The hoist can be in four different states: At the bottom, on the way up, at top or on the way down. Ore can be filled into the hoist when it is at the bottom. Ore can be taken out of the hoist when it is at its top position. The hoist calculates how much space there is left for ore, in order for up-stream equipment to limit the output to the hoist. The hoist can also limit the output if downstream equipment requires so. The ore is mixed in the hoist.

6.3.1.8 Crusher

The crusher is not implemented as a specific model. A store with small maximum output rate is used to model crushers at the moment.

6.3.2 Simulation framework

An example of a simple mine with three conveyor belts and a hoist is presented in Figure 6.3. The figure is a snapshot of the simulator during simulation, with graphics update activated. Conveyor belts C1 and C2 transport the ore to conveyor belt C3. C3 transports the ore to store S1. The hoist H1 brings the ore from S1 to store S2. Triangles indicate that there is a heap of

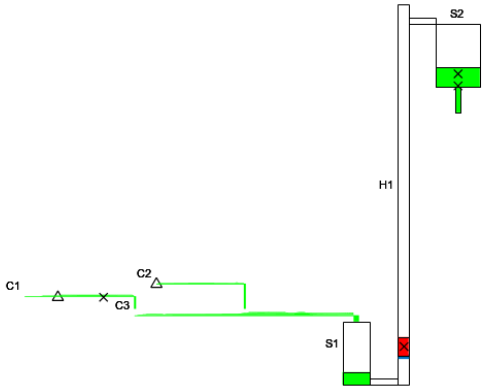

Figure 6.3: A simple mine with three conveyor belts and a hoist.

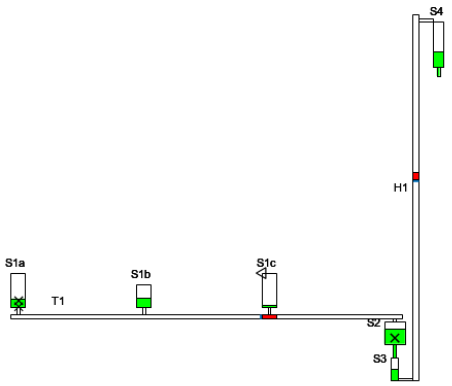

Figure 6.4: A simple mine with three conveyor belts and a hoist.

ore on the grid where the ore is dumped on the conveyor belt. “X” indicates the position of a RFID-tag. There is one RFID-tag on C1, one in the hoist, and two in S2. Green colour on the store/hoist indicates the relative ore level. Green turns to red to indicate “full”. The thickness of the green lines of the conveyor belts indicated the level of the ore at specific positions on the belts.

An example of a simple mine with a train, a crusher and a hoist is presented in Figure 6.4. The figure is a snapshot of the simulator during simulation with graphics update activated. The train T1 transports the ore from the stores S1a, S1b and S1c to the store S2. The ore that is leaving S2 is crushed before it enters store S3. The hoist H1 brings the ore from S3 to store S4. Triangles indicate that there is a LHD or truck that fill a store. “X” indicates the position of a RFID-tag. There are two RFID-tags in S1a and one in the S2. Green colour on the store/train/hoist indicates the relative ore level. Green turns to red to indicate repletion.

A schematic view of the small part of the KGHM mine is presented in Figure 6.6. This part of the mine was built up in the simulation framework.

An animated simulation of the small part of the KGHM mine is presented in Figure 6.6. The

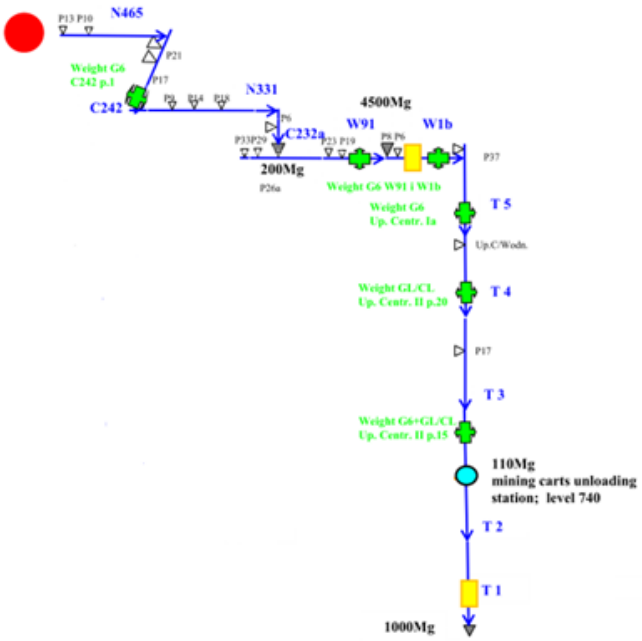

Figure 6.5: Schematic view of a small part of the KGHM mine with four conveyor belts followed by a store, and finally two more conveyor belts.

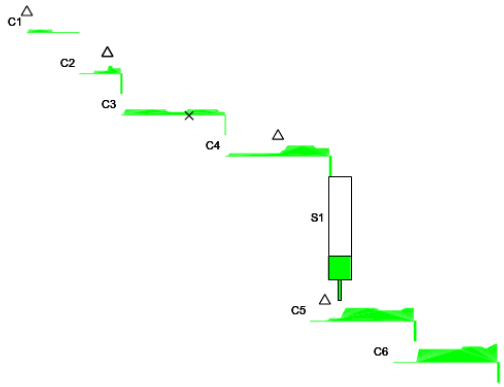

Figure 6.6: Simulation of a small part of the KGHM mine with four conveyor belts followed by a store, and finally two more conveyor belts.

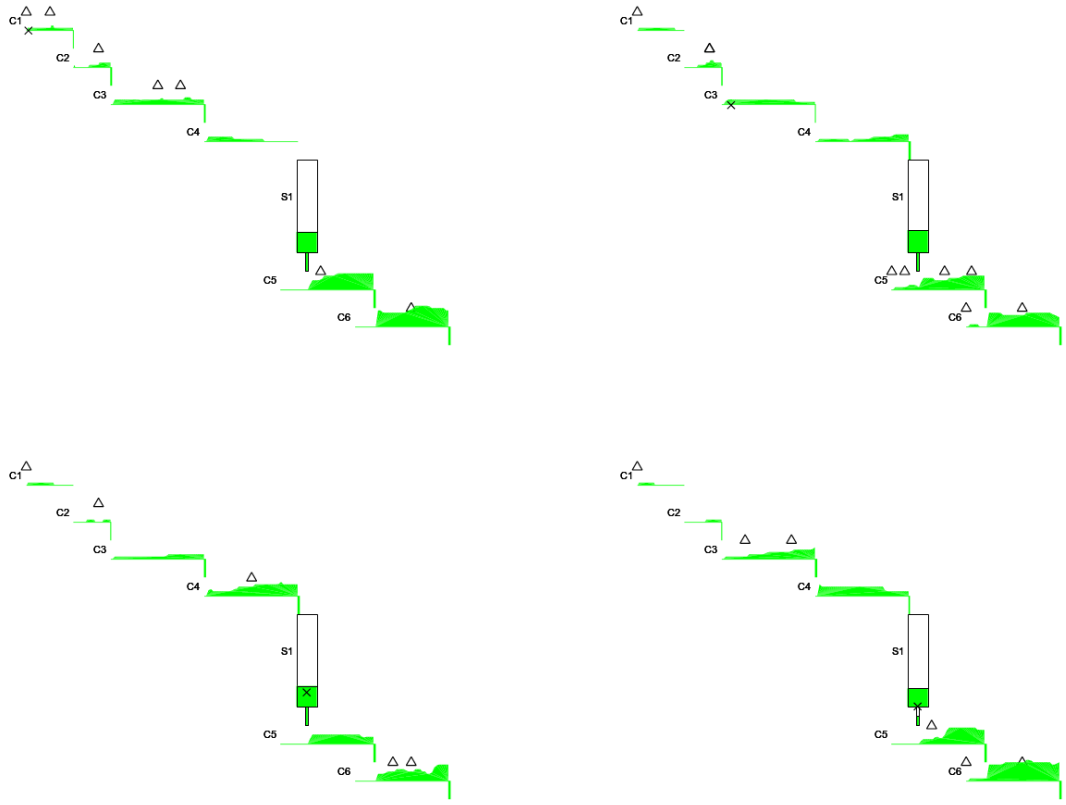

Figure 6.7: Simulation at four different time instants.

model consists of four conveyor belts followed by a store, and finally two more conveyor belts. The figure is a snapshot of the simulator during simulation, with graphics update activated. Conveyor belts C1 to C4 are operating in sequence and they transport the ore to store S1. Two more conveyor belts continues the ore transportation. Triangles indicates heaps of ore on the grids where the ore is dumped on the conveyor belts. “X” indicates the position of RFID-tags. There is one RFID-tag on C3,. Green colour on the store indicates the relative ore level. Green turns to red to indicate 100% repletion. The thickness of the green lines of the conveyor belts indicates the level of the ore at specific positions on the belts.

In Figure 6.7 shows how the ore flow and RFID position is changing over time. The uppermost sub-figure to the left is the first figure, the uppermost figure to the right is the second figure, etc.

6.3.3 Advanced storage models

Since the bunker/storage models is the most difficult and important part in the modelling of the conveyor system and for future control, a lot of effort has been put into the storage modelling. The first approach was to look at *cellular automata* (CA) and later the plan is to evaluate storage models with SPH or similar.

6.3.3.1 The cellular automata approach

The first approach for the advanced storage modelling was to use the statistical method of CA.

The cellular automata (CA) approach is a powerful modelling technique for describing and simulating complex behaviour of physical systems, without paying much respect to the physical processes actually involved (Chopard & Droz, 1998 [109]). The approach is based on a logical abstraction where each element is defined to be self-replicating according to simple and repetitive transition rules that change the elements internal state, where the rules are either deterministic or probabilistic (based on some criterion) (Martinez & Masson, 1998 [110]). Each cell within the lattice is therefore assigned a set of state variables that can be said to represent the physical state of that cell (Hallberg, 2011 [111]). A more formal definition of the CA is presented by Chopard and Droz (1998) [109], where they state the formal requirements that is needed:

1. A regular lattice of cells covering a portion of a d-dimensional space
2. A set $\Phi(r,t) = (\Phi_1(r,t), \Phi_2(r,t), \ldots, \Phi_m(r,t))$ of Boolean variables attached to each site of the lattice and giving the local state (r) of each cell at the time $t = 0, 1, 2, \ldots$
3. A rule $R = (R_1, R_2, \ldots, R_m)$ which specifies the time evolution of the states in each time step.

It is important to remember that the time and space always is discrete in a CA and that the rules for the transitions is homogenous for the entire lattice, even though some inhomogeneity can be introduced, typically because of the conditions for neighbouring cells (Chopard & Droz, 1998 [109]). The neighbouring cells, which can be defined as the cells in the spatial region surrounding the central cell that are expressed within the demarcation for the rule set of the updating cells (Chopard & Droz, 1998 [109]). The neighbouring cells can include just the nearest

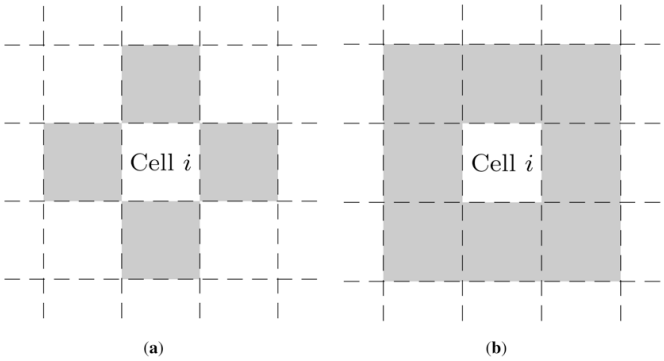

Figure 6.8: Examples of neighboring cells: a) Von Neumann neighborhood; b) Moore neighborhood. (Source: Hallberg, 2011 [111]).

cells or cells even further away (Hallberg, 2011 [111]). Different rule sets often come with different types of neighbourhoods which can be seen in Figure 6.8.

At each time step, the neighbourhood is identified so that the correct transition can be made based on the switching rules for this specific environment (Hallberg, 2011 [111]). As long as the CA approach is kept deterministic, the outcome of a certain environment will always give the same evolution, but sometimes it is useful to add a degree of randomness to the rules (Chopard & Droz, 1998 [109]). This is for example of interest when a certain situation can lead to several probable outcomes and changes for the states (Chopard & Droz, 1998 [109]) similar to the scenario-based approach discussed in Chapter 3. The probabilistic cellular automata is a very useful generalization since it allows the user to more easily model flow rates and velocities that is a common factor when dealing with physical systems, since the only parameters that govern the particle motion are the transition probabilities and the dimensions of cells [112].

CA have been used and developed since the 1940s in many different research fields and areas (Chopard & Droz, 1998 [109]). Some of the most commonly application areas are flows of a fluid, a gas or a granular medium within specific volumes (Martinez & Masson, 1998 [110]; Chopard & Droz, 1998 [109]). The computational efficiency of the method is largely because of the fact that the calculations only pay respect to the neighbouring cells and not the whole lattice and therefore only a small amount of information is taken into account (Hallberg, 2011 [111]).

