Doctoral Dissertation

Magnus Komperød

Empirical Modeling, State Estimation, and Process Control with Real-Life Applications to the Czochralski Crystallization Process

Magnus Komperød

Doctoral Dissertation at TUC 1:2012

Telemark University College
Faculty of Technology

Magnus Komperød

Empirical Modeling, State Estimation, and Process Control with Real-Life Applications to the Czochralski Crystallization Process

Thesis for the degree of Doctor Philosophiae

Telemark University College
Faculty of Technology

Telemark University College

Telemark University College
Faculty of Technology
Department of Electrical Engineering, Information Technology and Cybernetics
Postboks 203
N - 3901 Porsgrunn

www.hit.no

Doctoral Dissertations at TUC 1:2012

ISBN 978-82-7206-336-7 (print version)
ISBN 978-82-7206-337-4 (electronic version)
ISSN 0000-1111
Printed by the Copy Center at TUC - Bø

PhD Thesis

Empirical Modeling, State Estimation, and Process Control with Real-Life Applications to the Czochralski Crystallization Process

Magnus Komperød

September 29, 2011

Preface

This document is my PhD thesis, which is a partial fulfillment of the requirements for the degree Doctor of Philosophy (PhD) at Telemark University College, Norway. From August 1st 2008 to July 31st 2011, I held a PhD scholarship position at Østfold University College, Norway, which provided me salary during the PhD study.

The research of this PhD study has focused on data preprocessing, empirical modeling, state estimation for the purpose of measurement noise filtering, and process control. Real-life applications are emphasized, in particular applications to the Czochralski (CZ) crystallization process. The research on the CZ process is based on a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. The PhD thesis also includes work based on the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway. During this PhD study, two journal articles and five conference papers have been published. Two of the conference papers were drafted during my master thesis. These two papers were finished and presented at a conference during my PhD study. Hence, the two papers are partly achievements of the master thesis and partly achievements of the PhD study.

Xstrata Nikkelverk and Dr. Tor Anders Hauge are acknowledged for providing logged process data from the copper refining process at Xstrata Nikkelverk and for allowing these data to be used in two conference papers.

SINTEF Materials and Chemistry is acknowledged for giving access to the organization's CZ process and for allowing logged process data to be used in scientific publications. Dr. Eivind Johannes Øvrelid, Bendik Sægrov, and John Atle Bones have been most helpful. In particular Bones' contributions have been decisive for running experiments at the CZ process and for improving process sensors at the plant. Bones is currently taking his master degree, and he will work with the CZ process in his master thesis. I wish him the very best luck!

I am very grateful for the financial support of my PhD study from NorSun AS, Østfold Energi AS, and the Norwegian Research Council. This support has been used for conference fees, a publication fee, travels, books, and software.

I am very grateful to Østfold University College for providing me the PhD scholarship. The research leader at Faculty of Engineering, Dr. Ole Kristian Førrisdahl,

has been very kind and helpful. He has given me large freedom and flexibility, while following my work closely to ensure I had decent progress in my PhD study.

The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is very much appreciated. Although the cooperation with NorSun and Prediktor has not been very close, I knew I had excellent knowledge and experience to rely on when needed.

During the PhD study, I have published two articles in the open-access journal Modeling, Identification and Control (MIC). I have also published in total five papers at the conferences SIMS 2008, SIMS 2009, and SIMS 2010. I am very grateful to the reviewers of MIC and SIMS for their efforts and their constructive feedback. I also acknowledge the organizers of the SIMS conferences. I am particularly grateful to MIC and its editor, Professor Geir Hovland, for providing an open-access journal with a low publication fee.

I am very grateful to my supervisors during the PhD study. The main supervisor has been Professor Bernt Lie at Telemark University College. The co-supervisors have been Dr. Steinar Sælid and Dr. Helge Mordt, both at Prediktor AS. Professor Lie, Dr. Sælid, and Dr. Mordt were also my supervisors during my master thesis. Professor Lie was very important for me during my master thesis and the first year of my PhD study. Although I have mainly stood on my own feet during the last two years of the PhD study, Professor Lie has always been there when I needed him. When needing advise or having questions, his knowledge, experience, and helpfulness are priceless to any student.

Magnus Komperød

Sarpsborg, Norway
September 29th 2011

Summary

This PhD thesis presents research work within the field of systems and control engineering, with emphasis on applications to real-life processes, the Czochralski (CZ) crystallization process in particular. During the PhD study, two journal articles and five conference papers have been published. All seven publications are based on logged data from real-life processes or include examples based on such data. For four of the publications, logged process data are essential. The seven publications are referred to as Paper A through Paper G. The publications focus on data preprocessing, empirical modeling, process control, and state estimation for the purpose of noise filtering.

The Czochralski (CZ) crystallization process is a batch process that converts multicrystalline materials into monocrystalline materials, i.e. materials that have homogeneous crystal structures. Among the most important applications of the CZ process is production of monocrystalline silicon. This is the only application of the CZ process that has been considered during this PhD study. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Solar cells based on monocrystalline silicon have higher efficiency than those based on multicrystalline silicon.

During the CZ batch process, multicrystalline silicon is melted in a crucible. The silicon is then solidified on a monocrystalline seed crystal, thereby growing a crystal. The grown crystal is monocrystalline and is referred to as an ingot. There are several challenges associated with modeling and control of the CZ process: (i) The process dynamics is challenging to model using mechanistic (first principle) modeling. (ii) The process has multivariable character. (iii) The process is time-variant due to its batch nature. (iv) There are several difficulties regarding sensor technologies. In particular the ingot diameter is difficult to measure online.

The candidate's literature search indicates that most published research works considering modeling and control of the CZ process are simulation studies, which are not validated against real-life processes. Only one publication was found that documents that a suggested control strategy works on real-life CZ processes. During the PhD study, the candidate had access to a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. As published research results

that are validated on real-life CZ processes seem to be rather sparse, the candidate focused his research on experiments at this plant.

Unfortunately, issues regarding sensor technologies forced the candidate to focus on other parts of the CZ process than initially planned. However, these issues have also given useful experiences and provided ideas for further research. The work of this PhD study has focused on the heating element power and the temperature of the molten silicon. The ingot diameter has not been considered, partly because of unreliable diameter sensor, partly because the diameter depends on the silicon temperature. Hence, it is reasonable not to consider the ingot diameter until the heating element power and the silicon temperature are properly measured, modeled, and controlled.

Logged process data from the SINTEF CZ plant are used extensively during this PhD study. Paper D and Paper E consider empirical modeling of the heating element power, Paper F suggests a cascade control strategy for improving temperature control of the molten silicon, and Paper G presents state estimation for the purpose of measurement noise filtering. Also, logged process data from the SINTEF CZ plant are used as an example in Paper C.

Paper A and Paper B include work on logged process data from the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway. These data were made available from the process by Dr. Tor Anders Hauge. Paper A presents work on data preprocessing, using the Xstrata data as real-life examples. Paper B considers system identification and compares two system identification algorithms using process data from Xstrata. System identification is the science of developing dynamic, empirical models based on process inputs and the corresponding process outputs.

Paper A and Paper B were drafted during the candidate's master thesis and were published in the beginning of his PhD study. The other five publications were written during the PhD study. The following text briefly summarize each publication in the order they were published.

Paper A *Preprocessing of Experimental Data for Use in Model Building and Model Validation.* By Magnus Komperød, Tor Anders Hauge, and Bernt Lie. The paper was presented at the 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008) and is included in the conference's proceedings. The conference was held October 7th-8th 2008 at Oslo University College, Oslo, Norway.

Two issues within data preprocessing are considered:

(i) Ljung (1999) shows an example of how model residuals can be used for outlier detection. An alternative approach to this method is developed in Paper A. The approach is based on identification of the innovation process

directly from logged process data, i.e. without relying on process models. It can be mathematically shown that the suggested approach turns out to be a special case of the method presented in Ljung (1999). The suggested approach is tested on real-life process data from Xstrata Nikkelverk.

(ii) The MATLAB command `delayest` is included in the System Identification Toolbox. The command's purpose is to estimate time delays between process inputs and process outputs. Paper A shows both in a simulation study and on logged process data from Xstrata Nikkelverk that `delayest` is sensitive to several factors that limit its practical usefulness. Paper A suggests an improvement of `delayest` that handles one sensitivity issue better than the original method under certain ideal assumptions. However, also this improvement has very limited usage on real-life process data.

Paper B *Empirical Modeling: Approximating the DSR E Sub-Space System Identification Algorithm by a Two-Step ARX Algorithm.* By Magnus Komperød, Tor Anders Hauge, David Di Ruscio, and Bernt Lie. The paper was presented at the 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008) and is included in the conference's proceedings. The conference was held October 7th-8th 2008 at Oslo University College, Oslo, Norway.

This paper considers the DSR E system identification algorithm presented in Nilsen (2006) and Di Ruscio (2008). The DSR E algorithm is developed to give consistent model estimates both for process data logged in open loop and for process data logged in closed loop.

Paper B shows that the DSR E algorithm can be approximated by a two-step ARX algorithm. This two-step ARX algorithm is referred to as DARX. In addition to the mathematical reasoning, the DSR E and DARX algorithms are compared on real-life process data from Xstrata Nikkelverk. As expected, the two algorithms identify identical models.

It is emphasized that DARX is too similar to DSR E to be considered a "new" system identification algorithm. The purpose of DARX is to show the similarity between DSR E and a two-step ARX algorithm. From the candidate's point of view, DARX may be easier to understand and to implement than DSR E. DARX has been developed only for single input / single output (SISO) systems. It is believed that the algorithm can be extended to multiple input / multiple output (MIMO) systems. However, the candidate has not done any research on this topic.

Paper C *Solution to an Implementation Issue for a Two-Step ARX Algorithm, with Application to the Czochralski Crystallization Process.* By Magnus Komperød, John Atle Bones, and Bernt Lie. The paper was presented at

the 50th International Conference of Scandinavian Simulation Society (SIMS 2009) and is included in the conference's proceedings. The conference was held October 7th-8th 2009 at DONG Energy, Fredericia, Denmark.

The second ARX step of the DARX algorithm presented in Paper B slightly differs from the standard ARX form. This apparently prevents standard ARX software from being used, such as the command `arx` of the MATLAB System Identification Toolbox. A tailor-made ARX algorithm must then be implemented for using DARX. Implementing this tailor-made algorithm is not very difficult, but it significantly complicates the otherwise simple implementation of DARX.

Paper C presents a rewriting of the second ARX step of DARX that allows standard ARX software to be used. The paper compares various implementations of DARX by testing them on real-life process data from the CZ process at SINTEF. The results of these tests back up the mathematical derivation of the rewriting.

Paper D *Empirical Modeling of Heating Element Power for the Czochralski Crystallization Process.* By Magnus Komperød and Bernt Lie. The article is published in the open-access journal Modeling, Identification and Control (MIC).

In the CZ process at SINTEF, the crucible containing the molten silicon is heated by an electric heating element. The heating element power is manipulated by a triode for alternating current (TRIAC). There is a *dynamic* relationship between the TRIAC input signal (control system output) and the actual (measured) heating element power. The assumed reason for this dynamics is that when the TRIAC input signal is increased, more power is applied to the heating element, which increases the heating element's temperature over time. Increased temperature gives increased electric resistance, which decreases the power.

Paper D considers system identification of the dynamics from the TRIAC input signal to the measured heating element power. Both linear and nonlinear model structures are considered. To avoid overfitting the nonlinear model, significant effort is done to minimize the number of parameters to be identified. The best model identified is a Hammerstein model, i.e. a linear, dynamic transfer function, which input is processed through a static, nonlinear function.

Paper E *Adaptive System Identification of Heating Element Power for the Czochralski Crystallization Process.* By Magnus Komperød, John Atle Bones, and Bernt Lie. The paper was presented at the 51st Conference on Simulation

and Modelling (SIMS 2010) and is included in the conference's proceedings. The conference was held October 14th-15th 2010 in Oulu, Finland.

Paper E also considers system identification of the dynamics from the TRIAC input signal to the measured heating element power, i.e. the same dynamics as considered in Paper D. Paper E compares three linear ARMAX models in terms of the best one-step-ahead predictions. The first model is non-adaptive. The second model has four adaptive parameters. The parameters are adapted using the MATLAB command `rarmax` (Recursive ARMAX) of the System Identification Toolbox. This model is referred to as the adaptive ARMAX model. The third model has adaptive gain, while the pole, the zero, and the noise model are fixed. This model is referred to as the adaptive gain model.

The adaptive ARMAX model performs better than the non-adaptive model for some choices of the forgetting factor. However, the model's performance is very sensitive to the choice of the forgetting factor. The model's main disadvantage is that the model's pole crosses the unity circle as a result of the parameter adaptation. Hence, the model changes between being stable and unstable. This issue makes the model useless for most real-life applications.

The adaptive gain model outperforms the other models in terms of one-step-ahead predictions. This model's performance is very robust to the choice of the forgetting factor. As the pole is fixed, the model is always stable. This model can be implemented using the recursive least squares method, which is easy to implement and has numerical advantages.

Paper F *Rejection of Power Disturbances in the Czochralski Crystallization Process Using Cascade Control.* By Magnus Komperød, John Atle Bones, and Bernt Lie. The paper was presented at the 51st Conference on Simulation and Modelling (SIMS 2010) and is included in the conference's proceedings. The conference was held October 14th-15th 2010 in Oulu, Finland.

At the SINTEF CZ process, the crucible containing the molten silicon is heated by the heating element discussed above. To produce high crystal quality, tight control of the silicon temperature is most important. The temperature was initially controlled by a single-loop PID controller. The controlling element is the TRIAC input signal, which manipulates the heating element power.

During experiments at the SINTEF CZ process, process disturbances were discovered at the heating element power. That is, there are responses in the heating element power that can not be explained by the TRIAC input signal. These disturbances were sufficiently large to have a significant influence on the silicon temperature.

Paper F presents a cascade control strategy to effectively compensate the power disturbances. The inner loop (slave control loop) is a power control loop, which controls the heating element power using the TRIAC input signal as controlling element. The outer loop (master control loop) controls the silicon temperature. The temperature controller sets the reference (setpoint) to the power controller, which ensures that the power desired by the temperature controller is actually applied to the heating element, regardless of process disturbances. This cascade control strategy has been tested on the SINTEF CZ process. The power controller rejects the power disturbances quickly and effectively. A simulation study shows that the dynamics from the reference of the power controller to the measured heating element power is robust to parameter variations in the inner loop.

Paper G *A Sensor Fusion Algorithm for Filtering Pyrometer Measurement Noise in the Czochralski Crystallization Process.* By Magnus Komperød, John Atle Bones, and Bernt Lie. The article is published in the open-access journal Modeling, Identification and Control (MIC).

As discussed above, tight control of the temperature of the molten silicon is most important. At the CZ process at SINTEF, the temperature control is based on a pyrometer that measures the temperature of a graphite ring. This pyrometer is referred to as the graphite pyrometer. The pyrometer has little measurement noise, but it has the significant disadvantage that it does not measure the temperature of the molten silicon. Hence, it is the temperature of the graphite ring that is actually controlled, not the temperature of the molten silicon.

During one CZ batch another pyrometer was tested. This pyrometer measures the temperature of the molten silicon. This pyrometer is referred to as the silicon pyrometer. This pyrometer is assumed to be accurate, but it has much high-frequency measurement noise. There is quite a high correlation between the graphite pyrometer and the silicon pyrometer.

Paper G presents a sensor fusion algorithm that merges the two pyrometer signals. The algorithm produces a temperature estimate that has little measurement noise, while giving significant less phase lag than traditional lowpass-filtering of the silicon pyrometer. The algorithm consists of two subalgorithms: (i) A dynamic model is used to estimate the silicon temperature based on the graphite pyrometer, and (ii) a lowpass filter and a highpass filter designed as complementary filters. The complementary filters are used to lowpass filter the silicon pyrometer and highpass filter the output of the dynamic model. These filtered signals are then summed, giving the silicon temperature estimate.

Contents

Part I

Overview

Chapter 1

Introduction

1.1 Background

After a slight decrease in the world's energy consumption in 2009, due to the so-called financial crisis, the consumption increased by 5% in 2010 (Wikipedia, 2011). At the same time, mankind is worried about global warming, mainly due to combustion of fossil fuels. The skepticism to nuclear power is significantly increased after the earthquake and tsunami disaster in Japan, March 2011, which caused the second largest nuclear accident in human history at the Fukushima nuclear power plant.

Although it is not realistic to replace fossil fuel and nuclear power by renewable energy in the near future, increasing energy prices and environmental concerns favor development and industrial scale production of renewable energy sources. A very interesting technology in this respect is solar cells, which convert solar energy directly into electric energy.

Industrial scale energy production based on solar cells requires industrial scale production of solar cell panels. A main component of solar cell panels is silicon wafers. Silicon wafers are produced from either monocrystalline silicon or multicrystalline silicon. Monocrystalline materials are materials that have a homogeneous crystal structure through the entire material. Solar cells based on monocrystalline silicon have higher efficiency than solar cells based on multicrystalline silicon. Monocrystalline silicon is also used in computers and electronics.

The Czochralski (CZ) crystallization process is a batch process for producing monocrystalline materials. Among the process' most important applications is production of monocrystalline silicon. This is the only application of the CZ process considered during this PhD study. The CZ process produces a rod-shaped crystal that is referred to as an ingot. The produced ingot is cut radially to thin discs that are used for solar cell wafers and in computers and electronics.

Producing high crystal quality using the CZ process is associated with several challenges. These include challenges within the field of systems and control engineering: (i) The process dynamics is difficult to model by mechanistic (first principle) modeling. (ii) The process has multivariable character. (iii) The process is time-variant due to its batch nature. (iv) There are several difficulties regarding sensor technologies for online measurement of process variables.

This PhD thesis presents research work within systems and control engineering. Applications to the real-life process industry are emphasized, in particular applications to the CZ process. The research on the CZ process is based on a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. Also the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway, is considered in this PhD thesis. The work presented in the PhD thesis has focused on empirical modeling, process control, and state estimation used for measurement noise filtering.

1.2 Previous Work

A literature search reveals a large range of modeling approaches for the CZ process, from very simple empirical models to complex mechanistic (first principle) models. Lee et al. (2005) includes empirical models based on step responses. The process' time-varying character is handled by using five different sets of models. The models are alternated as the batch progresses.

Hurle (1993) presents a modeling work that results in simple transfer functions. As these transfer functions are simple and have few parameters, they may be subject to empirical parameter estimation. Such parameter estimation is not demonstrated in the article.

Irizarry-Rivera and Seider (1997a) presents a complex mechanistic (first principle) model. A main disadvantage of this work is that the model is not validated on real-life process data. Irizarry-Rivera and Seider (1997b) presents model reduction of the model presented in the former publication for the purpose of using the model for model predictive control (MPC). Irizarry-Rivera and Seider (1997b) also considers parameter estimation. Again the contributions are not validated on real-life CZ processes.

Park et al. (2008) presents work within mechanistic modeling of the heat transfer and the ingot diameter in the CZ process. The article includes a trend plot from a real-life CZ process and concludes that the model explains the real process well. However, from the candidate's point of view, both the real-life trend plot, the simulation trend plots, and the connection between these plots are insufficiently explained. It is therefore difficult for the reader to conclude how good the model actually is.

Gevelber et al. (2001) considers some issues within modeling and control of the CZ process based on logged process data from a real-life CZ batch. However, neither the modeling work nor the control work are validated on real-life CZ processes.

Only one of the papers considered during the literature search, Lee et al. (2005), documents that a suggested control strategy works at real-life CZ plants. This is a rather simple control strategy based on single-loop control, cascade control, and feedforward trajectories. The article's main contribution is a method for developing a target temperature trajectory for the CZ process.

Hurle (1993) discusses some approaches for measuring the ingot diameter. The candidate has not been successful in finding publications that cover sensor technologies for other process variables at the CZ process.

Except for Lee et al. (2005) and Park et al. (2008), no works on modeling and control of the CZ process were found that have been validated on real-life CZ processes. Lee et al. (2005) emphasizes the lack of advanced control strategies tested on real-life CZ processes: "Although there have been several research efforts to apply state of the art control techniques to the control of the CZ crystal growers ... no open report for the real implementation of the advanced techniques on the commercial growers has been available thus far."

The candidate did not find any published works on system identification of the CZ process, except for the models developed in Lee et al. (2005). System identification is the science of developing dynamic models empirically. The negative result of the search for system identification work can perhaps be seen in connection with the sparse amount of modeling works that have been validated on real-life CZ processes: Both the validation of mechanistic models and the development of empirical models depend on access to logged process data from real-life processes.

Summing up the results of the literature search, the main conclusion is that there seem to be few published research works on modeling and control of the CZ process that are validated on real-life processes.

Paper A and Paper B of this PhD thesis include work on process data from the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway. The candidate's only source of information regarding this plant is Hauge (2007) and personal communication with the author of this reference. Hauge (2007) is a presentation of the plant, which is not publicly available. The candidate has not searched for scientific publications considering copper refining processes in general, because this is not relevant for the work presented in Paper A and Paper B. Lie and Hauge (2008) and Alic et al. (2009) present modeling works of the copper refining process at Xstrata Nikkelverk. However, Lie and Hauge (2008) was presented at the same conference as Paper A and Paper B, and Alic et al. (2009) was presented later.