According to Martinez and Masson “Despite their extreme simplicity, the degree of agreement with actual flows is very satisfactory” [110]. Osinov (1994) [112] has used the CA approach to model the deformation and flow of loose materials within solid walls in order to account for the effects of loosening (decrease in density) during the material discharge. Osinov (1994) identifies the following advantages with the CA approach [112]:

1. The algorithmic simplicity and minimal number of mathematical operations which makes it not require large computational resources
2. The possibility of application of the model to 3D flows, even though it is known that an extension like this comes with considerable complications.
3. The possibility of adequately model the flow near the free surface
4. The absence of restrictions for the value of deformations which enables us to trace the process of flow up to any stage

Savage (1993) proposed a model for the simulation of hopper flows based on a randomness

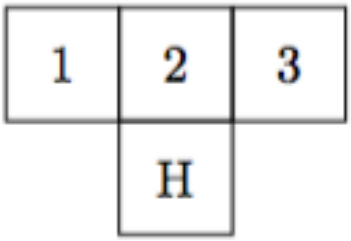

Figure 6.9: Mitigation rule 1 (Source: Kozicki and Tejchman, 2004) [114]

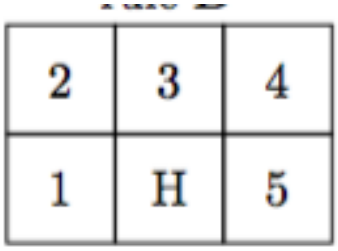

Figure 6.10: Mitigation rule 2 (Source: Kozicki and Tejchman, 2004) [114]

in the way that the void travels from the opening to the top of the hopper [113]. This is done by introducing a lattice of cells, where each non-empty cell has a probability of falling into the empty cell located in the closest neighbourhood. Savage (1993) introduced the lattice shown in Figure 6.9, where three cells are located directly over the empty cell (marked with an H) [113].

Kozicki and Tejchman (2004) used a similar type of model but developed it further, adding more cells into the lattice rules which enabled the model to be better at imitate a real-world hopper [114]. Figure 6.10 presents one of the mitigation rules that they proposed.

They also developed 3D models and hexagonal models, which has been considered but has not been evaluated.

The conditions of course comply that the following sum has to be fulfilled:

$$\sum_{j=1}^{5} p_j = 1 \tag{6.1}$$

6.3.3.2 Description of CA

The model that was developed is based on the theory of probabilistic cellular automata, where small particles in the silo are being modelled by small squares and where each cell movement is based on the neighbourhood and a probability factor. The cells in the model can be of one of the following three types:

1. The cell that is occupied with a hole
2. The cell that is occupied with a part of the wall
3. The cell that is occupied with a particle

The second state, the cells that makes up the wall, is the only fixed state that remains the same during the entire execution of the lattice cells. The other ones will follow the probabilistic transition rules. In reality the cells that is occupied with a particle would be divided into several

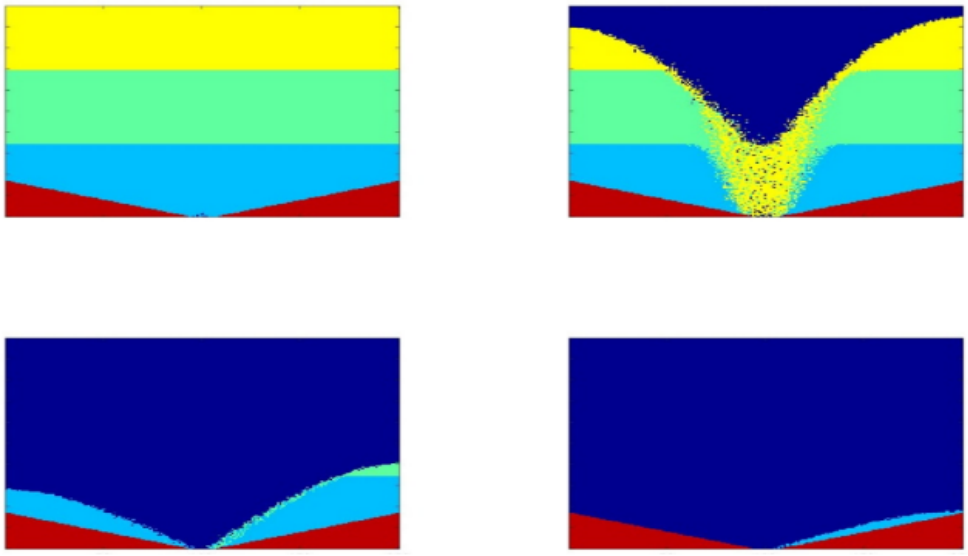

Figure 6.11: Silo simulation (at different iterations) with asymmetrical calculations from the left to the right.

subgroups due to the difference in size, weight and density of the material.

6.3.4 Asymmetrical behaviour inherited from the theory

In the Matlab code, the calculations are carried out from the bottom and up since the “holes” are created at each iteration where the opening is modelled to be. At each row the calculations are performed from one side of the silo to the other, and a movement is taking place if a hole is present surrounded by particles.

In Figure 6.11, the calculations are performed from the left to the right, which gives an interesting asymmetrical behaviour where the particles in the left side of the silo have a higher tendency of moving out towards the opening.

Due to this behaviour, a random factor was introduced before the iteration starts that randomises the calculation order in each row and therefore evens out the symmetry which can be seen in Figure 6.12. It is also interesting to notice that the calculations for the new model is much slower (which can be seen in the extended amount of iterations that it takes for the silo to be emptied).

6.3.4.1 The probability choice

The only factors of importance that can be varied in the model is: the size of the silo (the amount of particles), the inclination of the silo and the movement probability of the particles where the latter is the only one that have a significant impact on the behaviour of the material. Since the behaviour is symmetrical, the two parallel particles (2–4 and 1–5 in mitigation rule 2) will have the same probability of filling up the hole. Therefore, it is only 3 parameters of interest for the entire silo behaviour, but the results are still very interesting. Even though the model is heavily based on a random function, it still exhibits behaviours of stagnation at different parts of the silo which corresponds very well with the theories according to different flow zones and stagnant

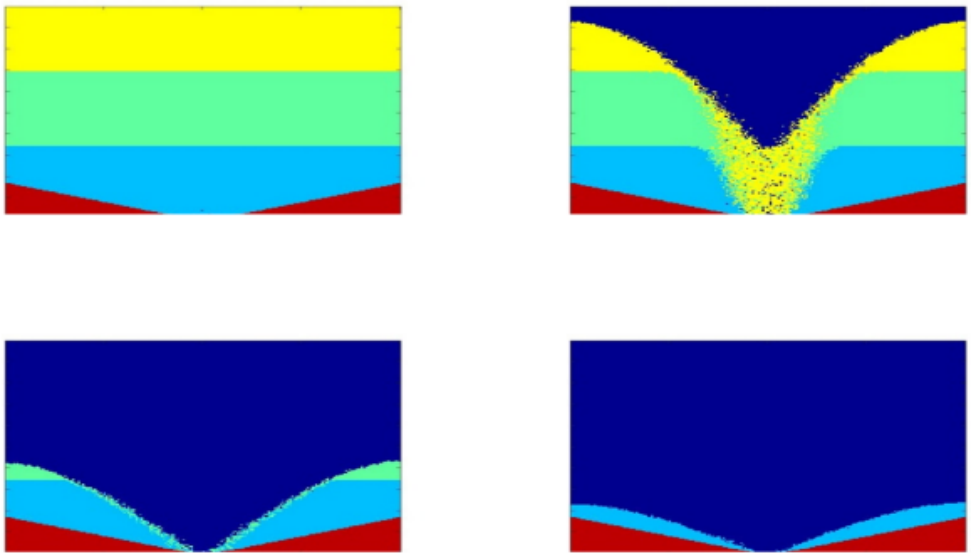

Figure 6.12: Silo simulation (at different iterations) with a randomised calculation order in each row. Top left: 1 iteration; Top right: 4000 iterations; Bottom left: 12000 iterations; Bottom right: 18000 iterations.

zones. The most active flow zones in the silo simulation almost lets the particles through in a FIFO-manner, the less active zones will reduce the particle speed and sometimes even submit it to total stagnation.

The first visualisation, shown in Figure 6.13, shows the behaviour when the centre probability, p_2, is set to zero which almost can be likened to the mass flow situation which was presented in the theory section.

The second visualisation, shown in Figure 6.14, shows the behaviour when the middle probability, p_2, is set to 0.6 which is similar to the funnel flow behaviour of the silo.

The disallowed case where the sum of p_j is greater than 1, creates some interesting behaviour, where the outflow movement is curved upwards which can be seen in Figure 6.15.

6.3.4.2 Inflow simulation

When ore are being added to the silo during discharge, the random function is still applicable, which makes the particles shatter in an unnatural way, which can be seen in Figure 6.16 at iteration 2100. In the case where a particle is not surrounded by neighbouring particles it is obvious that gravity would have the biggest impact on the particle movement (since there is no wind in the silo) and therefore the probability that the particle falls straight down would be closer to zero.

If the position of the inflow is changed so that the material is added at another part of the silo, it is obvious that Schwedes flow areas in the theory is a part of this simulation, which can be seen in Figure 6.17.

6.3.4.3 The efficiency of the model and how to speed it up

A reference test on how to improve the execution time has been performed, in order to compare different solutions.

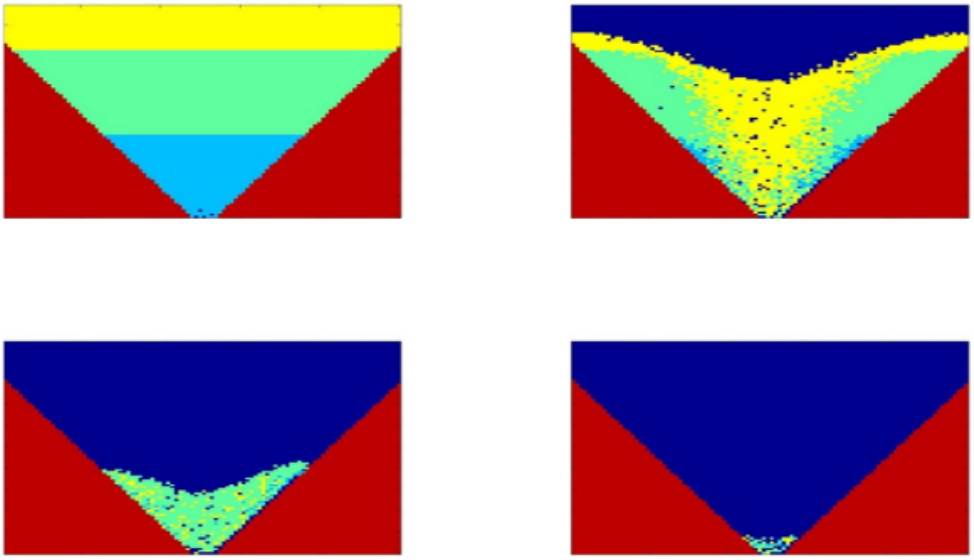

Figure 6.13: Silo simulation (at different iterations) with angle of 60 degrees and the probabilities $p_2 = 0$, $p_1 = p_3 = 0.25$ and $p4 = p5 = 0.25$. Top left: 1 iteration; Top right: 1800 iterations; Bottom left: 3900 iterations; Bottom right: 4500 iterations.

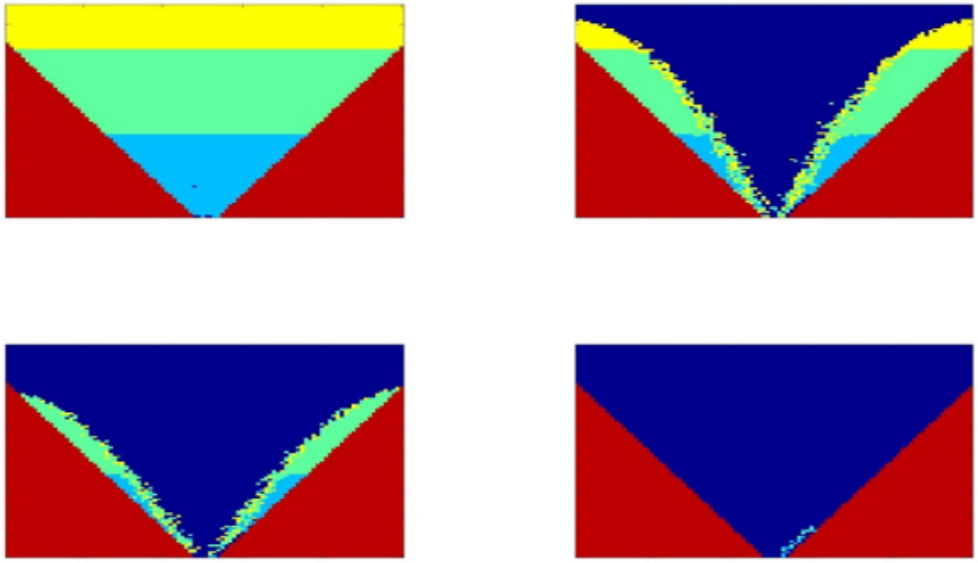

Figure 6.14: Silo simulation (at different iterations) with angle of 60 degrees and the probabilities: $p_2 = 0.6$, $p_1 = p_3 = 0.1$ and $p_4 = p_5 = 0.1$. Top left: 1 iteration; Top right: 1800 iterations; Bottom left: 2700 iterations; Bottom right: 3900 iterations.