The candidate's work relies heavily on established disciplines within systems and control engineering. The literature covering these disciplines is extensive. The most important references for the candidate during his PhD study have been: Ljung (1999) gives a comprehensive introduction to system identification and provides an extensive number of references for further reading. Strang (2003, 2006) give good introductions to linear algebra, which is a very important topic in systems and control engineering. Chen (1999) and Rugh (1996) present linear system theory. Haugen (2004) covers many interesting and useful topics within dynamic systems. Brown and Hwang (1997) introduces random signals and Kalman filtering. Lyons (2011) gives an introduction to digital signal processing. Process control is covered in Seborg et al. (2004) and Haugen (1994, 2001). The latter two books are in Norwegian.

1.3 Main Contributions

There have been several issues regarding sensor technologies at the SINTEF CZ process. These issues forced the candidate to focus on other parts of the CZ process than he would have done otherwise. However, the issues also gave valuable experiences and were the motivation behind the research presented in Paper G. The main sensor technology issues are the sensor for the temperature of the molten silicon and the sensor for the ingot diameter. Also the power of the heating element, which heats the molten silicon, is associated with significant sensor weaknesses. The candidate's research includes work on the silicon temperature measurement. A SINTEF engineer and the candidate did some search to find a more reliable sensor for the heating element power. A better sensor was found, however, due to time constraints, the sensor was not replaced during the PhD study. The ingot diameter measurement has not been considered.

The research work on the SINTEF CZ process has focused on the heating element power and the temperature of the molten silicon. The ingot diameter has not been considered, partly due to its sensor technology issue, and partly because it depends on the silicon temperature. Hence, it is reasonable to finish the work on measurement and control of the heating element power and the silicon temperature, before the ingot diameter is considered. The main contributions of this PhD thesis are:

1. Ljung (1999) shows an example of how model residuals can be used for outlier detection. An alternative approach to this method is developed in this PhD thesis. This approach is based on identification of the innovation process directly from logged process data, i.e. without relying on process models. Please refer to Paper A for an explanation of the term innovation process. It

can be mathematically shown that the suggested approach turns out to be a special case of the method presented in Ljung (1999). This work is presented in Paper A.

Outliers in datasets can have very negative effects on the quality of empirical models identified from these datasets. Such outliers will also affect validation of models against logged process data. The latter issue affects all kinds of models, not only empirical models.

This method for outlier detection is applied on process data from Xstrata Nikkelverk and from the SINTEF CZ process in Paper A and Paper D, respectively. The candidate assumes that the method can be used on process data from any industry, as well as other applications where some measurable variables are logged at fixed time intervals. However, the method assumes that an autoregressive model (AR or ARX model) can give reasonable good one-step-ahead predictions of the measured variables.

As the suggested method turns out to be a special case of a method presented in Ljung (1999), the suggested method is mainly of academic interest. Depending on what software is available, the suggested method may be somewhat easier to implement, because it identifies the innovation process directly, without relying on process models.

From the candidate's point of view, the demonstrations of the suggested method on real-life process data from Xstrata Nikkelverk and the SINTEF CZ process are useful contributions, as Ljung (1999) only presents a simulation study. These real-life examples illustrate the usefulness of the method, and they will hopefully inspire the readers to have outliers in mind when working on real-life process data.

2. The command `delayest` is included in the MATLAB System Identification Toolbox. The command estimates time delays between process inputs and process outputs. This PhD thesis shows in a simulation study and on process data from Xstrata Nikkelverk that the command is sensitive to several factors, which limit its practical usefulness. The PhD thesis suggests an improvement of `delayest` that handles one sensitivity issue better than the original method under certain ideal assumptions. However, also this improvement has very limited usefulness on real-life data. This work is presented in Paper A.

   From the candidate's point of view, it is important for the users of the MATLAB System Identification Toolbox to be aware of the weaknesses of the command `delayest`. The candidate advises to validate the time delay estimates computed by `delayest` against process knowledge or against other methods for time delay estimation.

Paper G also include some discussion on time delay estimation, using another estimation approach than `delayest`. The method used in Paper G also gives variable time delay estimates depending on parameters that ideally should not influence the estimates.

3. DSR E is a system identification algorithm developed to give consistent parameter estimates both for process data logged in open loop and for process data logged in closed loop. This PhD thesis shows that the DSR E algorithm can be approximated by a two-step ARX algorithm. This two-step ARX algorithm is referred to as DARX. In addition to the mathematical reasoning, the DSR E and DARX algorithms are compared on real-life process data from Xstrata Nikkelverk. This work is presented in Paper B.

   From the candidate's point of view, DARX may be easier to understand and to implement than DSR E. DARX has been developed only for single input / single output (SISO) systems. It is believed that the algorithm can be extended to multiple input / multiple output (MIMO) systems. However, the candidate has not done any research on this topic.

   The second ARX step of the DARX algorithm slightly differs from the standard ARX form. This apparently prevents standard ARX software, such as the command `arx` of the MATLAB System Identification Toolbox, from being used. A tailor-made ARX algorithm must then be implemented in order to use DARX. Implementing this tailor-made algorithm is not very difficult, but it significantly complicates the otherwise simple implementation of DARX. This PhD thesis presents a rewriting of the second ARX step of DARX that allows standard ARX software to be used. To validate that the rewriting is correct, various implementations of the DARX algorithm are used to identify a transfer function at the SINTEF CZ process. This work is presented in Paper C.

   The purpose of DARX is mainly to show the similarity between DSR E and a two-step ARX method. DARX is too similar to DSR E to be considered a new system identification algorithm. DARX is also quite similar to a system identification algorithm presented in Ljung (1999).

4. At the CZ process at SINTEF, the crucible containing the molten silicon is heated by an electric heating element. The heating element power is manipulated by a triode for alternating current (TRIAC). There is a dynamic relationship between the TRIAC input signal (control system output) and the actual (measured) heating element power. In this PhD thesis system identification is used to model the dynamics from the TRIAC input signal to the measured heating element power. Paper D presents modeling work

based on linear and nonlinear system identification. Paper E presents work on adaptive system identification.

The assumed reason for the dynamics from the TRIAC input signal to the measured heating element power is: When the TRIAC input signal is increased, the heating element power increases instantaneously. There is a time constant from the heating element power to the heating element temperature. As the temperature increases, the electric resistance also increases, which decreases the electric power.

The candidate has not seen this dynamics been described in other scientific publications. If the dynamics has not been considered before, it is assumed to be of some interest for the CZ research and the CZ industry. Including this dynamics in the modeling work will affect the model order of the dynamics from the TRIAC input signal to the temperature of the molten silicon.

5. At CZ processes, tight control of the temperature of the molten silicon is most important for achieving high crystal quality. At SINTEF the temperature was initially controlled by a single-loop PID controller. The controlling element was the TRIAC input signal, which manipulates the heating element power. Experiments at SINTEF reveal significant process disturbances to the heating element power. These disturbances are most unfortunate for the temperature control, and hence for the crystal quality.

   This PhD thesis presents a cascade control strategy for compensating the power disturbances quickly and effectively. The inner loop (slave control loop) is a power control loop, which controls the heating element power using the TRIAC input signal as controlling element. The outer loop (master control loop) controls the temperature. The temperature controller sets the reference (setpoint) to the power controller, which ensures that the power requested by the temperature controller is actually applied to the heating element, regardless of process disturbances. This work is presented in Paper F.

   Lee et al. (2005) is the only publication found that documents that a suggested control strategy works on real-life CZ processes. Control of the heating element power is not discussed in this article, and is not included in a process and instrument diagram (P&ID) that illustrates the applied control strategy. Hence, to the candidate's knowledge, Paper F is the only published work where control of the heating element power is tested on a real-life CZ process.

6. At the SINTEF CZ process a pyrometer is used as temperature sensor for the purpose of controlling the temperature of the molten silicon. However,

this pyrometer measures actually the temperature at a graphite ring, not in the molten silicon. Hence, the temperature of the graphite ring is actually controlled, not the temperature of the silicon. This pyrometer is referred to as the graphite pyrometer.

During an experiment at SINTEF, a new pyrometer was tested. This pyrometer measures the temperature directly in the molten silicon. The output signal of the pyrometer seems reasonable based on the melting point of silicon, and based on measured temperature responses after steps in the TRIAC input signal. Unfortunately, the pyrometer output signal has much high-frequency measurement noise. This pyrometer is referred to as the silicon pyrometer.

This PhD thesis presents a sensor fusion algorithm that takes the output signals of the two pyrometers as inputs and computes an estimate of the silicon temperature. The computed estimate attenuates the measurement noise of the silicon pyrometer, while giving significant less phase lag than traditional lowpass filtering. This work is presented in Paper G.

The candidate has not been able to find any scientific publications that present work on attenuation of pyrometer measurement noise at the CZ process. Hence, the algorithm presented in Paper G may be unique in its application.

Papers A through G are presented in Part II of this PhD thesis. Papers A, B, C, E, and F were presented at conferences and are included in the conferences' respective proceedings. Papers D and G are published in the open-access journal Modeling, Identification and Control (MIC). Papers A and B were drafted during the candidate's master thesis (Komperød (2008)). These papers were finished and presented at a conference during the PhD study. The basic idea of Paper C was also presented in the candidate's master thesis. Paper C was written and presented at a conference during the PhD study. For Papers D to G all the candidate's efforts were done during the PhD study. The candidate has not reached any significant results that are not presented in Papers A through G.

From the candidate's point of view, the most important work that has *not* been considered during this PhD study is modeling of the dynamics (i) from the heating element power to the silicon temperature, (ii) from the silicon temperature to the ingot diameter, and (iii) from the ingot pulling speed to the ingot diameter. When these dynamics are modeled, the models can be used for process control. The sensor technology issues discussed above are the reason for not including this research work in the PhD study.

Chapter 2

The Czochralski Crystallization Process

The Czochralski (CZ) crystallization process is a batch process for converting multicrystalline materials into monocrystalline materials. A monocrystalline material has a homogeneous crystal structure. The CZ process is named after its inventor, the Polish scientist Jan Czochralski, who discovered the method in 1916. Lan (2004) classifies the CZ process as one of three groups of melt growth technologies. The other two groups are the Bridgman method and the zone-melting method.

Among the most important applications of the CZ process is production of monocrystalline silicon. This PhD thesis considers only this application of the CZ process. Monocrystalline silicon is used for solar cell wafers and in computers and electronics. Solar cells based on monocrystalline silicon have higher efficiency than those based on multicrystalline silicon.

The work on the CZ process included in this PhD thesis considers empirical modeling (Paper D and Paper E), process control (Paper F), and state estimation for the purpose of noise filtering (Paper G). In Paper C real-life process data from the CZ process are used to compare different implementations of a system identification algorithm. Mechanistic (first principle) modeling of the CZ process is not considered in this PhD thesis. Section 1.2 presents some references on mechanistic modeling of the CZ process.