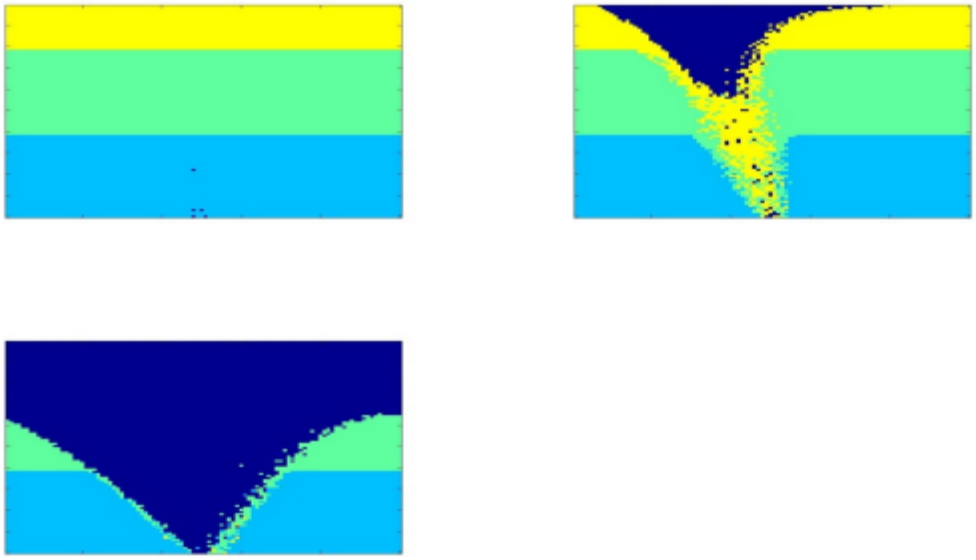

Figure 6.15: Silo simulation (at different iterations) with angle of 0 degrees and the probabilities (with a total sum over 1): $p_2 = 0.6$, $p_1 = p_3 = 0.3$ and $p_4 = p_5 = 0.3$. Top left: 1 iteration; Top right: 400 iterations; Bottom left: 2800 iterations.

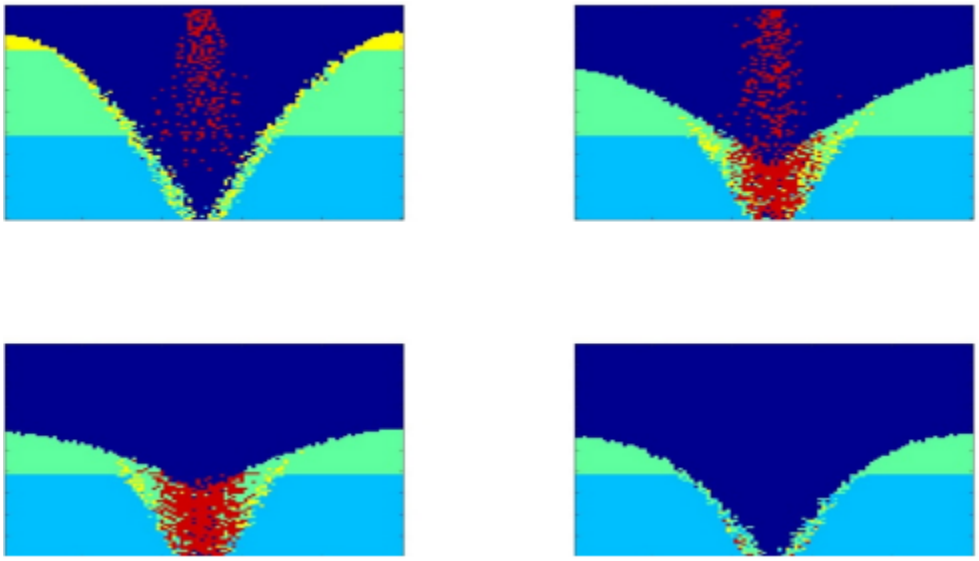

Figure 6.16: Silo simulation (at different iterations) with angle of 0 degrees and an inflow from the middle (in red color). Top left: 2100 iteration; Top right: 3300 iterations; Bottom left: 5100 iterations; Bottom right: 5700 iterations.

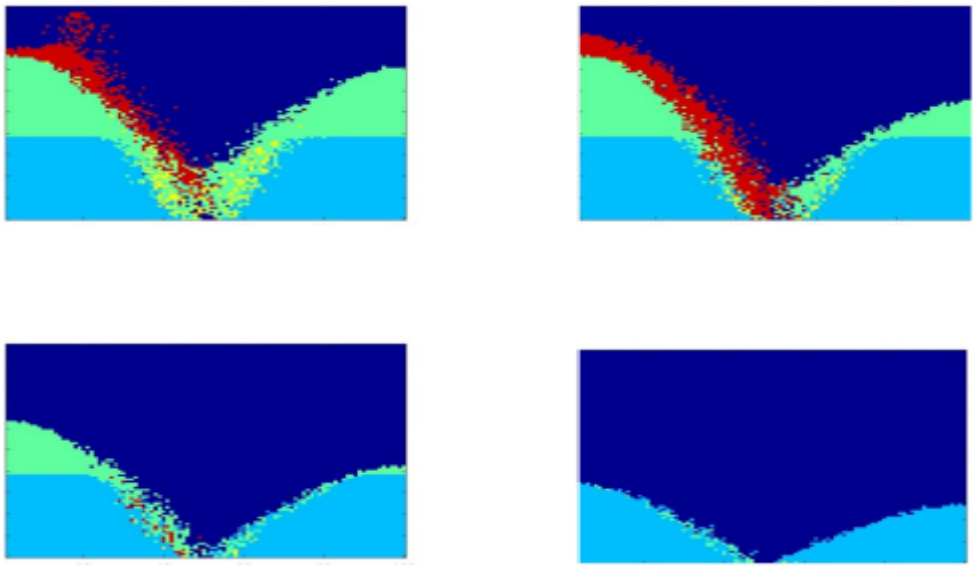

Figure 6.17: Silo simulation (at different iterations) with angle of 0 degrees and an inflow from the left (in red color). Top left: 2400 iteration; Top right: 3200 iterations; Bottom left: 4000 iterations; Bottom right: 5200 iterations.

As a first step, the Profiler built-in function in Matlab was used in order to reduce the execution time of the code as much as possible. As a second step, MEX-files of the while-loop functions was made, using the Xcode Compiler and the built-in Matlab function Matlab Coder.

From the beginning, the execution time were almost 51 seconds and after using the profiler analysis, the execution time went down to 36 seconds. The time for the calculations (without the plotting) were 24.78 seconds and this was reduced to 7.34 seconds after the conversion to C-code of some of the main functions (through Matlab Coder). Furthermore, the C code can now be used without the need to install Matlab.

Parallelisation of the code has been considered, but not tested.

6.3.4.4 Smooth particle hydrodynamics

The statistical model (CA) developed so far have shown good results, but it has some limitations when it comes to 3D modelling and does not include particles of different size. To be able to include this in the model the plan is to work evaluate the storage model with smooth particle hydrodynamics (SPH) later, so far only some minor investigation in the area have been performed although it seems promising. A lot of work have been made in the area of SPH in related areas so hopefully this results can be used as a base for further development for the material flow. One of the challenges and potential with this approach is the possibility to work with different particle sizes to really be able to understand the material flow in the silo. This can later be used for advanced control of the material flow.

6.3.5 Ore mass flow models

The ore transportation model given in Section 6.3.2 requires, inter alia, the input data describing the ore mass discharged on each grid. Then the general transportation model provides the ore

mass flow on each conveyor. In particular, the flow on conveyors charged by bunkers will be provided. In order to enable transportation simulation, a model of ore flow on the conveyors supplied only by grids has to be provided. In the following paragraph a compound Poisson process is investigated. The model parameters are fitted using real data provided by the Lubin mine, thus such models could be used as real-world scenarios. In order to validate the ore mass flow given by the general transportation model on conveyors supplied by bunkers, we also provide stochastic models of the flow on such conveyors (the second paragraph of this section). Parameters of these models are also estimated using the real data provided by the Lubin mine. Both kinds of the ore flow models are validated using statistical tests. Thus, these models could serve as reliable input data for the general transportation model and a benchmark of the storage models.

6.3.5.1 Modeling of ore flow in belt conveyor system with cyclic charging

Introduction

In analyzed data the ore is supplied to the grid cyclically. It is related to the time which the loader spent on an ore transportation from the mining face to the dumping point. One conveyor belt can be supported by several loaders, which transport the ore from different mining faces. In this paragraph we fit a stochastic model to the real data acquired from conveyor scale located on a conveyor supplied by the grid only. We model arrival times of the loaders and ore mass charged by them. Such model could be used then as an artificial input data to the general ore transportation model. We fit the models, estimate their parameters and validate the results using statistical tests. This ensures reliability of the conclusions that could be drawn from the simulation study.

The application of compound Poisson process to the ore flow on the grid and the conveyor belt seems to be relevant in this problem. The compound Poisson process is defined as:

$$X_t = \sum_{i=1}^{N_t} Y_i, \tag{6.2}$$

where $\{N_t\}_{t\geq 0}$ is homogeneous Poisson process with the intensity parameter λ and Y_i are independent and identically distributed (i.i.d.) the random variables. The process N_t can be considered as the moments of loaders' arrivals to the dumping point. To estimate parameters of the N_t process we use times between consecutive discharges, which will be called *waiting times*. The amount of ore discharged by the loader will be treated as the random variable Y described by a distribution function and called *increments*. In that approach two tasks are essential:

- estimation of intensity λ of the homogeneous Poisson process $\{N_t\}_{t\geq 0}$ based on real data,
- fitting the distribution of compound Poisson process increments Y_i and estimation of its parameters.

Generally, the continuous-time stochastic process $\{N_t\}_{t\geq 0}$ is homogeneous Poisson process (HPP) with the intensity parameter $\lambda > 0$ if [115]:

- $\{N_t\}$ is the counting process,
- the waiting times are independent and identically exponentially distributed.

To solve above problems it is needed to analyze the real data of the weight of ore supplied on the conveyor belt. The data acquired from the weight measurement system installed on the conveyor belt of ID $L413$ will be taken to show the results of the proposed methodology. This conveyor is not supplied by any other conveyor or any bunker, moreover it is supplied by only one grid. Its length is equal to 830 meters and the nominal speed is 2 m/s. The available data include the measurements of weight from November 2015 to March 2016. To use collected data it is necessary to clean acquired signals from outliers observations and measurement errors. In this report we present general results based on the pre-processed data. The entire pre-processing scheme as well as the entire modeling results are provided in Deliverable D5.1.

In Fig. 6.18 we present the investigated data of the ore mass transported through the conveyor scale. It is a binary signal. Each value of 1 might be interpreted as one metric tonne of ore transported through the conveyor scale. Zeros does not stand for idle state (no ore on the conveyor belt), since it usually takes a few seconds for a one tonne to be transported through the scale. In Fig. 6.19 one can see zoom of Fig. 6.18 with discharging moments marked. The time elapsed between each discharging is called waiting time (see Fig. 6.20a). The weight related to each discharging is presented in Fig. 6.20b. These data constitute a base for modeling of the ore mass flow in the case of cyclic transportation. Long waiting times are related to breaks in transportations, thus these longer than 1500 seconds are not taken into consideration in the analysis. Moreover, weights lower than 1 tonne and higher than 40 tonnes are not considered due to technical reasons (a single LHD or a truck cannot transport more than 40 tonnes here).

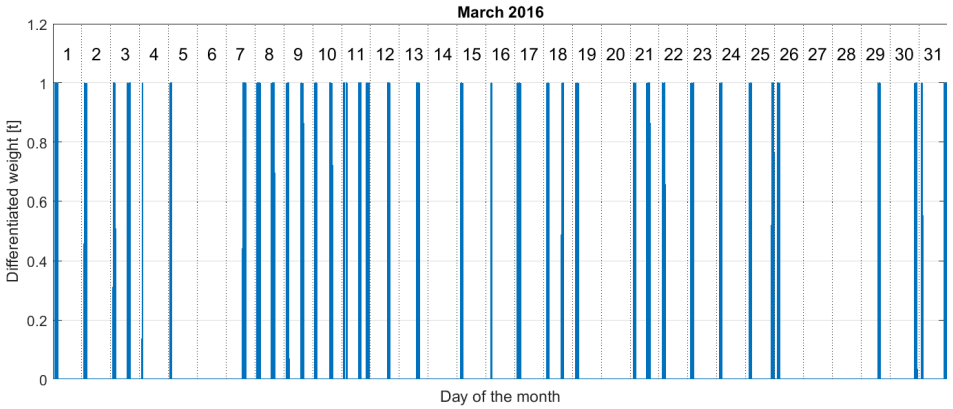

Figure 6.18: Cleaned differentiated weight data from $L413$ conveyor belt (March 2016).

Estimation of intensity of the Poisson process $\{N_t\}_{t\geq 0}$

The estimation of the intensity λ of the Poisson process $\{N_t\}_{t\geq 0}$ starts with by fitting a kernel density estimator to waiting times data from each month from the period between November 2015 and March 2016. As one can see in Fig. 6.21a the shape of the kernel density estimator is similar for different months. It may suggest that variables describing behaviour of waiting times have the same distribution. The differences between particular months are reflected only in the parameters of this distribution, thus in further analysis we investigate a single month, i.e. November 2015. The entire analysis are provided in deliverable D5.1.

The variable describing distances between two consecutive points of a homogeneous Poisson process are exponentially distributed with the intensity parameter $\lambda > 0$. The cumulative distribution function in that case is defined as [115]:

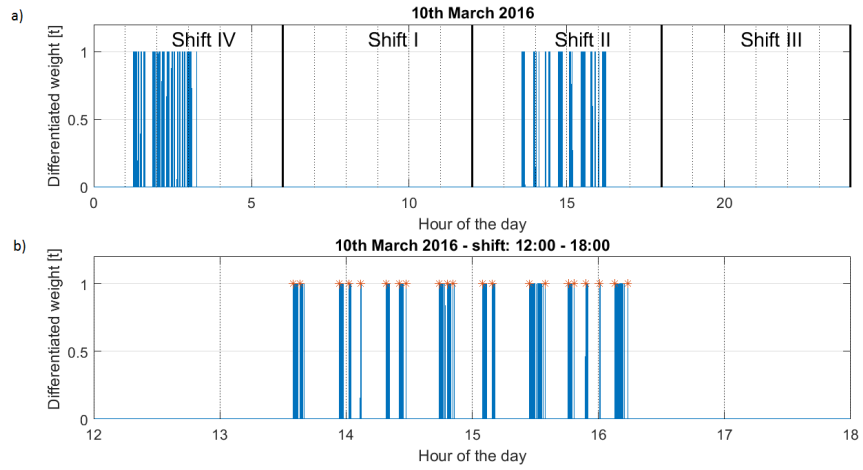

Figure 6.19: Differentiated weight data from $L413$ conveyor belt: a) data acquired from $10th$ March 2016, b) data acquired from II shift at $10th$ of March 2016. The moments of beginnings of discharging are marked by red stars.

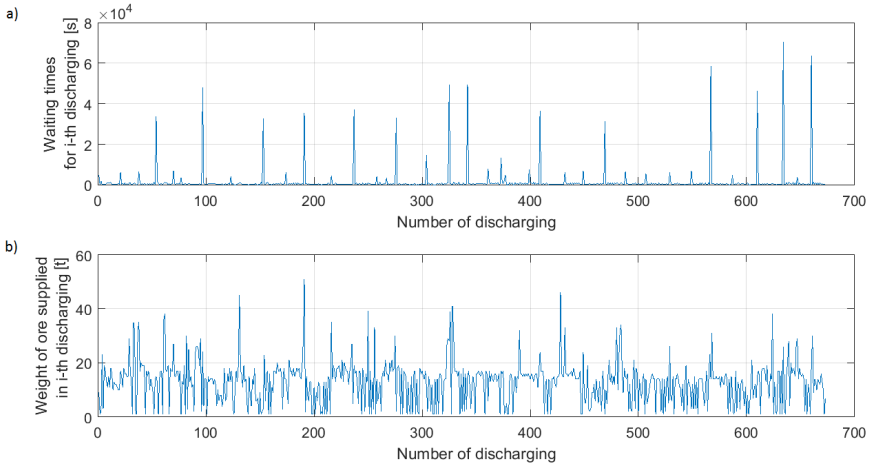

Figure 6.20: a) Waiting times for *i*-th discharges, b) weight of ore mass supplied on the conveyor in *i*-th discharge.

$$F(t) = 1 - e^{-\lambda t}, \quad t > 0. \tag{6.3}$$

Remind that in our case the described variable is related to the waiting times. One of the solutions for estimation the intensity parameter λ is fitting the exponential function $f(x) = ae^{-bx}$ to the tail of the empirical distribution function, which is defined as follows [115]:

$$\hat{F}_n(t) = \frac{1}{n} \sum_{i=1}^{n} 1_{\{x_i \leq t\}}, \tag{6.4}$$

where x_i is the i-th observation. Now it is easy to define the tail of the empirical distribution function as $1 - \hat{F}_n(t)$.

The tails of the empirical distribution function from real data are showed in Fig. 6.21b. The non-linear least squares method [116] was used to fit exponential function to the tail of empirical distribution function. The results of the analysis are presented in Fig. 6.22. The estimated b

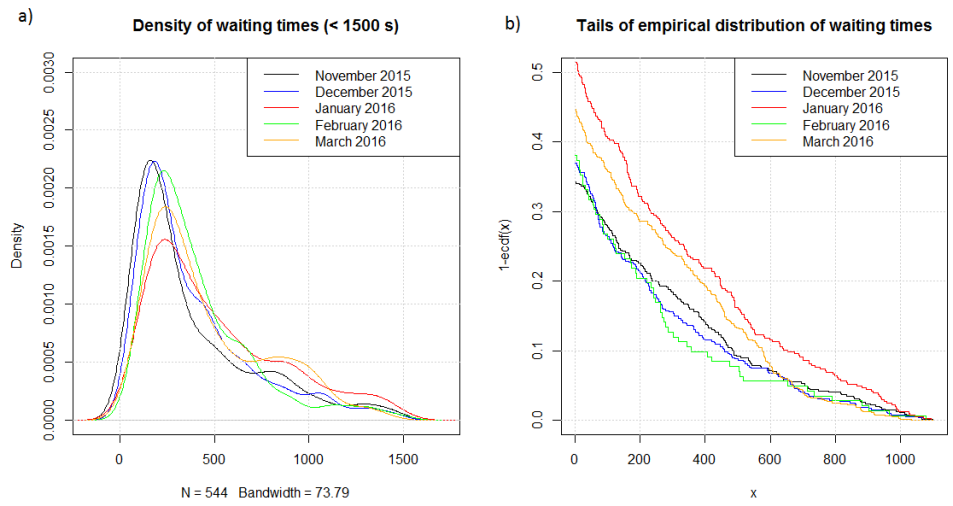

Figure 6.21: For waiting times of each month between November 2015 and March 2016 there are presented: a) fitted kernel density estimator and b) tails of empirical distribution function

parameters of exponential function for November 2015 is equal 0.003. The values of a parameter is close to 1, so the assumption about exponential distribution of tails is fulfilled. Therefore the intensity estimator $\hat{\lambda}$ of the homogeneous Poisson process is equal to the estimated parameter b equals 0.003.

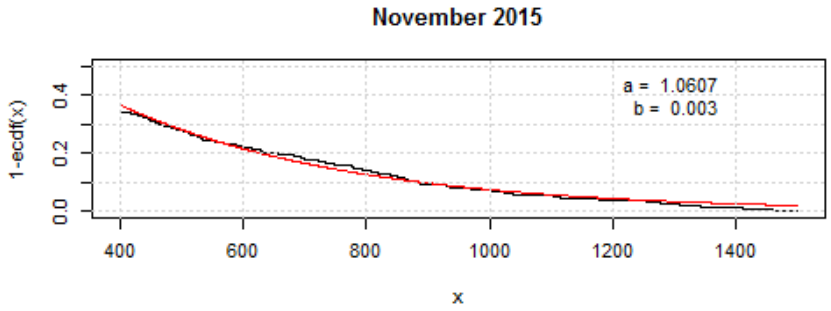

Figure 6.22: Fitted exponential function $f(x) = ae^{-bx}$ (red line) to the tails of empirical distribution function (black line) related to data from November 2015. Estimators of a and b parameters are showed on the plots.

Estimation of the compound Poisson process increments

Assuming that analysed signal follows compound Poisson process the increments Y_i are i.i.d. random variables. Clearly, the proper distribution has to be fitted to the data. Analysed data sample consists of ore mass, which flow through the grid. Obviously, the mass cannot be smaller than zero, hence only non-negative distribution can be fitted. In modelling process we tried to fit some positive distributions. Let us firstly recall some of the well-known distributions that can be useful in this case.

Let Z be a Gaussian random variable with mean equals to μ and variance σ^2. Then the random variable $X = \exp(Z)$ is log-normal. Its probability density function (pdf) $f_X(x)$ and

cumulative distribution function (cdf) $F_X(x)$ are given by following equations:

$$f_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right),$$

$$F_X(x) = \frac{1}{2} erfc\left(-\frac{\ln(x)-\mu}{\sigma\sqrt{2}}\right),$$

where $x > 0$, $\mu \in \mathbb{R}$, $\sigma > 0$ and $erfc(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt$ is the complementary error function. When the parameter σ is small the log-normal distribution can be similar to normal one. On the other hand, for large σ the tails are semi-heavy [115].

The second analysed distribution was the Gamma one. It is described by two parameters shape $k > 0$ and scale $\theta > 0$, then the probability density function and cumulative distribution function are following:

$$f_X(x) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} \exp\left(-\frac{x}{\theta}\right),$$

$$F_X(x) = \frac{1}{\Gamma(k)} \gamma\left(k, \frac{x}{\theta}\right),$$

where $x > 0$, $\Gamma(\cdot)$ is a gamma function and $\gamma\left(k, \frac{x}{\theta}\right) = \int_0^{\frac{x}{\theta}} t^{k-1} e^{-t} dt$ is a lower incomplete gamma function. It is worth mentioning that the sum of k i.i.d. exponentially distributed random variables $\sum_{i=1}^k X_i$ follows Gamma distribution with shape parameter k and scale λ^{-1}, where λ is an intensity of the X_i.

Another non-negative distribution, which we tried to fit, is the Weibull distribution. It also depends on two positive parameters: λ - scale and k - shape. For $x \geq 0$ the probability density function and the cumulative distribution function are given by the formulas:

$$f_X(x) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \exp\left(-\left(\frac{x}{\lambda}\right)^k\right),$$

$$F_X(x) = 1 - \exp\left(-\left(\frac{x}{\lambda}\right)^k\right).$$

In case of $k = 0$ the Weibull distribution reduces to the exponential one.

Finally, let us present the three parameters family of positive Burr distribution. Its probability density function and cumulative distribution function are:

$$f_X(x) = \frac{\frac{kc}{\alpha}\left(\frac{x}{\alpha}\right)^{c-1}}{\left(1+\left(\frac{x}{\alpha}\right)^c\right)^{k+1}},$$

$$F_X(x) = 1 - \frac{1}{\left(1+\left(\frac{x}{\alpha}\right)^c\right)^k},$$

where $x > 0$, $\alpha > 0$ is the scale parameter, $c > 0$ and $k > 0$ are the shape parameters. This distribution is very flexible and can express many types of real data. The limit distribution when $k \to \infty$ is the Weibull one [117]. In Fig. 6.23 there are presented probability density function estimators for the weights distribution. One can observe that the distributions are bimodal, hence none of the previously introduced distribution can be fitted directly. In order to overcome this problem we would like to analyze the mixture of the distributions. Namely, let us assume

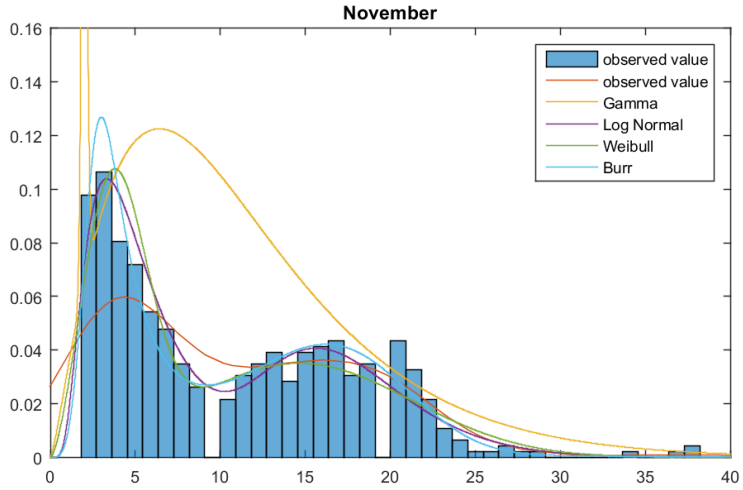

Figure 6.23: Comparison of histogram for real data, kernel density estimator and several mixtures of fitted distributions.

that X is a random variable which probability distribution function $f_X(x)$ is following:

$$f_X(x) = pg_1(x) + (1-p)g_2(x), \tag{6.5}$$

where $p \in [0,1]$ is a mixing parameter, $g_1(x)$ and $g_2(x)$ are the probability distribution functions. One can observe that it is a linear combination of two other probability distribution functions. Clearly the similar relation can be observed in the cdf:

$$F_X(x) = pF_1(x) + (1-p)F_2(x), \tag{6.6}$$

where $F_1(x)$ and $F_2(x)$ are the cumulative distribution functions related to g_1 and g_2 respectively. Using such mixing techniques we are able to create a wide class of the distributions. Each distribution with the explicit form of pdf can be applied. Moreover, as one can expect it is possible to create bimodal density functions.