2.1 Principle of Operation

Figure 2.1 illustrates the CZ batch process. The main components in the figure are: (i) The crucible is a container holding the silicon. (ii) The seed crystal is used to initiate the crystal growth. The seed crystal has the crystal structure that is to be produced. (iii) The produced crystal, referred to as an ingot. The subfigures

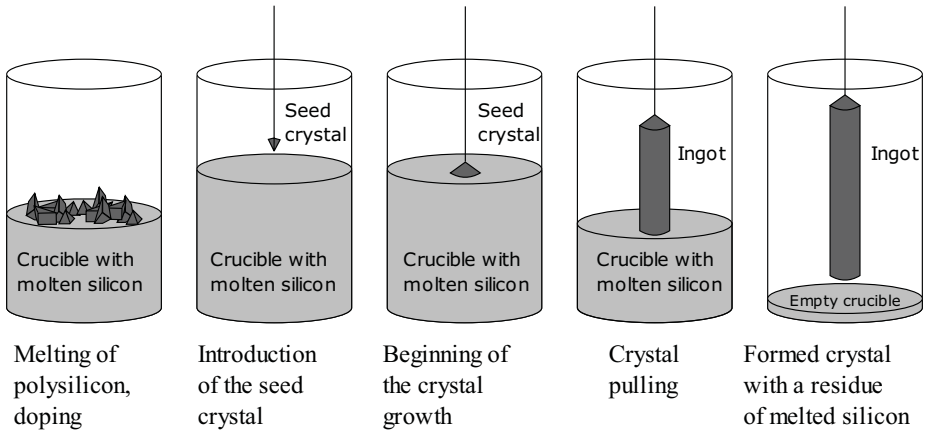


Figure 2.1: The main batch steps of the CZ process. Illustration from Wikipedia (the illustration is released to the public domain by the copyright holder).

illustrate the main batch steps of the CZ process:

Subfigure 1 (leftmost) High-purity multicrystalline silicon is melted in the crucible. Dopant impurity atoms, for example boron or phosphorus, may be added to the silicon for producing n-type or p-type silicon.

Subfigure 2 When the silicon is melted, the tip of the monocrystalline seed crystal is dipped into the melt.

Subfigure 3 When the tip of the seed crystal begins to melt, the seed crystal is slowly elevated. As the seed crystal is lifted, the molten silicon solidifies on the crystal. The seed crystal then grows radially and axially. During this growing stage, the crystal structure of the seed crystal is extended onto the solidifying silicon.

Subfigure 4 The produced crystal is referred to as an ingot. The ingot diameter is controlled by manipulating the ingot pulling speed and the temperature of the silicon melt. Stable growing conditions are essential for producing high crystal quality. According to Lan (2004) ingots of diameter up to 16 inches (approximately 400 millimeters) have been grown.

Subfigure 5 When the ingot has reached its desired length, or the crucible is about to become empty, the crystal growth is terminated by slowly decreasing the ingot diameter to zero.

The CZ process is operated in an inert atmosphere, typical an argon atmosphere. During the batch process, the seed crystal / ingot is rotated in one direction and the crucible is rotated in the opposite direction. The produced ingot will later be cut radially into thin discs, which are used for solar cells and in computers and electronics. Lan (2004) gives a survey on crystal growth, including the CZ process, and provides an extensive number of references for further reading.

2.2 Control Objectives

The quality parameters of the silicon ingot include complex physical and chemical properties, such as dislocation levels (violation of the desired crystal structure), impurity content, and dopant distribution. These quality parameters describe the ingot in microscopic perspective. Unfortunately, with today's sensor technology, it is not possible to measure these microscopic quality parameters online. The candidate has neither found any publications where online *estimation* of these parameters has been tested on real-life CZ processes. In addition to the microscopic quality parameters, the ingot diameter is most important. If the diameter is too large, there will be unnecessary cutting waste. If the diameter is too small, the customers' specifications are violated. The ingot diameter can be measured online and is commonly used as a control objective.

The main reason for fluctuations in crystal quality parameters is variations in growing conditions, such as variable ingot pulling speed and fluctuations in thermal conditions, during the crystal growth. Therefore, control strategies for the CZ process usually focus on maintaining stable growing conditions, in addition to controlling the ingot diameter. Controlling the ingot diameter also contributes to reducing fluctuations in the growing conditions, and thereby improving the crystal quality. What are the optimal growing conditions vary through the progress of the CZ batch (Irizarry-Rivera and Seider, 1997a; Lee et al., 2005).

2.3 Control Strategies

When developing control strategies, a most important choice is the sensors and controlling elements (actuators), and their locations in the process. Without receiving reliable information of the process states, or without the ability to manipulate these states, even the most clever control algorithm will fall short. The literature shows some variations with respect to the choice of sensors and controlling elements in the CZ process. Lee et al. (2005) presents a basic control strategy for the CZ process. As pointed out in Section 1.2, Lee et al. (2005) is the only publication found during the literature search which documents that a suggested

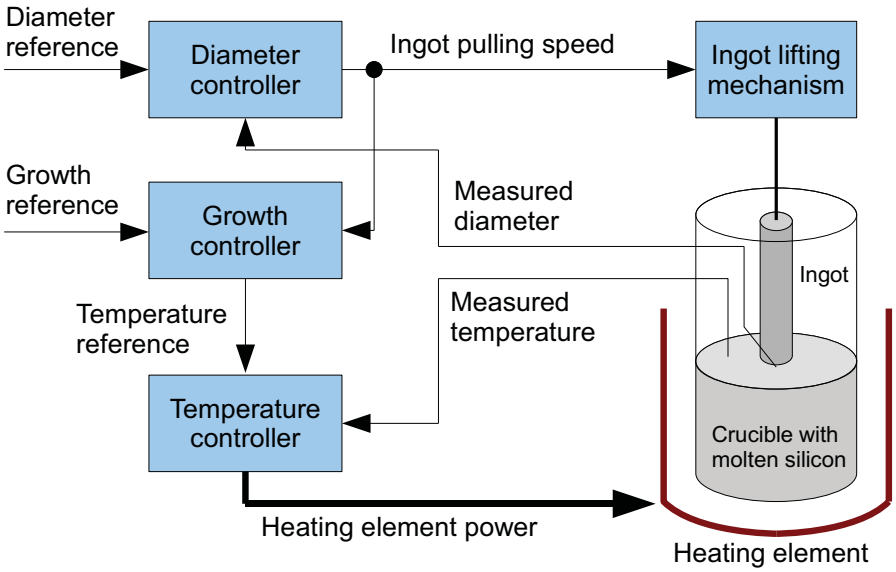


Figure 2.2: A basic control strategy for the CZ process. The illustration is inspired by Lee et al. (2005, Fig. 1). The shape of the heating element in the figure is not meant to reflect the shape of the real heating element.

control strategy works on real-life CZ processes. This control strategy is illustrated in Figure 2.2. The sensors are:

1. The ingot diameter [mm]. The diameter is measured at the melt / ingot interface, because the diameter can not be changed once the silicon is solidified.

2. The temperature [°C] of the molten silicon.

The controlling elements shown in Figure 2.2 are:

1. The ingot pulling speed [mm/h].

2. The power [kW] to the heating element, which heats the crucible.

Please note that the ingot pulling speed, which is the output of the diameter controller, is the process value of the growth controller. Hence, there is no sensor for the ingot pulling speed.

The sensors and controlling elements listed above are common for most control strategies considered during the literature search. Some control strategies found in literature also include additional sensors and / or controlling elements. However, these control strategies are not documented to work on real-life CZ processes and are therefore not further explored during this PhD study.

The control strategy shown in Figure 2.2 utilizes both single-loop control and cascade control. The ingot diameter is controlled by the diameter controller (DC), which is a single-loop controller. This controller has the measured ingot diameter as sensor and uses the ingot pulling speed as controlling element. Increased pulling speed gives decreased diameter.

The purpose of the growth controller (GC) is to grow the ingot at a desired ingot pulling speed [mm/h] defined by the growth reference, while allowing the DC to use the ingot pulling speed as its controlling element. This is achieved as the GC indirectly manipulates the DC: The GC is the master controller of a cascade control loop. The cascade slave controller is the temperature controller (TC), which controls the temperature of the melt by manipulating the heating element power. When the melt temperature increases, the ingot diameter decreases. Hence, the DC will then detect a diameter control error and manipulate the ingot pulling speed to compensate this error. The control strategy in Lee et al. (2005) also includes feedforward trajectories for all three controllers. These feedforward trajectories are not shown in Figure 2.2. Also Gevelber et al. (2001) considers the importance of such feedforward trajectories. According to Irizarry-Rivera and Seider (1997a) and Lee et al. (2005), PID controllers are the most used controllers in the CZ industry.

From the candidate's point of view, the control strategy of Lee et al. (2005) described above seems to be sub-optimal: The GC indirectly controls the ingot pulling speed as explained above. The path from the GC output to the ingot pulling speed goes through the physical ingot diameter. Hence, a temporary control error of the ingot diameter is necessary for the GC to manipulate the ingot pulling speed, because the ingot pulling speed is only manipulated by the output of the DC. This problem can probably be reduced, ideally prevented, by using a decoupler from the GC output to the ingot pulling speed. It is desired that the GC output has influence only on the ingot pulling speed, not on the ingot diameter. As both the temperature reference and the ingot pulling speed influence the ingot diameter, it should be possible to manipulate the ingot pulling speed to cancel the ingot diameter response caused by changes in the temperature reference. Such a decoupler must be a dynamic function based on process models. Chapter 5 suggests how to use a model predictive controller (MPC) to control the CZ process based on the sensors and controlling elements included in Figure 2.2.

2.4 The Czochralski Crystallization Process at SINTEF

The candidate was fortunate to have access to a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. A picture of this process is shown in Figure 2.3. The black cylinder ("barrel"), label (1), contains the crucible and the heating element. To simplify the explanation, this cylinder will be referred to as the barrel. The large, blue device, label (2), is the ingot lifting mechanism. A flexible metal pipe, label (3), is hanging from the lifting mechanism. This pipe has an "accordion structure". When the picture was taken, the lifting mechanism was near its upper position. The locking mechanism holding the seed crystal is thus hidden inside the flexible pipe. Label (4) is a flange, which the flexible pipe is attached to.

When a CZ batch is prepared, the barrel is lifted away, and the crucible containing the multicrystalline silicon is placed on a rotating device. The barrel is thereafter lifted in place. The seed crystal is fastened in its locking mechanism. Next, the lower end of the flexible pipe is attached to the flange at the top of the barrel. Finally, the heating element power is turned on and gradually increased. After a few hours the silicon is molten. The ingot lifting mechanism is then run to its lower position, dipping the tip of the seed crystal into the melt. Thereafter, the batch progresses as illustrated in Figure 2.1.

Figure 2.4 illustrates the inside of the barrel (label (1) in Figure 2.3) seen from above. The grey area in center, label (A), is the molten silicon contained in the crucible, label (B). The crucible is placed on a rotating device, label (C), shaped as a cylinder with bottom, and without top. The red color, label (D), is the heating element. The next circle, label (E), is a graphite ring. The outer circle, label (F), represents insulation and the outer wall.

The SINTEF CZ process can produce ingots of diameter up to 4 inches (approximately 100 mm). The maximum silicon charge in the crucible is 15 kg.

The CZ process at SINTEF is operated through a control system, developed from scratch by the engineers at SINTEF. The control system is based on a personal computer (PC) running National Instrument LabVIEW. The applied control strategy is identical to the one presented in Figure 2.2, except that the growth controller and the growth reference are not present. Instead, the temperature reference is set by the human process operator.

The heating element power is manipulated by a triode for alternating current (TRIAC). The output of the temperature controller is the input signal to the TRIAC. The heating element power is currently measured, but not included in the control strategy. There is significant dynamics from the TRIAC input signal to the measured heating element power. This dynamics is modeled in Paper D

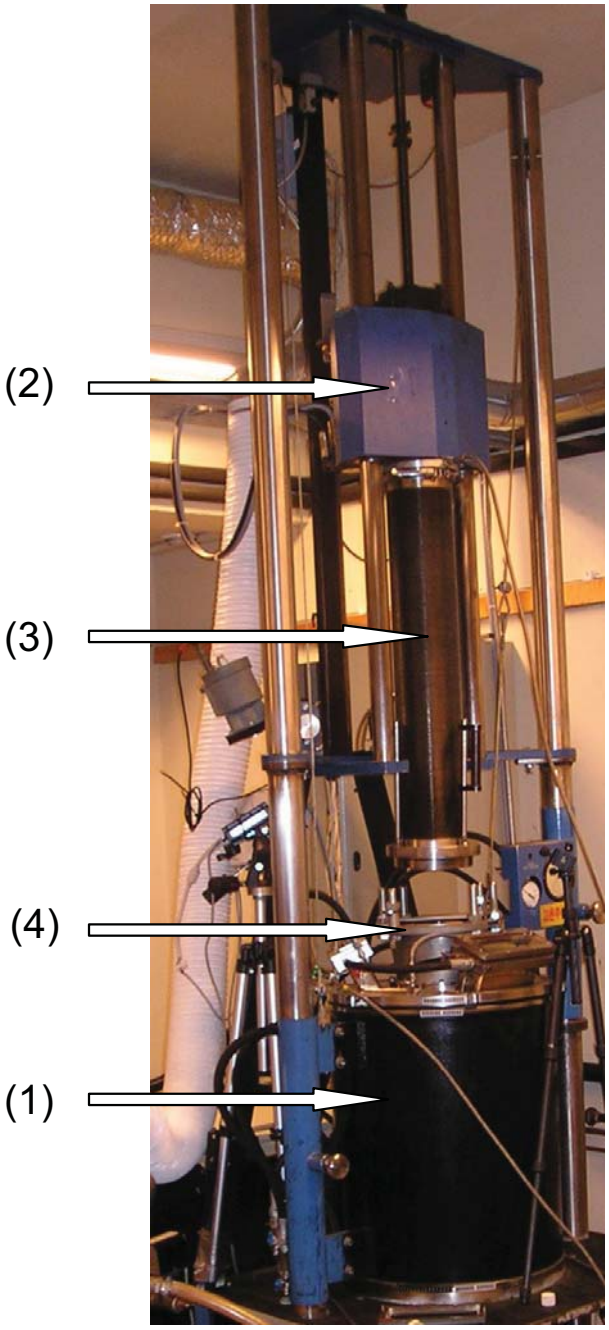


Figure 2.3: A picture of the CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. Label (1) is the cylinder ("barrel") containing the crucible. Label (2) is the ingot lifting mechanism. Label (3) is a flexible pipe. Label (4) is a flange, which the flexible pipe is attached to.

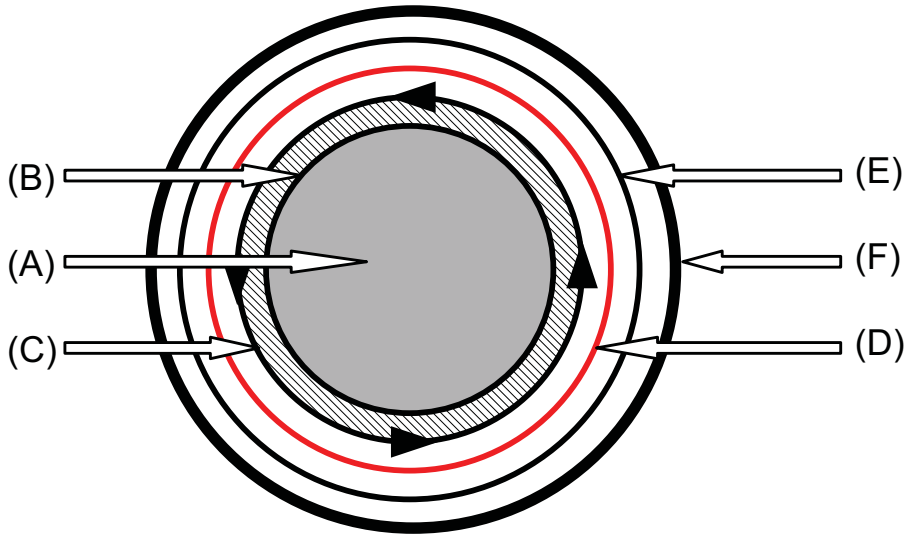


Figure 2.4: The barrel (label (1) in Figure 2.3) seen from above. Label (A) is the molten silicon. Label (B) is the crucible. Label (C) is a rotating device on which the crucible is placed. Label (D) is the heating element. Label (E) is a graphite ring. Label (F) is the insulation and the outer wall.

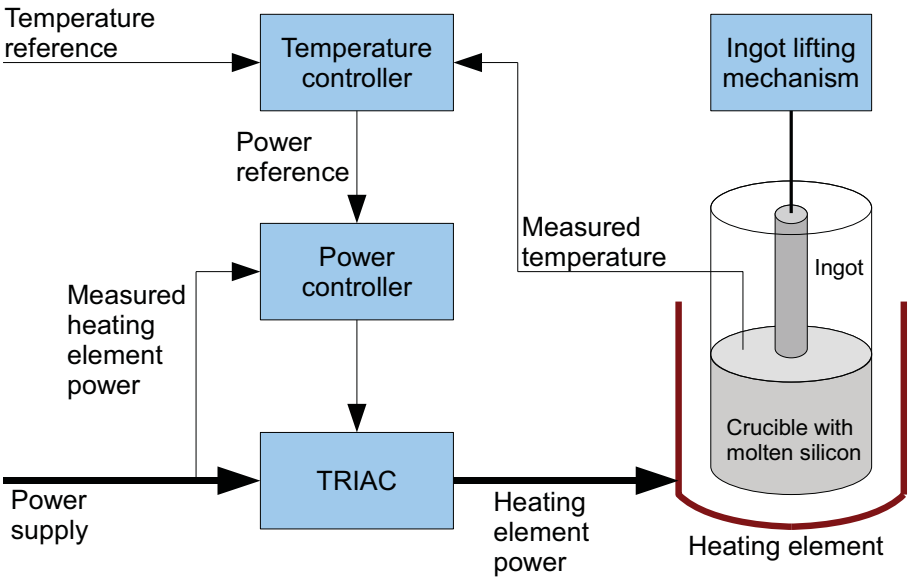


Figure 2.5: The suggested cascade control strategy to be used for quicker and more effective compensation of the power disturbances.

and Paper E of this PhD thesis. During experiments at the CZ process, process disturbances to the heating element power were discovered. That is, there are responses in the measured heating element power that can not be explained by the TRIAC input signal. These responses are known to be process disturbances, not measurement errors, because there are corresponding responses in the temperature. The candidate then suggested to use cascade control for quicker and more effectively compensate these process disturbances. The cascade inner loop (slave control loop) is a power control loop, which controls the heating element power using the TRIAC as controlling element. The cascade outer loop (master control loop) is a temperature control loop, which sets the power reference (setpoint) to the power controller. The power controller then ensures that the power requested by the temperature controller is actually applied to the heating element, regardless of process disturbances. This cascade control strategy is illustrated in Figure 2.5. Paper F of this PhD thesis presents this cascade control strategy, including tests at the SINTEF CZ process.

It is desirable to measure and control the temperature of the molten silicon in the crucible. However, the temperature sensor at the SINTEF CZ process is

a pyrometer which measures the temperature at the outer surface of the graphite ring (label (E) in Figure 2.4). This pyrometer will be referred to as the graphite pyrometer. Using the graphite pyrometer for temperature control, the temperature of the graphite ring is actually controlled, not the temperature of the molten silicon. Hence, this sensor location is based on the assumption that stable temperature of the graphite ring implies stable silicon temperature.

During a CZ batch, engineers at SINTEF and the candidate tested a new pyrometer which measures the temperature directly in the molten silicon. This pyrometer will be referred to as the silicon pyrometer. The signal from the silicon pyrometer seems reasonable based on the melting point of silicon and based on measured temperature responses to steps in the TRIAC input signal. Unfortunately, the output of this pyrometer has much high-frequency noise. An intuitive and feasible solution to this issue is to use a traditional lowpass filter. However, this approach will give a significant phase lag over the filter, which is unfortunate for the temperature control.