The parameter estimation for such mixture of distributions can be performed with Maximum Likelihood Estimation (MLE) method. For the vector of independent observation $x = (x_1, \ldots, x_n)$ and the set of the distribution parameters θ the estimation is done by maximizing the log-likelihood function:

$$\max_{\theta} \{L(\theta)\} = \max_{\theta} \left\{ \sum_{i=1}^{n} \tilde{f}(x_i; \theta) \right\}.$$

The method can be easily applied for the distribution with known pdf. In Fig. 6.24 there is presented a flowchart of the estimation method. One can observe that MLE method is used to estimate the set of parameters for densities $g_1(x)$ and $g_2(x)$ with a priori given value of the mixing parameter p. In order to find the best p the Kolmogorov-Smirnov test is performed. The test statistics is defined as:

$$K(\theta, p) = \max_{x} \{|F_E(x) - F_T(x)|\},$$

where $F_E(x)$ is the empirical cumulative distribution function and $F_T(x)$ is the fitted theoretical

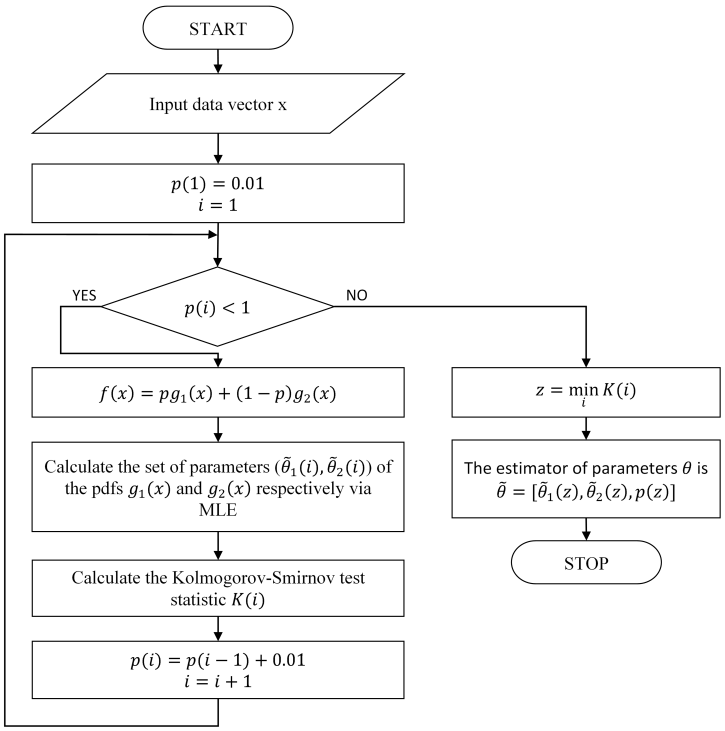

Figure 6.24: Flowchart of the modeling scheme in case of ore supplied from the bunker. Preprocessing step is included.

cdf. The statistic measures the maximum distance between the theoretical and empirical cdf. Hence, minimizing the value of the $K(\theta, p)$ we are able to chose the best value for the mixing parameter. This procedure allows us to fit the bimodal distribution to the data.

In Fig. 6.23 the fitted densities are presented and compared with the normalized histogram and the kernel density estimator. Some of the distributions explain the data really good. Indeed, especially mixture of log-normal and Weibull distributions fit to the observation. They are able to detect both modes in the data. One can observe that only mixture of Gamma distributions significantly differs from the real data.

In order to ensure the correctness of the fitting, the statistical tests can be performed. Obviously, the Kolmogorov-Smirnov test can be applied. Let us also recall three other tests based on the empirical distribution function: Kuiper test, Cramer-von Mises test and Anderson-Darling test. Let us assume that $F(x)$ is a true cdf of the random variable X and $z_i = F(x_i)$, where Z is uniformly distributed on $(0, 1)$ and z_i is a realization of Z. Then the distance between empirical cdf of z_i and uniform distribution are the same as for empirical cdf of x_i and its true cdf [115] for any $i = 1, \ldots, n$. This fact can be used in computation of the test statistics. Let us assume that $z_{(1)} < \cdots < z_{(n)}$ are the order statistics of z then the test statistics (V- Kuiper, W^2-Cramer-von Mises, A^2-Anderson-Darling) are following:

$$V = \max_{1 \le i \le n} \left\{ \frac{i}{n} - z_{(i)} \right\} + \max_{1 \le i \le n} \left\{ z_{(i)} - \frac{i-1}{n} \right\},$$

$$W^2 = \sum_{i=1}^{n} \left(z_{(i)} - \frac{2i-1}{2n} \right)^2 + \frac{1}{12n},$$

$$A^2 = -n - \frac{1}{n}\sum_{i=1}^{n}\left[(2i-1)\log z_{(i)} + (2n+1-2i)\log(1-z_{(i)})\right].$$

Having four statistical tests we are able to compare the results for different distributions. In order to calculate test p-value, the Monte Carlo method can be applied. Namely, the n-long random vector from assumed distribution with estimated values of parameters is generated 10000 times. Then for each random vector the test statistics are computed and compared to the test statistics for real data. The p-value is given by equation:

$$p - value = \frac{\#(TS_i > TR)}{10000},$$

where TS_i is a test statistics for $i-$th random vector and TR is a test statistics for real data. The output of the tests is presented in Tab.6.2. For November the best results are obtained for mixture of Burr distributions. The test results ensure us that the fit of mixture of two positive distributions can by beneficial in case of modelling the ore weight.

Table 6.2: Test statistics and the p-value for different mixing distributions and estimation performed by algorithm presented in Fig.6.24.

Mix distribution	K	p-val	V	p-val	W^2	p-val	A^2	p-val
November 2015								
Gamma	2.13	0.00	4.05	0.00	1.09	0.00	7.19	0.00
log-normal	1.48	0.01	2.46	0.00	0.35	0.13	2.28	0.03
Weibull	1.42	0.03	2.82	0.00	0.38	0.06	2.99	0.02
Burr	1.30	0.04	2.58	0.00	0.20	0.21	2.05	0.05

Conclusions

In this report there was considered the problem of modeling the process of loaders' arrivals to the dumping point and the mass of the ore supplied on the conveyor belt. The compound Poisson process was considered as a mathematical tool describing that stochastic process. Firstly, we pre-processed weight data from the conveyor belt to obtain information about waiting times between two consecutive loaders' arrivals and the discharged ore mass. Then the homogeneous Poisson process was identified as a process model of the waiting times. Based on the fitted exponential function to the tail of the empirical distribution function, the intensity parameter was estimated. The next step was to identify the distribution of the variable describing supplied ore mass. Since the kernel density estimator fitted to the real weight data revealed the bimodal probability distribution function, we considered the mixture of the different distributions with the mixing parameter p. The best mixture of distributions were selected by statistical test and their parameters were estimated using Maximum Likelihood Method (MLE). The analysis were performed for the data from November 2015 and conveyor $L413$ in order to present the methodology of modeling such kind of process. The modeling results are validated using statistical tests, thus simulations based on the model can be considered as close-to-real data.

6.3.5.2 Modelling of ore flow in Belt Conveyor System with continuous charging from retention bunker

Introduction

The ore mass on the belt conveyor can be contained in separable heaps or in relatively constant ore mass flow. The first case primarily characterise the ore supplied from the grid, and the second one characterise the case when the ore comes from the retention bunker. The retention bunker task is to fulfil one of following purposes:

- equalise the level of ore mass flow,
- provide the constant character of ore mass flow,
- store the mined ore in place where the haulage switches from district to main.

In this report the problem of modelling ore mass flow from retention bunker is undertaken. It will be shown that in general, the process describing how the flow of ore mass falls from retention bunker can be modelled with relatively low order ARMA process .

The base for this report analysis are recordings from belt conveyor $L212$ from Polkowice-Sieroszowice copper ore mine which transports ore mass from retention bunker. The belt conveyor is equipped with a conveyor scale which collects the data describing temporary mass. The data consists of periods that have stationary or changing character and in this report only the stationary ones will be considered (as they describe the ore flow from retention bunker). To validate the results an exemplary segment with relatively constant temporary mass will be analysed. The entire analysis is performed in D5.1.

Methodology

In this section all essential definitions and algorithms which were the part of the analysis are included.

Definition 1. *[118]*
*Time series $\{X_t\}$ is **weakly stationary** if*

1. *$\mu_X(t)$ is independent of t,*
2. *$\forall h$, $\gamma_X(t+h,t)$ is independent of t,*

where $\mu_X(\cdot)$ and $\gamma_X(\cdot,\cdot)$ are mean and covariance function respectively.

In the following part of this report, whenever the term 'stationary' is used in time series context it means that the time series is weakly stationary.

Definition 2. *[118]*
*The time series $\{X_t\}$ is an **ARMA(p,q) process** if it is stationary and $\forall t$ it satisfies*

$$X_t - \phi X_{t-1} - ... - \phi_p X_p = Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q}, \tag{6.7}$$

where $\{Z_t\} \sim WN(0,\sigma^2)$ and polynomials $1-\sum_{i=1}^{p}\phi_i z^i$ and $1+\sum_{i=1}^{q}\theta_i z^i$ do not posses common factors.
Classical ARMA version assumes that $\{Z_t\} \sim \mathcal{N}(0,\sigma^2)$

Theorem 1. *[118]*
Stationary solution of $X_t - \phi X_{t-1} - ... - \phi_p X_p = Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q}$ for $\{X_t\}$ exists (and

is unique) if and only if

$$\phi(z) = 1 - \phi_1 z - ... - \phi_p z^p \neq 0, \forall |z| = 1. \tag{6.8}$$

Modelling time series $\{X_t\}$ with ARMA requires $\{X_t\}$ observations being made at fixed constant time intervals, namely $t \in t_0 + k\Delta t$, $k \in \mathbb{N}$, where Δt- time interval. When dealing with time series whose time intervals are varied one can interpolate the values in new time points (which are distributed equally) with e.g. linear interpolation.

Definition 3. *[119] A random variable X with distribution f has **characteristic function** defined as*

$$\phi_X(t) = \int_{\mathbb{R}} e^{its} f(s) ds = \mathbb{E} e^{itX} \tag{6.9}$$

Definition 4. *[115, 120] A random variable X is from α-stable distribution with parameters:*

- *α- index of stability ($\alpha \in (0, 2]$)*
- *β- skewness parameter ($\beta \in [-1, 1]$)*
- *σ- scale parameter ($\sigma > 0$)*
- *μ- location parameter ($\mu \in \mathbb{R}$)*

if and only if it's characteristic function $\phi_X(t)$ appears as

$$\log \phi(t) = \begin{cases} -\sigma^\alpha |t|^\alpha \{1 - i\beta \textit{sign}(t) \tan \frac{\pi\alpha}{2}\} + i\mu t, & \alpha \neq 1 \\ -\sigma |t| \{1 + i\beta \textit{sign}(t) \frac{2}{\pi} \log |t|\} + i\mu t & \alpha = 1 \end{cases} \tag{6.10}$$

and it is denoted as $X \sim S_{\alpha,\sigma,\beta,\mu}$

In most cases the form of α-stable random variable's probability density function is unknown in terms of elementary functions (with exceptions: $\alpha = 1$- Cauchy distribution, $\alpha = 2, \beta = 0$- normal distribution, $\alpha = 0.5, \beta = 1$ Levy distribution, $\alpha = 0.5, \beta = -1$ Levy distribution reflection).

Definition 5. *The time series $\{X_t\}$ is an **ARMA(p, q) process with α-stable residuals** if it is stationary and $\forall t$ it satisfies*

$$X_t - \phi X_{t-1} - ... - \phi_p X_p = Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q}, \tag{6.11}$$

where polynomials $1 - \sum_{i=1}^{p} \phi_i z^i$ and $1 + \sum_{i=1}^{q} \theta_i z^i$ do not posses common factors, and $\{Z_t\}$ is a time series of independent identically distributed (i.i.d.) random variables from α-stable distribution.

The problem of evaluating optimal model order is often solved with use of Akaike information criterion (AIC) [121]. However when dealing with models that has residuals from α-stable distribution one has to use e.g. modified AIC [122] as the classical one requires to know the form of distribution function. The modified AIC uses approximation of probability distribution function. The value of AIC is evaluated as follows:

$$AIC = -2 \sum_{i=1}^{N} \log p_{S_{\alpha,\sigma,\beta,\mu}}(Z_i) + 2p, \tag{6.12}$$

where $p_{S_{\alpha,\sigma,\beta,\mu}}$ is approximated density function, Z_i are residuals of the model, p- number of model parameters. The criterion suggests that the model is optimal when the AIC is minimized.

Definition 6. *Let $\{X_t\}$ be a time series which is ARMA process. Its **periodogram** is defined as:*

$$I_{n,X}(\lambda) = |\frac{1}{n^{1/\alpha}} \sum_{i=1}^{n} X_t e^{-i\lambda t}|^2, \tag{6.13}$$

where $-\pi < \lambda < \pi$

To estimate the parameters of ARMA model **Whittle estimator** $\overline{\beta_n}$ is used [123]. The method is universal as it does not require any assumptions about residuals' distribution. The estimator is defined as a value which minimises following function:

$$\hat{\sigma}_n^2(\beta) = \frac{1}{n} \sum_j \frac{I_{n,X}(\lambda_j)}{g(\lambda_j, \beta)}, \tag{6.14}$$

where $\lambda_j = \frac{2\pi j}{n} \in (-\pi, \pi]$, $\beta = (\phi_1, ..., \phi_p, \theta_1, ..., \theta_q)$- ARMA$(p, q)$ estimated parameters, $g(\lambda, \beta) = \frac{|1+\sum_{k=1}^{q} \theta_k e^{-i\lambda k}|^2}{|1-\sum_{k=1}^{p} \phi_k e^{-i\lambda k}|^2}$, $-\pi < \lambda < \pi$.