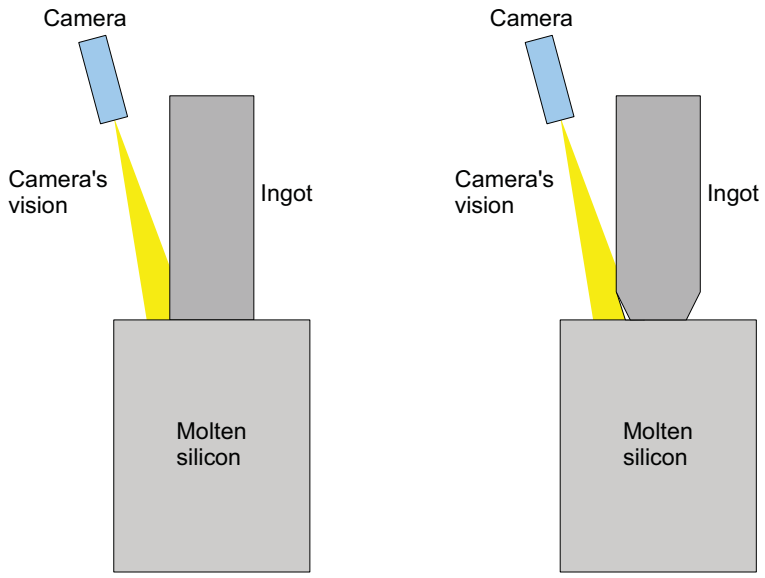
Comparison of the graphite pyrometer and the silicon pyrometer shows that there is quite high correlation between the signals of the two pyrometers. The silicon pyrometer has the disadvantage of much measurement noise, but it has the advantage of measuring the temperature directly in the molten silicon. On the other hand, the graphite pyrometer has the disadvantage of not measuring the silicon temperature directly, but it has the advantages of little measurement noise and quite high correlation with the silicon pyrometer. Based on these advantages and disadvantages, the candidate has developed a sensor fusion algorithm that fuses the measurement signals of the two pyrometers. The algorithm provides an estimate of the temperature of the molten silicon. For a given lowpass filter cut-off frequency, this estimate gives the same amount of measurement noise as a traditional lowpass filter, but with significant less phase lag. This algorithm is published in Paper G of this PhD thesis.

At the SINTEF CZ process, the ingot diameter is measured by a camera. The camera is located outside the CZ process and observes the ingot / melt interface through a small window in the barrel (label (1) of Figure 2.3). Because of the intense heat radiation from the process, a shield is covering the camera, protecting it from the heat radiation, while letting through enough light for the camera to observe the ingot / melt interface. The camera is constantly taking pictures of the ingot / melt interface. These pictures are sent to LabVIEW running on the control system computer. LabVIEW then uses image processing to estimate the ingot diameter. The camera is located above and somewhat to the side of the ingot as illustrated in Figure 2.6(a). The main disadvantage of this sensor technology is that the diameter measurement fails if the diameter decreases rapidly. The ingot / melt interface will then be hidden behind the ingot as illustrated in Figure 2.6(b).

(Figures 2.6(a) and 2.6(b) are somewhat simplified. An accurate illustration of the measurement issue would require three-dimensional drawings.)

During experiments at the SINTEF CZ process, the candidate faced this diameter measurement problem when making steps at the heating element power and at the ingot pulling speed for the purpose of observing the corresponding response of the ingot diameter. One can argue that if there are perfect reference tracking and disturbance rejection in the CZ process, the diameter will not decrease during the crystal growth (except for the last stage, where the diameter is decreased intentionally). However, if the diameter actually decreases, the diameter measurement signal may be erroneous. It is then difficult to predict how the diameter controller will respond to the erroneous signal. As a worst case scenario the erroneous signal may indicate too large diameter. The diameter controller will then decrease the diameter further. The candidate has not done any effort to quantify how small diameter reduction that may trigger this measurement error. It is desirable to have a diameter measurement also during the last stage of the crystal growth, where the diameter is intentionally decreased to zero.

The power to the CZ heating element is taken from the 3×400 VAC (three phase, 400 Volt, alternating current) power grid. The power is manipulated using a TRIAC at each phase (for simplicity the three TRIACs are presented as one TRIAC in this PhD thesis, because the TRIACs share the same input signal). After the TRIACs, the power is transformed to DC (direct current), which is connected to the heating element. The sensor for the heating element power measures only the electric current of one phase before the TRIACs. Hence, the power measurement can not properly detect voltage variations at the power grid nor asymmetry between the phases. Neither varying power loss in the rectifier (the converter from alternating current to direct current) can be detected. These issues are of course significant disadvantages, making the power measurement less reliable.



(a) Camera vision is OK.

(b) The ingot / melt interface is outside the camera's vision.

Figure 2.6: The figure to the left illustrates correct diameter measurement. The figure to the right illustrates how rapidly decreasing diameter causes erroneous diameter measurement, because the camera can not observe the ingot / melt interface.

Chapter 3

The Copper Refining Process

Paper A and Paper B of this PhD thesis were presented at the conference SIMS 2008 and are included in the conference's proceedings. The papers were drafted during the candidate's master thesis (Komperød (2008)). Both papers include examples based on logged process data from the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway. The present chapter gives a brief presentation of this process. A more thorough explanation of the process is not relevant for this PhD thesis. The candidate's only source of information for this plant is Hauge (2007), which is a presentation of the plant that is not publicly available, and personal communication with the author of this reference. The candidate has not done any literature search on copper refining processes in general, because this is not relevant for the PhD thesis. The illustrations and most of the text in this chapter are from the candidate's master thesis (Komperød (2008)).

Xstrata Nikkelverk has an annual capacity of 86,000 tonnes of nickel, 40,000 tonnes of copper, 5,200 tonnes of cobalt, and 115,000 tonnes of sulphuric acid (Hauge, 2007).

Figure 3.1 illustrates the copper refining process. Only the most relevant chemical components are shown in the figure. From the roasting furnace, copper oxide, CuO, enters the copper leaching process. Sulphuric acid, H_2SO_4, is added for leaching the copper. The following chemical reaction is the most significant with respect to the copper refining process (Hauge, 2007)

$$CuO + H_2SO_4 \rightarrow CuSO_4 + H_2O. \tag{3.1}$$

Hence, the copper is present as the salt copper sulfate, $CuSO_4$, dissolved in water. This solution is pumped to the filter presses. In the filter presses nondissolved metal oxides, including CuO, are removed from the solution. Leaving the filter presses, the solution enters the scrap columns. These columns are filled with copper metal for cementation of metals like silver (Ag) and bismuth (Bi) and the

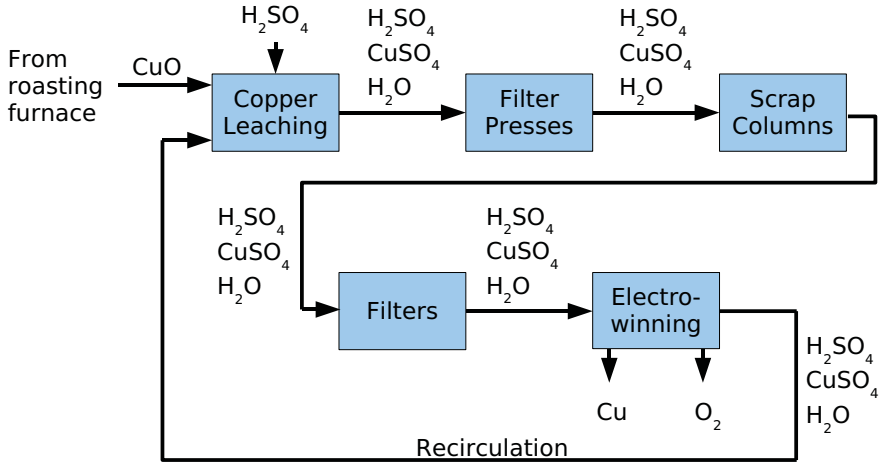


Figure 3.1: Overview of the copper refining process at Xstrata Nikkelverk. Only the most relevant chemical components are shown.

metalloid tellurium (Te). During cementation these metals and metalloids are reduced through a reaction with the copper. Here the reaction between bismuth and copper is shown (Hauge, 2007)

$$\mathrm{Bi}^{4+} + 2\mathrm{Cu} \rightarrow \mathrm{Bi} + 2\mathrm{Cu}^{2+}. \tag{3.2}$$

In the filters downstream of the scrap columns, these newly reduced metals and metalloids are removed. After this last filtering, the solution enters the electrowinning. In the electrowinning the copper is reduced to metallic form through the following reactions (Hauge, 2007)

$$\mathrm{Cu}^{2+} + 2\mathrm{e}^{-} \rightarrow \mathrm{Cu}, \tag{3.3}$$

$$\mathrm{SO_4}^{2-} + \mathrm{H_2O} \rightarrow \mathrm{H_2SO_4} + \frac{1}{2}\mathrm{O_2} + 2\mathrm{e}^{-}. \tag{3.4}$$

The copper reduction, (3.3), takes place on a copper cathode where the reduced copper metal accumulates. Not all the copper is reduced in the electrowinning process. The remaining copper sulfate and the sulphuric acid are recirculated by leading them back to the leaching process (Hauge, 2007).

Figure 3.2 shows a very simplified process and instrument diagram (P&ID) of the copper refining process. The diagram shows only the measurements that are relevant for Paper A and Paper B of this PhD thesis. These papers consider three

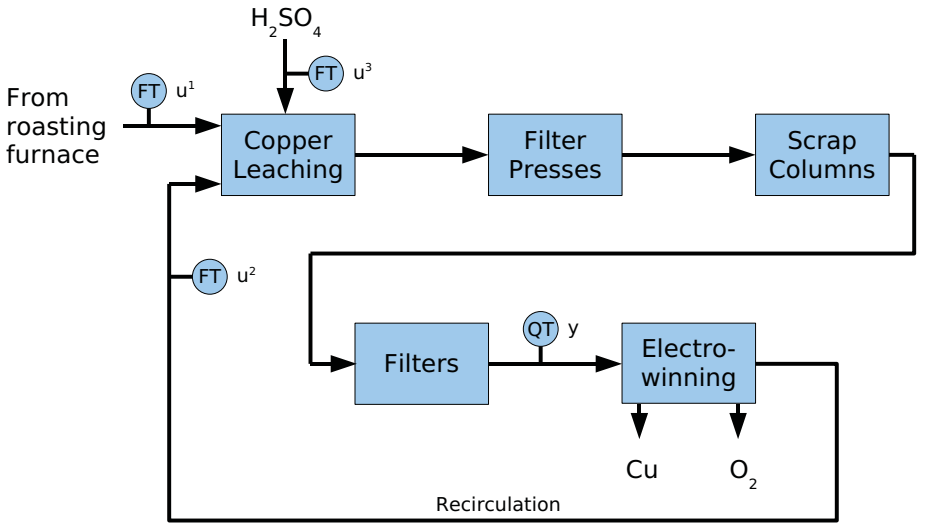


Figure 3.2: Simplified process and instrument diagram (P&ID) of the copper refining process at Xstrata Nikkelverk. FT means flow transmitter (flow sensor), and QT means quality transmitter (quality sensor). The quality transmitter measures the concentration of sulphuric acid.

process inputs, u^1, u^2 and u^3, and one process output, y. Input u^1 is the mass flow [tonne/hour] from the roasting furnace to the copper leaching process. Input u^2 is the volumetric recirculation flow [m^3/hour] from the electrowinning to the copper leaching process. Input u^3 is the volumetric flow [litre/hour] of sulphuric acid to the copper leaching process. Output y is the concentration of sulphuric acid [gram/litre] in the solution before the electrowinning.

In Paper A, u^1, u^2, u^3, and y are considered. In Paper B, only u^1 and y are considered (u^1 is referred to as u in Paper B). In Paper A the logged process data are used as real-life examples of outlier detection and time delay estimation. In Paper B two system identification algorithms are used to identify models of the dynamics from u^1 to y. The purpose of this modeling work is to validate that the algorithms give identical models. System identification is the science of developing dynamic models empirically.

A mechanistic (first principle) model of this copper refining plant is presented in Lie and Hauge (2008). In Alic et al. (2009) the plant is modeled using Modelica.

Chapter 4

System Identification

This PhD thesis includes seven publications, Papers A through G, which all consider topics within systems and control engineering. Each of the seven publications includes references that provide the necessary background information. In general, topics within systems and control engineering are extensively covered in easily accessible books. Section 1.2 presents some good books that have been used during this PhD study. From the candidate's point of view, there is no need to explain common topics of systems and control engineering in this PhD thesis. However, one exception will be made: System identification is to some extent relevant for all seven publications. System identification is the science of developing dynamic models empirically. A brief introduction to this topic is given in this chapter.

4.1 System Identification Methods

For many fields of science and engineering, including systems and control engineering, it is essential to understand and model the dynamic behavior of physical and chemical processes. There are basically two different approaches for such modeling. One approach is to describe the process in terms of laws from physics and chemistry, for example Newton's laws and the laws of thermodynamics. This modeling approach is known as mechanistic modeling or first principle modeling. The other approach is empirical modeling, where the process' dynamic behavior is revealed through observations of the process, either during normal operation or through experiments. Mathematical models are then developed from these observations.

Maybe the most used method for empirical modeling is the ordinary least squares (OLS) method, which estimates parameters for describing the output variables as mathematical functions of the regressors (input variables). This method was developed by Gauss as early as 1794. Several closely related methods exist, such as principal component regression (PCR) and partial least squares regression

(PLS-R). OLS, PCR, and PLS-R are based on the assumption that there are linear relationships between the regressors and the outputs, or that these relationships can be rewritten to linear forms. There are also methods for empirical modeling of systems with nonlinear relationships between the regressors and the outputs, for example the nonlinear least squares method and artificial neural networks (ANN).

System identification is the science of developing *dynamic* models empirically. Hence, system identification is a special case of empirical modeling. System identification can be divided into two different approaches: (i) Black-box modeling and (ii) grey-box modeling. Using black-box modeling, the human model builder needs in principle no knowledge of the process at all. That is, the process is a "black box" in the sense that only the process' inputs and outputs are observed, while the mechanisms that make the process' outputs respond to the process' inputs are hidden to the model builder. Hence, the black-box model is developed based only on the observations of the process inputs and the corresponding process outputs.

Using grey-box modeling, the human model builder combines process knowledge with empirical modeling. Typically, a model structure is developed based on mechanistic (first principle) modeling, where one or more parameters are unknown. These unknown parameters are then estimated using empirical modeling. A closely related approach is to use process knowledge to preprocess the process inputs and / or outputs before applying a black-box method. Assume for example that the process inputs are measured voltage and measured electric current to a heating element, and that the process output is the temperature of water heated by this heating element. It is then reasonable to multiply the voltage and the current to obtain the electric heating element power. A black-box model which takes the electric power as input and the water temperature as output is then to be identified.

In general, both for black-box modeling and for grey-box modeling, the human model builder must decide a model structure, which parameters are to be identified by the chosen system identification method. Some system identification methods can estimate the model order. Ljung (1999) divides system identification methods into three basic approaches:

The Prediction Error Method The prediction error method (PEM) identifies the model parameters as the ones that minimize the prediction errors. A scalar optimization criterion, $V(\theta)$, defines exactly what is meant by "minimizing the prediction errors". Here, θ is the parameter vector, which contains all the parameters to be identified. Hence, PEM forms an optimization problem, which aims at minimizing $V(\theta)$ with respect to the elements of the vector θ. The most commonly used optimization criterion is probably

$$V(\theta) = \frac{1}{2} \sum_{k=1}^{N} \varepsilon_k^T(\theta) \Lambda^{-1} \varepsilon_k(\theta). \tag{4.1}$$

Here, k is the sample number in the dataset. $\varepsilon_k(\theta)$ is the prediction error, which is a column vector of the same dimension as the process' output vector. For example, if the process has three outputs, which are stacked in an output column vector of three elements, $\varepsilon_k(\theta)$ is also a column vector of three elements, each element representing the prediction error of an output. Λ^{-1} is a weighting matrix. For the common case of single output systems, the optimization criterion simplifies to

$$V(\theta) = \frac{1}{2} \sum_{k=1}^{N} \varepsilon_k(\theta)^2. \tag{4.2}$$

The prediction error, $\varepsilon_k(\theta)$, is often understood as the one-step-ahead prediction error. That is, $V(\theta)$ is optimized to give the best prediction of the process outputs at timestep k based on all information available at timestep $k-1$. However, in some cases one chooses to minimize the ballistic simulation error instead of the one-step-ahead prediction error. That is, $\varepsilon_k(\theta)$ is then the ballistic simulation error.

Ljung (1999) considers PEM to be the basic system identification method. Among the main advantages of PEM is that the method can be applied to any model structures, including tailor-made grey-box model structures. The PEM method is extensively covered in Ljung (1999). The PEM method is used in Papers D, E, and G of this PhD thesis.

The Correlation Method The correlation method identifies the model parameters as the ones that give zero correlation between the prediction errors, $\varepsilon_k(\theta)$, and previous process inputs and outputs. The basic idea is that if this correlation is nonzero, then parts of the prediction error can be explained by previous inputs and outputs. If all relevant information in previous inputs and outputs was properly picked up in the model, no such information would have been left in the prediction errors. By relevant information, it is here meant relevant for predicting the process outputs.

The correlation method is not considered in any of the seven publications included in this PhD thesis. This method will therefore not be given further attention here. The interested reader is referred to Ljung (1999, Sections 7.5 and 7.6) for an introduction to this topic.

The Subspace Method The subspace methods are a family of closely related system identification algorithms. These methods are based on the ordinary least squares method and avoid iterative optimization algorithms that are in the risk of being trapped in local minima. Subspace methods are thoroughly explained in Van Overschee and De Moor (1995) and Katayama (2005).

Paper B of this PhD thesis considers the relationship between the subspace algorithm DSR E and a two-step ARX algorithm. Paper C considers implementation of this two-step ARX algorithm. Paper A is also somewhat related to subspace methods.

4.2 System Identification Versus Mechanistic Modeling

When a process is to be modeled, it must be decided whether to use mechanistic (first principle) modeling, black-box modeling, or grey-box modeling. This section discusses some of the main advantages and disadvantages of these three modeling approaches. Hauge (2003, Section 3.3) also discusses mechanistic versus empirical modeling.

Empirical modeling, including system identification, depends on logged data of the process' inputs and outputs. In some cases such data are not available, for example because the plant is not built yet. For a black-box or grey-box model to be identified successfully, the logged process data must be informative in the sense that they contain information of the complete process dynamics of interest, not only a subset of the process dynamics, for example only a narrow frequency range. Some system identification methods, including some subspace methods, give biased models if the process data are logged in closed loop. The term "biased models" refers to models that have *systematic* parameter errors. On the other hand, mechanistic models can be build even if no logged process data are available, provided that all model parameters are known.

System identification identifies parameters in a given model structure. The choice of model structure must be done by the human model builder. Using black-box modeling, it is very common to apply linear model structures. There also exist several flexible, nonlinear black-box model structures, for example the Hammerstein-Wiener (HW) model structure, the non-linear ARX and ARMAX model structures, and model structures based on artificial neural networks (ANN). However, a common problem using such flexible, nonlinear black-box model structures is that the structures have many parameters to be identified. The dataset used for identification must then be very informative to be able to identify all parameters uniquely. Such model structures are usually identified using PEM,

which is in considerable risk of being trapped in local minima when many parameters are to be identified. Paper D of this PhD thesis considers nonlinear system identification using the HW model structure. In this article, a general, flexible HW model structure is used to model a part of the SINTEF CZ process. During the identification, PEM is trapped in a local minimum and gives a very poor model. This problem is avoided by developing a tailor-made HW model structure with few parameters based on human inspection of the dataset. As the number of parameters is significantly reduced, the PEM identification succeeds.

A main advantage of using grey-box modeling over black-box modeling is that the model structure is developed based on process knowledge. In addition to explain the process' true physics, a grey-box model structure should not have too many parameters to be identified.

Using black-box modeling, the internal states of the models do not have physical meanings. Hence, black-box models can not be used to estimate unmeasurable physical states. On the other hand, mechanistic models may be used for this purpose, which is very useful in many applications.

Developing mechanistic models may be very challenging and time-consuming tasks, even for clever scientists and engineers. In such modeling work, it is often necessary to make assumptions. In many cases it may be difficult to verify that these assumptions actually are valid. Also, mechanistic models may depend on parameters that are unknown.

Even if a process has very complex physics, the dynamic behavior from the process inputs to the process outputs *may* be rather simple. Black-box modeling considers only the behavior of the process' input-output dynamics, not the physics causing this dynamics. Hence, black-box modeling may give a simple and quite accurate input-output model, even if the process' physics is very complex. This black-box advantage is demonstrated on the CZ process in Lee et al. (2005). In this article, the dynamics from the ingot pulling speed to the ingot diameter, and from the heater temperature to the ingot diameter, are modeled using *very* simple, linear, empirical models. The process' time-varying character is handled by identifying different models for five different operating points. The models are then alternated as the CZ batch progresses. On the other hand, modeling these dynamics using mechanistic modeling may be a very challenging task.

When developing models, it is important to keep in mind the models' intended usages. Many processes are very complex, and it may be necessary to use a very complex model to give an accurate description of the process' dynamic behavior. However, if the model is to be used for online applications, such as model predictive control (MPC) or state estimation, it is most important to be aware that a limited amount of computations can be done for each update step. If the model is too complex, the computations may be so demanding that the computer will not be

able to keep up with the real process. Hence, for many online applications, simple models have to be used, even if complex models of better accuracy are available. Black-box modeling has the advantage that the model complexity is specified by the human model builder through the chosen model structure.

When modeling a process using system identification, a noise model may be identified in addition to the input-output model. A noise model takes the one-step-ahead prediction errors as inputs and corrects the model states in order to improve the one-step-ahead prediction for the next timestep. Noise models are extensively used in the publications included in this PhD thesis. A further explanation of this very interesting topic will not be given here, as it is assumed to be well known in the systems and control engineering community. The topic is well explained in Ljung (1999, Chapter 3).