The parameters of residuals distribution is obtained from following characteristic function relation [115, 120]:

$$\log(-\log|\phi_X(t)|^2) = \log(2\sigma^\alpha) + \alpha \log|t|, \tag{6.15}$$

where $\phi_X(t)$- characteristic function of $X \sim S_{\alpha,\sigma,\beta,\mu}$.
The equation above depends only on α and σ which suggests to estimate parameters by regressing $y = \log(-\log|\phi_n(t)|^2)$ on $w = \log|t|$ in the model:

$$y_k = m + \alpha w_k + \epsilon_k, k = 1, 2, ..., K, \tag{6.16}$$

where $m = \log(2\sigma^\alpha), t_k$ is an appropriate set of real numbers (e.g. $t_k = \frac{\pi k}{25}, k = 1, 2, ..., K$, and K from $\{9, ..., 134\}$ [124]). When $\hat{\alpha}, \hat{\sigma}$ are obtained, the α, σ are fixed at these values, and the estimates $\hat{\beta}, \hat{\mu}$ can be obtained from

$$\arctan(\frac{Im(\phi(t))}{Re(\phi(t))}) = \mu t + \beta\sigma^\alpha \tan(\frac{\pi\alpha}{2}) sign(t)|t|^\alpha. \tag{6.17}$$

Next the regressions are repeated with $\hat{\alpha}, \hat{\sigma}, \hat{\beta}, \hat{\mu}$ as initial parameters. The iterations continue until previously fixed convergence criterion is satisfied.

To test the differences between the empirical distribution $F_n(x)$ and the estimated one $F(x; \theta)$ of residuals (θ- estimated parameters' vector) 5 different tests are used (Kolmogorov–Smirnov, Kuiper, Cramer–von Mises, modified Cramer–von Mises, Anderson–Darling). These tests are based on following statistics:

1. $D = max(D^+, D^-) = max(max_{1 \le i \le n}\{\frac{i}{n} - z_{(i)}\}, max_{1 \le i \le n}\{z_{(i)} - \frac{i-1}{n}\})$,

2. $V = D^+ + D^-$,
3. $W^2 = \sum_{i=1}^{n} \{z_{(i)}, -\frac{2i-1}{2n}\}^2 + \frac{1}{12n}$
4. $U^2 = W^2 - n(\frac{1}{n}\sum_{i=1}^{n} z_{(i)} - \frac{1}{2})^2$
5. $A^2 = -n - \frac{1}{n}\sum_{i=1}^{n}\{\log z_{(i)} + \log(1 - z_{(n+1-i)})\}$,

where $z_{(i)}, i = 1, ..., n$ is sample i-th order statistics.

We test the null hypothesis [115]:

$$H_0 : F_n(x) = F(x; \theta), \tag{6.18}$$

against the:

$$H_1 : F_n(x) \neq F(x; \theta). \tag{6.19}$$

Small values of test statistics T (any of previously proposed), for sample from distribution $F(x; \theta)$ confirm H_0, and the large values give evidence to reject the null hypothesis and accept H_1. The decision is to made upon the tests' p-value quantity which is defined as:

$$p - value = P(T \geq t), \tag{6.20}$$

where t is statistic's value for sample which distribution is estimated. Common p-value which borders acceptance and rejection of H_0 is 0,05. The probability is estimated with Monte Carlo simulations.

The complete block diagram how the temporary mass data is modelled and simulated with ARMA model with residuals from α-stable distribution is included in Fig. 6.25.

Application

In this report temporary mass signal of belt conveyor L212 is analyzed. The conveyor is located in Polkowice-Sieroszowice copper ore mine. The raw signal needs to be pre-processed due to outliers. In this report we present only a modeling result based on an exemplary part of the measurements dated from 15^{th} June to 30^{th} September. The chosen segment is presented in Fig. 6.26.

The signal was subjected to modified AIC criterion to obtain optimal ARMA order. The results can be seen in Tab. 6.3. The table also includes the parameters of optimal model.

Segment	1.
AR part order	1
MA part order	2
ϕ_1	0.8456
θ_1	-0.1081
θ_2	-0.1492

Table 6.3: Optimal ARMA order and estimated parameters

After the optimal ARMA orders and parameters are obtained, residual signal might be computed. Their sample autocorrelation and partial autocorrelation functions are included in Fig. 6.27.

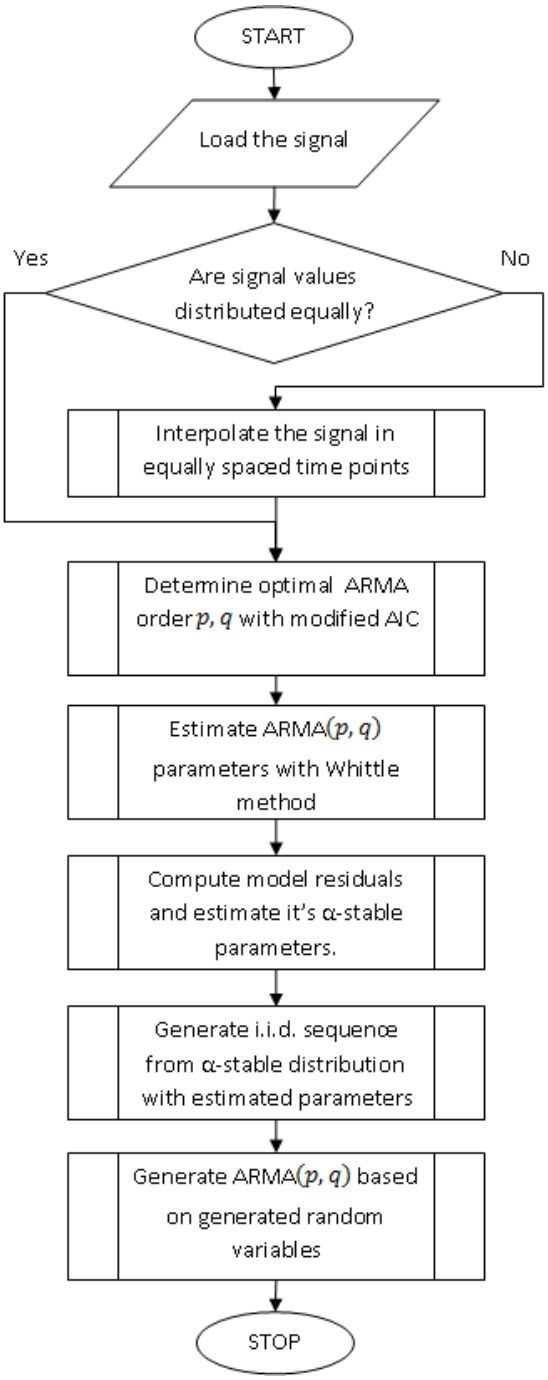

Figure 6.25: Flowchart of the estimation method for the mixture of the densities parameters

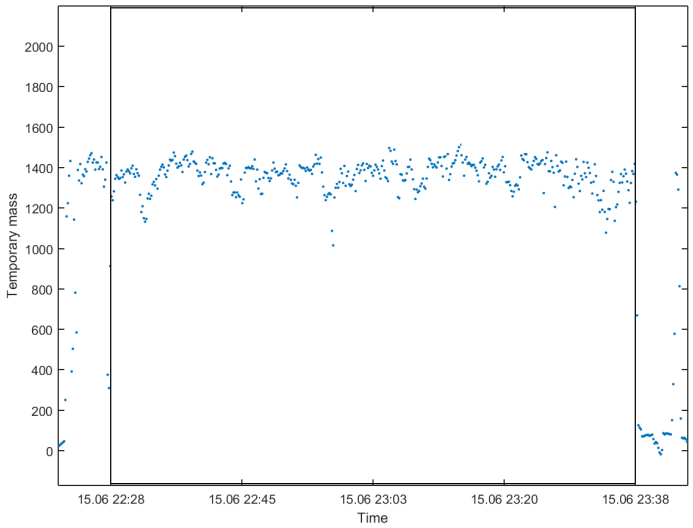

Figure 6.26: Chosen segment of temporary mass.

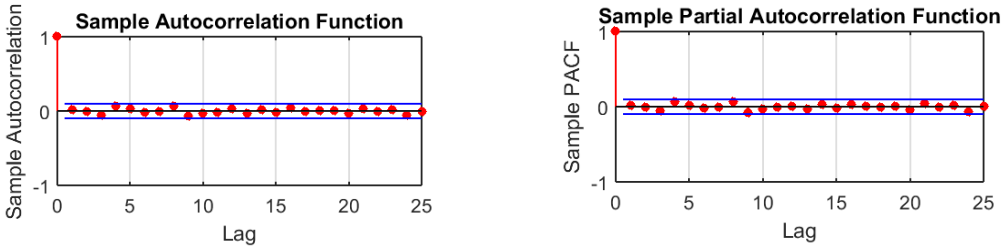

Figure 6.27: Sample autocorrelation and partial autocorrelation functions of the residuals

The analysis of Fig. 6.27 leads to a conclusion that for each segment residuals are sequence of independent random values. This suggest, that the ARMA models are fitted properly.

The QQ-plots of residuals empirical distribution (vs standard normal distribution) can be seen in Fig. 6.28.

Based on results presented in Fig. 6.28 one can observe that the residuals distribution possess heavier tails than the standard normal distribution. Thus it was determined to check whether the residuals belong to α-stable distribution which is characterized by heavy tails. In Tab 6.4 the p-values of 5 different tests are included.

The p-values in Tab. 6.4 do not give reasons that the residuals do not belong to α-stable distribution as all values are greater than 0.05. Thus, the α-stable distribution was fitted to residuals.

In Tab. 6.5 there are estimated α-stable parameters of residuals distribution. The parameters are estimated by regression method [115].

Based on results included in Table 6.5 one can observe that residuals are characterised by α parameter (index of stability) significantly lower than 2, which is indication of heavy tails. The

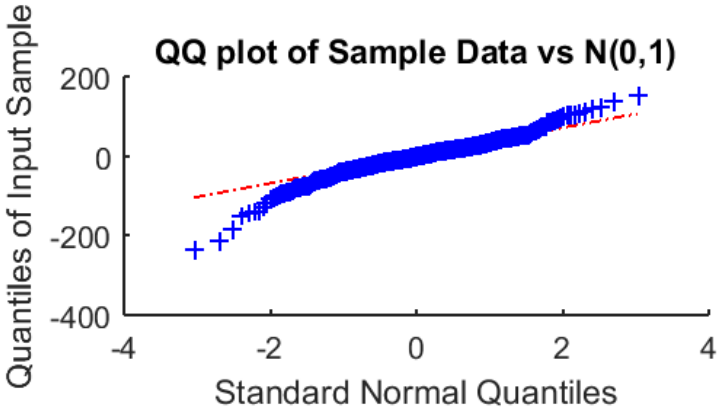

Figure 6.28: Residual quantiles vs quantiles of standard normal distribution

Table 6.4: *p*-values of 5 statistical tests, namely Kolmogorov–Smirnov, Kuiper, Cramer–von Mises, modified Cramer–von Mises, Anderson–Darling

Segment	1.
p-value D	0.6880
p-value V	0.6060
p-value W^2	0.6700
p-value U^2	0.5440
p-value A^2	0.3980

Table 6.5: Estimated α-stable parameters

Segment	1.
α	1.6548
σ	25.1136
β	-0.2498
μ	0.3557

scale parameter that is higher than 25. All distributions are left-skewed. Considering high scale parameters, location parameters are nearly close to 0.

Statistical analysis proved that the ARMA model with α-stable innovations is a suitable model for the considered signal representing ore mass flow on the conveyor supplied by the bunker. Such model and its parameters can be exploited in the future as a benchmark for the general ore transportation model. Comprehensive analysis of more signals related to the conveyor supplied by the bunker is performed in deliverable D5.1.

Conclusion

In this report problem of modelling the retention bunker originated temporary mass with ARMA process with residues from α-stable distribution was undertaken. The report is based on temporary mass signals from belt conveyor L212 that is placed in Polkowice-Sieroszowice copper ore mine. The signal was dated from 15^{th} June to 30^{th} September. Five different segments which were registered during the time when the retention bunker was opened and the temporary mass possessed relatively equal characteristic was chosen.

For each of these segments the optimal order of ARMA model was fitted. For received models residuals were computed and tested whether they are from α-stable distribution. The confirmation of these tests allowed to fit α-stable distribution to residuals. With fitted distribution it was possible to generate random variables from corresponding distribution. This allowed to simulate the process of temporary mass. Multiple iterations let the build of quantile lines which confirmed that the ARMA model with residuals from α-stable distribution precisely describe the temporary mass which originated from retention bunker. The fact that the result was repeated for each of the segments can be treated as an evidence that the retention bunker originated temporary mass can be modelled with ARMA process in general.

6.4 Key Performance Indicators

The most important KPI for the mine is the number of tonnes of ore produced per unit of time. The most important goal is, therefore, to maximise the production, given available equipment. The hoist is the bottleneck and it is important that the production is not limited by other parts in the transportation system. If the origin of the ore is known and the predicted time when it will come to the crusher and the ball mill it is possible to predict how close to maximum capacity the equipment will be operating, as the geological data contains information about the type of ore, its hardness and the distribution of the size of the ore.

More than 25% of the vehicles in the KGHM mine are RFID-tagged today (2016), and all new vehicles are RFID-tagged. When all the necessary equipment is in place and all LHD and truck transports can be tracked in the mine, then it is possible to track the ore from the faces to the conveyor belts. If the conveyor belt system models are available and able to describe the ore transport out of the mine and geological data are available where the ore is mined, then it is possible to predict the ore flow and also the ore contents all the way to the flotation.

A large number of transportation tasks are executed in parallel in the mine. If information about equipment, production and store levels is updated continuously to a control system, then

the tasks can be scheduled automatically. The schedules will not need to be completed as planned, as deviations between estimations and actual outcome with respect to time, moved ore, and due to machine breakdown etc. may occur. The model can be updated and new better schedules can be made automatically as information is fed back from the mine to the control system. The schedules can be made for longer time horizon, in order to get rid of apparently sub-optimal solutions, and still be able to react instantly, and propose better scheduling as response to every status update about the mine.

A secondary KPI is the *occupancy* of the conveyor belts and other active transportation element of the transportation network as detailed above. In case a conveyor belt is not occupied, it makes sense to switch it off in anticipation of new ore arriving. Of course it does not make sense to switch off the conveyor belts only for a few seconds, just to switch it off immediately afterwards, so the conveyor belt motors should be turned off only if they are to remain switched off for a minimum period of time, otherwise, not only do motors consume manifold amounts of energy at start up, but such an operation is likely to damage the motors. Overall, the objective is to decrease the amount of energy required to transport the ore. The occupancy of the conveyor belts will be determined in the percentage ($\%$) of time when they are loaded with respect to the time when they are running and the controller's goal will be to maximise the occupancy while respecting the imposed constraints (e.g., minimum off time). Clearly, such a control system should be able to have adequate foresight to predict the upcoming occupancy of the conveyor belts, so a dynamical system model is required.

6.5 Model predictive control

Since the mine is far from being a closed loop control problem today, it is hard to propose any classic MPC solution at this stage. The aim is to work on control from a more basic level that first focus on what is needed to increase the automation level in the control and to evaluate the different control strategies of mines with conveyor system and mines with trucks/trains where conveyors very often are a compliment to the trucks and trains.

6.6 Conclusions and future work

The work has initially focused on the development of a basic version of a modelling and simulation framework. Later, more accurate models can be added as well as simulation and finally control functionality. At this first stage, the focus was to make a modular framework that describes the material flow all the way from the face to the mill. Already in an early phase, the storage was identified as the main player to get the material flow under control.

CA has been initiated to get a first understanding of the possible modeling. The developed CA model seems to be working well and it can most likely be used to simulate data that can be used to develop a better basic model or as is in the framework, since the performance is promising. To get even more information around the storage modeling, also SPH is considered to be used later in the project. In the first modelling attempts of the KGHM mine, only limited process knowledge was used to get a fundamental, but hopefully from a modelling perspective

correct model structure of mine. The next step is to increase the collaboration with KGHM and to retrieve more information about the conveyor system, and then later receive real data, that can be used to develop better models and tune them to the Lubin mine. Up until know, the control problem of the material flow in the mine has just been evaluated loosely, to make sure that the model developed later are suitable for control. Even if the detailed investigation is yet to come we suspect, due to the limited instrumentation etc. in the mine, that it is not possible to get all the way to a closed loop MPC solution. Once more information around the KGHM process is gathered, next step in the development is to evaluate the best and most promising control solution for the KGHM conveyer system, as well as other underground mining processes.

Chapter 7

Conclusions and planned work

7.1 Conclusions

In this report we have presented various methodological tools for modelling and control, especially in a data-driven fashion as in Section 2.1; the algorithms we developed for data-driven system identification (and compressed sensing which is, in some sense, a dual problem to identification) are based on the assumption that the process data are available in a *stream*, rather than stored for offline batch processing. This makes such methodologies particularly useful for online adaptive identification which aligns with the spirit of DISIRE's data-driven proposition. Furthermore, we have shown that the proposed online methodologies are approximately one order of magnitude faster than all other existing streaming approaches — *a fortiori* faster than any batch processing approach.

We focused on the formulation and efficient algorithmic solution of data-drive algorithms such as streaming-recursive compressed sensing in Section 2.1.1, steaming online model identification in Section 2.1.2, data-driven scenario generation in Sections 3.1.1 and 3.1.2, and scenario-based stochaastic model predictive control in Section 3.2.

In Chapter 3 we presented a complete control scheme which is suitable for the control of the industrial processes of WP7 (walking beam furnace) and WP8 (NLPG cracking furnace). Stochastic model predictive control is an advanced control approach where the inevitable uncertainty which acts on the system is modelled using scenario trees and without any restrictive assumptions regarding the shape of their probability density functions (e.g., normality assumptions) or assumptions of independence or Markovianity. Instead, we pursue a completely data-driven approach to model these random processes which we then incorporate into the formulation of the MPC problem.

We applied the proposed methodologies in the case study of the walking beam furnace in Chapter 4 following all steps from data-driven control-oriented modelling to scenario generation, formulation and solution of stochastic MPC which was used to produce closed-loop trajectories in simulations. In Chapter 5, we provide some initial modelling results.

7.2 Planned work

WP2's plan of actions for the current and coming period will focus on the following items

1. How SMPC can be solved online very efficiently: we have been experimenting with a gradient-based algorithm which is based on the LBFGS method to solve scenario-based stochastic optimal control problems using the forward-backward envelope of the dual optimisation problem
2. Development of anomaly/fault detectors using conventional mutlivariate methods (Principal Component Analysis, Canonical Variate analysis) as well as machine learning methods (one class classifiers, autoencoder neural networks) for monitoring of the processes and application to the NLPGCF case study
3. Use of deep learning techniques for dynamic modelling of the existing processes (deep recurrent neural networks, Long Short Term Memory (LSTM) networks) and for state evaluation (Deep belief networks, Convilutional Neural Networks for the case of images etc)
4. Use of causality analysis for building more robust predictive models
5. Implement the algorithm of Task 1 on GPU and provide a solver for large-scale stochastic MPC problems
6. We are currently working on a proposition of a new risk-aware formulation of a stochastic model predictive control problem where the expectation operator (which is *risk-neutral*) is replaced with a *risk-sensitive* operator (such as the average value at risk). We will propose a computationally tractable formulation of this *risk-averse MPC* problem along with a solver and provide stochastic stability conditions
7. Perform further simulations with SMPC for WP7 and WP8: demonstrate the suitability of the SMPC control approach and quantify its benefits using KPIs
8. Continue our initial modelling exercise using nonlinear machine-learning-based models for the processes of WP7 and WP8, especially to the extent that we will receive data from new campaigns; validate these models and provide pertinent statistics
9. Develop models which will allow us to predict the average concentration of Oxygen (or Oxygen excess) and the concentration of carbon monoxide for the combustion processes in WPs 7 and 8 using machine learning techniques
10. Perform such modelling exercises on the cloud-based system designed and implemented by G-stat
11. Use PAT data to reconfigure the SMPC controllers based on existing machine learning models and the ones that will be produced by planned task 3
12. Revisit the control problem formulation for WP5
13. Calibrate the bunker mass flow models in Chapter 6 using measurements for the mine using RFID tags

In Table 7.1 we outline the distribution of the planned work per partner (task numbers refer to the above enumeration).

Task No.	*Partners*
1	IMTL, LTU
2	LTU
3	LTU, IMTL
4	LTU
5	IMTL, ODYS
6	IMTL
7	IMTL, LTU, ODYS
8	LTU, IMTL (Campaigns by DOW Chemical and MEFOS)
9	LTU, IMTL
10	ODYS, IMTL, LTU
11	IMTL, LTU
12	WUT, KGHM-CUP, ABB, IMTL
13	ABB, KGHM-CUP, WUT

Table 7.1: Distribution of planned work per partner.

Bibliography

[1] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, pp. 1289–1306, 2006.

[2] E. Candès and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[3] E. Candès and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[4] S. Qaisar, R. Bilal, W. Iqbal, M. Naureen, and S. Lee, "Compressive sensing: From theory to applications, a survey," *Journal of Communications and Networks*, vol. 15, no. 5, pp. 443–456, 2013.

[5] B. Zhang, X. Yin, W. Wang, and Q. Lei, "Research on image processing with compressed sensing algorithm: Base on the improved layered discrete cosine transform," in *Communications (ICC), 2012 IEEE International Conference on*, pp. 6357–6361, June 2012.

[6] Y. Rivenson and A. Stern, "Compressive sensing techniques in holography," in *Information Optics (WIO), 2011 10th Euro-American Workshop on*, pp. 1–2, June 2011.

[7] N. Vo, D. Vo, S. Challa, and B. Moran, "Compressed sensing for face recognition," in *Computational Intelligence for Image Processing, 2009. CIIP '09. IEEE Symposium on*, pp. 104–109, March 2009.

[8] Y. Wiaux, L. Jacques, G. Puy, A. Scaife, and P. Vandergheynst, "Compressed sensing imaging techniques for radio interferometry," tech. rep., 2008.

[9] C.-Y. Chen and P. P. Vaidyanathan, "Compressed sensing in mimo radar," in *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pp. 41–44, Oct 2008.

[10] D. Materassi, G. Innocenti, L. Giarr, and M. Salapaka, "Model identification of a network as compressing sensing," *Systems & Control Letters*, vol. 62, no. 8, pp. 664 – 672, 2013.

[11] N. Freris, O. Öçal, and M. Vetterli, "Recursive Compressed Sensing," tech. rep., 2014.

[12] R. Tibshirani, "Regression Shrinkage and Selection via the LASSO," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, pp. 267–288, 1996.

[13] E. Candès and Y. Plan, "Near-ideal model selection by ℓ_1 minimization," *The Annals of Statistics*, vol. 37, pp. 2145–2177, 2009.

[14] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale ℓ_1-regularized least squares," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–617, 2007.

[15] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 123–231, 2013.

[16] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[17] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, "Sparse Reconstruction by Separable Approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[18] M. Afonso, J. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2345–2356, 2010.

[19] N. Freris, O. Öçal, and M. Vetterli, "Compressed Sensing of Streaming data," in *Proceedings of the 51st Allerton Conference on Communication, Control and Computing*, pp. 1242–1249, Oct. 2013.

[20] P. Patrinos and A. Bemporad, "Proximal Newton methods for convex composite optimization," in *IEEE Conference on Decision and Control*, (Florence, Italy), pp. 2358–2363, 2013.

[21] P. Patrinos, L. Stella, and A. Bemporad, "Forward-backward truncated Newton methods for convex composite optimization," tech. rep., 2014.

[22] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, pp. 1168–1200, jan 2005.

[23] J. Eckstein and D. P. Bertsekas, "On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, pp. 293–318, apr 1992.

[24] H. H. Bauschke, M. N. Dao, and W. M. Moursi, "The douglasrachford algorithm in the affine-convex case," *Operations Research Letters*, pp. –, 2016.

[25] F. Facchinei and J.-S. Pang, eds., *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer New York, 2004.

[26] Y. Nesterov, "Gradient methods for minimizing composite functions," *Mathematical Programming*, vol. 140, no. 1, pp. 125–161, 2013.

[27] R. J. Tibshirani, "The Lasso problem and uniqueness," tech. rep., 2012.

[28] L. Xiao and T. Zhang, "A proximal-gradient homotopy method for the sparse least-squares problem," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 1062–1091, 2013.

[29] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[30] L. Ljung, *System identification: theory for the user*. Prentice-Hall Englewood Cliffs, NJ, 1999.

[31] A. Bemporad, D. Bernardini, and P. Patrinos, "A convex feasibility approach to anytime model predictive control," tech. rep., IMT Institute for Advanced Studies, Lucca, Feb. 2015. `http://arxiv.org/abs/1502.07974`.

[32] K. Bennett and O. Mangasarian, "Multicategory discrimination via linear programming," *Optimization Methods and Software*, vol. 3, pp. 27–39, 1994.

[33] Y. Lee and O. Mangasarian, "SSVM: A smooth support vector machine for classification," *Computational Optimization and Applications*, vol. 20, pp. 5–22, 2001.

[34] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, "A clustering technique for the identification of piecewise affine systems," *Automatica*, vol. 39, no. 2, pp. 205–217, 2003.

[35] J. Roll, A. Bemporad, and L. Ljung, "Identification of piecewise affine systems via mixed-integer programming," *Automatica*, vol. 40, no. 1, pp. 37–50, 2004.

[36] H. Nakada, K. Takaba, and T. Katayama, "Identification of piecewise affine systems based on statistical clustering technique," *Automatica*, vol. 41, no. 5, pp. 905–913, 2005.

[37] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino, "A bounded-error approach to piecewise affine system identification," *IEEE Transactions on Automatic Control*, vol. 50, no. 10, pp. 1567–1580, 2005.

[38] A. L. Juloski, S. Weiland, and W. P. M. H. Heemels, "A bayesian approach to identification of hybrid systems," *IEEE Transactions on Automatic Control*, vol. 50, no. 10, pp. 1520–1533, 2005.

[39] L. Bako, "Identification of switched linear systems via sparse optimization," *Automatica*, vol. 47, no. 4, pp. 668–677, 2011.

[40] H. Ohlsson and L. Ljung, "Identification of switched linear regression models using sum-of-norms regularization," *Automatica*, vol. 49, no. 4, pp. 1045–1050, 2013.

[41] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal, "Identification of hybrid systems a tutorial," *European journal of control*, vol. 13, no. 2, pp. 242–260, 2007.

[42] S. Alexander and A. Ghirnikar, "A method for recursive least squares filtering based upon an inverse QR decomposition," *IEEE Trans. Signal Processing*, vol. 41, no. 1, pp. 20–30, 1993.

[43] D. Ruppert, "Efficient estimations from a slowly convergent Robbins-Monro process," tech. rep., Cornell University Operations Research and Industrial Engineering, 1988.

[44] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*, pp. 421–436, Springer, 2012.

[45] W. Xu, "Towards optimal one pass large scale learning with averaged stochastic gradient descent," *arXiv preprint arXiv:1107.2490*, 2011.

[46] A. Bemporad, V. Breschi, and D. Piga, "Piecewise affine regression via recursive multiple least squares and multicategory discrimination," tech. rep., IMT Institute for Advanced Studies, Lucca, Oct. 2015. Submitted to Automatica. Available at: `http://www.dariopiga.com/TR/TR_PWAReg_BBP_2015.pdf`.

[47] V. Breschi, A. Bemporad, and D. Piga, "Identification of hybrid and linear parameter varying models via recursive piecewise affine regression and discrimination," tech. rep., IMT Lucca, Oct. 2015. To be presented at the European Control Conference 2016. `http://www.dariopiga.com/TR/TR_PWAReg_BBP_ECC_2016.pdf`.

[48] A. Bemporad, V. Breschi, and D. Piga, "Piecewise Affine Regression via Recursive Multiple Least Squares and Multicategory Discrimination," tech. rep., TR-IMT-DYSCO-2016-01, 2016. Available online at: `http://www.dariopiga.com/TR/TR-IMT-DYSCO-2016-01.pdf`.

[49] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2002.

[50] F. Lauer and Y. Guermeur, "MSVMpack: a multi-class support vector machine package," *Journal of Machine Learning Research*, vol. 12, pp. 2269–2272, 2011. `http://www.loria.fr/~lauer/MSVMpack`.

[51] Y. Lee, Y. Lin, and G. Wahba, "Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data," *J. American Statistical Association*, vol. 99, pp. 67–81, 2004.

[52] M. van de Wal and B. de Jager, "A review of methods for input/output selection," *Automatica*, vol. 37, no. 4, pp. 487 – 510, 2001.

[53] E. Bristol, "On a new measure of interaction for multivariable process control," *Automatic Control, IEEE Transactions on*, vol. 11, pp. 133–134, Jan 1966.

[54] M. E. Salgado and A. Conley, "MIMO interaction measure and controller structure selection," *International Journal of Control*, vol. 77, pp. 367–383, mar 2004.

[55] W. Birk and A. Medvedev, "A note on gramian-based interaction measures," in *Proc. of the European Control Conference 2003, University of Cambridge, UK*, 2003.

[56] B. Halvarsson, "Comparison of some gramian based interaction measures," in *Computer-Aided Control Systems, 2008. CACSD 2008. IEEE International Conference on*, pp. 138–143, Sept. 2008.

[57] M. Castao and W. Birk, "New methods for interaction analysis of complex processes using weighted graphs," *Journal of Process Control*, vol. 22, no. 1, pp. 280 – 295, 2012.

[58] V. Volterra, *Theory of Functionals and of Integral and Integro-differential Equations*. Dover Books on Mathematics Series, Dover Publications, 2005.

[59] J.-W. Chang and C.-C. Yu, "The relative gain for non-square multivariable systems," *Chemical Engineering Science*, vol. 45, no. 5, pp. 1309 – 1323, 1990.

[60] J. Chen, J. Freudenberg, and C. Nett, "On relative gain array and condition number," in *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pp. 219 –224 vol.1, 1992.

[61] T. J. McAvoy, "A methodology for screening level control structures in plantwide control systems," *Computers and Chemical Engineering*, vol. 22, no. 11, pp. 1543 – 1552, 1998.

[62] T. McAvoy and R. Miller, "Incorporating integrating variables into steady-state models for plantwide control analysis and design," in *Industrial & Engineering Chemistry Research*, vol. 38, pp. 412,420, American Chemical Society, February 1999.

[63] Y. Arkun and J. Downs, "A general method to calculate input-output gains and the relative gain array for integrating processes," *Computers & Chemical Engineering*, vol. 14, no. 10, pp. 1101 – 1110, 1990.

[64] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control*. John Wiley and Sons, second ed., 2005.

[65] M. Witcher and T. McAvoy, "Interacting control systems: Steady-state and dynamic measurement of interaction," *ISA Transactions*, vol. 16, no. 3, pp. 35–41, 1977.

[66] B. Wittenmark and M. E. Salgado, "Hankel-norm based interaction measure for input-output pairing," in *Proc. of the 2002 IFAC World Congress, Barcelona*, 2002.

[67] P. Vuillemin, C. Poussot-Vassal, and D. Alazard, "A spectral expression for the frequency-limited h2-norm," *arXiv preprint arXiv:1211.1858*, 2012.

[68] M. C. Arranz, W. Birk, and P. Asplund, "Control configuration selection for integrating processes using graphs," in *2015 IEEE Conference on Control Applications (CCA)*, pp. 1606–1611, Sept 2015.

[69] P. Patrinos, P. Sopasakis, H. Sarmiveis, and A. Bemporad, "Stochastic model predictive control for constrained discrete-time Markovian switching systems," *Automatica*, vol. 50, pp. 2504–2514, October 2014.

[70] E. Çinlar, *Probability and Stochastics*. Springer New York, 2011.

[71] G. C. Pflug and A. Pichler, *Multistage Stochastic Optimization*. Springer International Publishing, 2014.

[72] H. Heitsch and W. Römisch, "Scenario reduction algorithms in stochastic programming," *Computational optimization and applications*, vol. 24, no. 2, pp. 187 – 206, 2003.

[73] J. Dupačová, G. Consigli, , and S. Wallace, "Scenarios for multistage stochastic programs," *Annals of Operations Research*, vol. 100, no. 1, pp. 25–53, 2000.

[74] K. Frauendorfer, "Barycentric scenario trees in convex multistage stochastic programming," *Mathematical Programming*, vol. 75, no. 2, pp. 277–293, 1996.

[75] A. Shapiro, "Inference of statistical bounds for multistage stochastic programming problems," *Mathematical Methods of Operations Research*, vol. 58, no. 1, pp. 57 – 68, 2003.

[76] R. Karuppiah, M. Martín, and I. E. Grossmann, "A simple heuristic for reducing the number of scenarios in two-stage stochastic programming," *Computers & Chemical Engineering*, vol. 34, no. 8, pp. 1246 – 1255, 2010.

[77] K. Høyland and S. Wallace, "Generating scenario trees for multistage decision problems," *Management Science*, vol. 47, no. 2, pp. 295 – 307, 2001.

[78] D. Xu, Z. Chen, and L. Yang, "Scenario tree generation approaches using k-means and LP moment matching methods," *Journal of Computational and Applied Mathematics*, vol. 236, no. 17, pp. 4561 – 4579, 2012.

[79] T. Pennanen and M. Koivu, "Epi-convergent discretizations of stochastic programs via integration quadratures," *Numerische mathematik*, vol. 100, no. 1, pp. 141–163, 2005.

[80] N. Gröwe-Kuska, H. Heitsch, and W. Romisch, "Scenario reduction and scenario tree construction for power management problems," in *IEEE Power Tech Conference Proceedings*, vol. 3, (Bologna, Italy), p. 7, 2004.

[81] H. Heitsch and W. Römisch, "Scenario tree modeling for multistage stochastic programs," *Mathematical Programming*, vol. 118, no. 2, pp. 371–406, 2009.

[82] C. Hans, P. Sopasakis, A. Bemporad, J. Raisch, and C. Reincke-Collon, "Scenario-based model predictive operation control of islanded microgrids," in *54 IEEE Conf. Decision and Control*, (Osaka, Japan), Dec 2015.

[83] G. Schildbach and M. Morari, "Scenario-based model predictive control for multi-echelon supply chain management," *European Journal of Operational Research*, vol. 252, no. 2, pp. 540 – 549, 2016.

[84] A. Sampathirao, P. Sopasakis, A. Bemporad, and P. Patrinos, "Distributed solution of stochastic optimal control problems on gpus," in *54 IEEE Conf. Decision and Control*, (Osaka, Japan), Dec 2015.

[85] M. Cannon, P. Couchman, and B. Kouvaritakis, *Assessment and Future Directions of Nonlinear Model Predictive Control*, ch. MPC for Stochastic Systems, pp. 255–268. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[86] I. Guyon, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[87] "Computational methods of feature selection," 2007.

[88] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, AAAI'92, pp. 129–134, AAAI Press, 1992.

[89] M. Robnik-Sikonja and I. Kononenko, "An adaptation of relief for attribute estimation in regression," in *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, (San Francisco, CA, USA), pp. 296–304, Morgan Kaufmann Publishers Inc., 1997.

[90] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Machine Learning*, vol. 53, no. 1, pp. 23–69, 2003.

[91] I. Kononenko and M. Šikonja, "Non-myopic feature quality evaluation with (R)ReliefF," *Computational Methods of Feature Selection*, pp. 169–191, 2008.

[92] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*. Academic Press, 2008.

[93] S. Theodoridis, A. Pikrakis, K. Koutroumbas, and D. Cavouras, *Introduction to Pattern Recognition: A Matlab Approach*. Academic Press, 2010.

[94] M. Wasikowski and X. w. Chen, "Combating the small sample class imbalance problem using feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1388–1400, Oct 2010.

[95] M. A. Maloof, "Learning when data sets are Imbalanced and when costs are unequal and unknown," in *International Conf. on Machine Learning*, 2003.

[96] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Mach. Learn.*, vol. 45, pp. 171–186, Oct. 2001.

[97] I. Guyon, "Practical feature selection: from correlation to causality. mining massive data sets for security," in *Advances in Data Mining, Search, Social Networks and Text Mining, and their Applications to Security*, pp. 27–43, 2008.

[98] P. Trnka, "Subspace identification methods," tech. rep., Czech Technical University in Prague, 2005. Available at `http://www.trnka.name/download/subspace/ThesisProposal.pdf`.

[99] S. Qin, "An overview of subspace identification," *Computers and Chemical Engineering*, vol. 30, pp. 1502–1513, 2006.

[100] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[101] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer Series in Statistics, New York, NY, USA: Springer New York Inc., 2001.

[102] D. Berberidis, V. Kekatos, and G. Giannakis, "Online censoring for large-scale regressions with application to streaming big data," *IEEE Transactions on Signal Processing*, 2016. To appear.

[103] V. Vapnik and V. Vapnik, *Statistical learning theory*, vol. 1. New York: Wiley, 1998.

[104] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, pp. 321–357, June 2002.

[105] *Principal Component Analysis*. Springer-Verlag, 2002.

[106] E. Frank and M. Hall, *A Simple Approach to Ordinal Classification*, pp. 145–156. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001.

[107] W. Chu and S. S. Keerthi, "New approaches to support vector ordinal regression," in *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, (New York, NY, USA), pp. 145–152, ACM, 2005.

[108] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, Nov. 2009.

[109] B. Chopard and M. Droz, *Cellular Automata Modelling of Physical Systems*. Cambridge: Cambridge University Press, 1998.

[110] J. Martinez and S. Masson, *Silos – Fundamentals of theory, behaviour and design*, ch. Lattice grain models. London: E. & F.N. Spon, 1998.

[111] H. Hallberg, "Approaches to modeling of recrystallization," *Metals*, vol. 1, pp. 16–48, 2011.

[112] V. Osinov, "A model of a discrete stochastic medium for the problems of loose material flow," *Continuum Mechanics and Thermodynamics*, vol. 6, pp. 51–60, 1994.

[113] S. Savage, *Disorder and Granular media*, ch. Disorder, diffusion and structure formation in granular flows. Amsterdam: Elsevier Science Publisher, 1993.

[114] J. Kozicki and J. Tejchman, "Simulations of flow patterns in silos with a cellular automaton: part 1," *Task quarterly*, pp. 81–102, 2004.

[115] P. Cizek, W. K. Härdle, and R. Weron, *Statistical tools for finance and insurance*. Springer Science & Business Media, 2005.

[116] D. M. Bates and D. G. Watts, "Nonlinear regression: iterative estimation and linear approximations," *Nonlinear Regression Analysis and Its Applications*, pp. 32–66, 1988.

[117] P. R. Tadikamalla, "A look at the burr and related distributions," *International Statistical Review/Revue Internationale de Statistique*, pp. 337–344, 1980.

[118] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*. Springer Science & Business Media, 2006.

[119] P. Billingsley, *Probability and Measure. Wiley Series in Probability and Mathematical Statistics*. Wiley New York, 1995.

[120] G. Samoradnitsky and M. S. Taqqu, *Stable non-Gaussian random processes: stochastic models with infinite variance*, vol. 1. CRC press, 1994.

[121] E. Parzen, K. Tanabe, and G. Kitagawa, *Selected Papers of Hirotugu Akaike*. Springer Science & Business Media, 2012.

[122] J. Smith, "The modified yule-walker method for alpha-stable time series models," *submitted*, 2016.

[123] T. Mikosch, T. Gadrich, C. Kluppelberg, and R. J. Adler, "Parameter estimation for arma models with infinite variance innovations," *The Annals of Statistics*, pp. 305–326, 1995.

[124] I. A. Koutrouvelis, "Regression-type estimation of the parameters of stable laws," *Journal of the American Statistical Association*, vol. 75, no. 372, pp. 918–928, 1980.