During this PhD study, the candidate has chosen to work with empirical modeling rather than mechanistic modeling. There are three reasons for this choice: (i) Based on the candidate's literature search, there seems to be very little published research on empirical modeling of the CZ process. (ii) The published work on mechanistic modeling of the CZ process often results in rather complex models. It may be difficult to use these complex models for online applications, such as model predictive control (MPC) and state estimation. (iii) Among the publications considered during the literature search, only Lee et al. (2005) presents a control strategy that is documented to work on real-life CZ processes. This article uses simple, empirical models.

Chapter 5

Further Work

Papers D, E, F, and G of this PhD thesis contain sections for further work / further research. These suggestions will not be repeated here. This chapter will focus on further work at the Czochralski (CZ) crystallization process at SINTEF Materials and Chemistry in Trondheim, Norway. As pointed out in Section 1.3, there are some measurement issues at this CZ plant, which forced the candidate to focus on other parts of the CZ process than he would have done otherwise.

This chapter focuses on measuring, modeling, and control of the heating element power, the temperature of the molten silicon, and the ingot diameter. The candidate recommends to start the work with the heating element power, because the other two variables strongly depend on the heating element power. In Section 2.4 it is explained that the present sensor for heating element power can not properly handle voltage variations at the power grid nor asymmetry between the phases. Because the heating element power is essential for temperature control and diameter control, the candidate strongly recommends to install a power sensor which properly compensates for voltage variations and asymmetry. Modeling and control of the heating element power are discussed in Papers D, E, and F of this PhD thesis and will not be further discussed here.

The next issue is to measure the temperature of the molten silicon. Paper G suggests a sensor fusion algorithm, which fuses the signals from two pyrometers in order to estimate the silicon temperature. As pointed out in Paper G, more work is required to model the nonlinear and time-varying character of the process for the sensor fusion algorithm to work properly over a wide temperature range.

The ingot diameter measurement seems to work well except at rapidly decreasing diameter. This issue is explained in Section 2.4. The candidate has not worked with sensor technologies for measuring the ingot diameter. He has therefore insufficient knowledge to further discuss this issue. Hurle (1993) provides a discussion on ingot diameter measurement.

When the sensors for heating element power, silicon melt temperature, and

ingot diameter operate properly, the candidate recommends to do some positive and negative steps in the heating element power at different temperatures, at different ingot diameters, and at different levels of melt in the crucible. The measured temperature and diameter responses will probably reveal much useful information about the dynamics from the heating element power to the silicon melt temperature and to the ingot diameter, including possible nonlinear and time-varying process characteristics, length of the dominant time constants, and time delays. Similarly, it is recommended to do steps in the ingot pulling speed to observe the ingot diameter response. There may also be some dynamics from the ingot pulling speed to the melt temperature, as changed silicon solidification rate [kg/h] will change the heat power [kW] released due to heat of solidification. It is essential that the temperature controller and the diameter controller are in open loops (manual mode) during these experiments. Otherwise, the closed loops dynamics will be observed, not the process' dynamics.

The further modeling work will highly depend on the conclusions of the step-response tests. From the candidate's point of view, it is important to keep the models simple as they are to be used for control purposes. As logged process data are available, it seems reasonable to include these data in the modeling work by applying a black-box or grey-box approach. For black-box or grey-box modeling, it may be necessary to run experiments with a wider range of excitations than only steps.

When process models are available, the next issue is process control. Control of the heating element power is discussed in Paper F. It seems reasonable to use the control strategy presented in Lee et al. (2005, Section 2) as a starting point for controlling the silicon melt temperature and the ingot diameter, because this control strategy is used in the commercial silicon industry and is not very complex. This control strategy is also briefly discussed in Section 2.3 of this PhD thesis.

From the candidate's point of view, a significant weakness of the control strategy of Lee et al. (2005, Section 2) is that it seems not to take the process' multivariable character properly into account. The control strategy has two references: (i) the ingot diameter and (ii) the ingot pulling speed. There are also two controlling elements: (i) the heating element power and (ii) the ingot pulling speed. Hence, the ingot pulling speed is both a reference and a controlling element. (In Lee et al. (2005) and in Section 2.3 of this PhD thesis, the reference of the ingot pulling speed is referred to as the growth reference.) The main dependencies of the process are illustrated in Figure 5.1. The dynamics from the ingot pulling speed to the melt temperature is not included in the figure, because this is assumed to be weak.

In Lee et al. (2005) the ingot diameter is controlled using the ingot pulling speed as controlling element. It is assumed that manipulating the ingot pulling

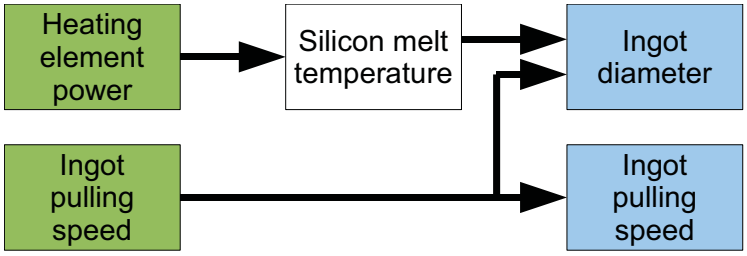


Figure 5.1: The main dependencies in the CZ process. The green color (to the left) represents controlling elements. The blue color (to the right) represents variables to be controlled.

speed gives faster response to the ingot diameter than manipulating the heating element power, and that this is the reason for using the ingot pulling speed as controlling element for the diameter controller. After manipulating the ingot pulling speed to compensate a diameter error, the ingot pulling speed will be off its reference. This can only be compensated by manipulating the heating element power, which will change the diameter. This introduces a new diameter error, which the diameter controller corrects by manipulating the ingot pulling speed. Hence, a well-calculated change of the heating element power will indirectly bring the ingot pulling speed back to its reference at the expense of a temporary diameter error. From the candidate's point of view, it seems sub-optimal to introduce a temporary diameter error to bring the ingot pulling speed back to its reference. Section 2.3 suggests solving this issue using a decoupler. Later in this chapter, a model predictive controller (MPC), which also handles this issue, is suggested.

It seems reasonable to assume that more variations in the heating element power give more variations in the temperature gradients of the silicon melt. Further, it seems reasonable to assume that varying temperature gradients conflict the desire of stable growing conditions, which are essential for producing high crystal quality. Hence, limiting power variations should be taken into account when developing the control strategy and tuning the controller parameters.

There are several possible ways to implement the improvements suggested above. From the candidate's point of view, a reasonable approach will be to develop a model predictive controller (MPC). The model to be used in the controller should have the heating element power and the ingot pulling speed as inputs. The outputs should be the ingot diameter and the silicon melt temperature. The candidate suggests to include the following quantities in the MPC optimization criterion:

1. The deviation between the ingot diameter and its reference (for limiting the

diameter error).

2. The deviation between the ingot pulling speed and its reference (for keeping the growth rate close to its reference).

3. The time-derivative of the heating element power (for limiting variations of temperature gradients in the silicon melt).

4. The time-derivative of the silicon melt temperature (for providing stable growing conditions).

Weighting the square of these quantities, the optimization criterion at timestep t, J_t, which is used to compute the controller outputs at timestep $t+1$, will then be on the form

$$J_t = \sum_{k=t+1}^{t+N} \left[w_k^1 (d_k - d_k^{\text{ref}})^2 + w_k^2 (v_k - v_k^{\text{ref}})^2 + w_k^3 \left(\frac{P_k - P_{k-1}}{\Delta t} \right)^2 + w_k^4 \left(\frac{T_k - T_{k-1}}{\Delta t} \right)^2 \right]. \tag{5.1}$$

Here, N is the prediction horizon. w^i are weights. d is the ingot diameter. v is the ingot pulling speed. P is the heating element power. T is the temperature of the silicon melt. Δt is the time between each update of the controller outputs. The superscript "ref" refers to the reference (setpoint).

In Figure 5.1 it may seem somewhat strange that the ingot pulling speed is both a controlling element and a variable to be controlled. Please note how smoothly this issue is solved using an MPC controller. In Figure 5.1 the dynamics from the ingot pulling speed to the silicon melt temperature is not drawn, because it is assumed to be weak. However, if it turns out that this dynamics is significant after all, it can easily be included in the MPC controller by including it in the process model.

When a control strategy is implemented and properly tuned at the SINTEF CZ process, one should consider to include feedforward trajectories in the control strategy. Lee et al. (2005) and Gevelber et al. (2001) consider the importance of such trajectories.

Chapter 6

Conclusions

This PhD study has focused on data preprocessing, empirical modeling, state estimation for the purpose of noise filtering, and process control. Applications to real-life processes are emphasized, in particular the Czochralski (CZ) crystallization process. The work on the CZ process is based on a CZ plant at SINTEF Materials and Chemistry in Trondheim, Norway. Logged process data from the copper refining process at Xstrata Nikkelverk in Kristiansand, Norway, is used in a minor part of the research work.

When using logged process data for empirical modeling, outliers in the dataset may cause poor model quality. Also when using logged process data for model validation, outliers may result in erroneous conclusions. Ljung (1999) shows that model residuals can be used for outlier detection. This PhD thesis presents an alternative approach to the method of Ljung (1999). This approach is based on identification of the innovation process directly from logged process data, i.e. without relying on process models. It can be mathematically proved that the approach presented in this PhD thesis turns out to be a special case of the method presented in Ljung (1999). The suggested approach was tested on a dataset from Xstrata Nikkelverk. Three outliers were detected in the dataset. These outliers were replaced by interpolation between their neighboring samples. After replacing the outliers, the innovation process was re-identified. Replacing the outliers reduces the mean squared innovation process by more than 80%. Hence, if this dataset was to be used for empirical modeling, replacing these outliers would strongly improve the model quality. The approach suggested in this PhD thesis was also used on a dataset from the SINTEF CZ process. At this dataset two outliers were detected.

System identification is the science of developing dynamic models empirically. Several commonly used system identification algorithms require the human model builder to specify the time delays between the process inputs and outputs. The System Identification Toolbox for MATLAB includes the command `delayest`, which estimates such time delays. This PhD thesis shows that the time delay

estimates produced by the command are sensitive to several factors which limit the command's practical usefulness. This conclusion is based on a simulation study, as well as on logged process data from Xstrata Nikkelverk. The PhD thesis suggests an improvement to `delayest`, which handles one sensitivity issue better than the original command under certain ideal assumptions. However, also this improved command has very limited usefulness when applied on datasets logged from real-life processes.

DSR E is a system identification algorithm developed to give consistent model estimates both for process data logged in open loop and for process data logged in closed loop. This PhD thesis shows mathematically that the DSR E algorithm can be approximated by a two-step ARX algorithm. This approximation is referred to as DARX. The DSR E and DARX algorithms are compared on logged process data from Xstrata Nikkelverk. As expected, the model developed by DARX is identical to the model developed by DSR E. DARX is too similar to DSR E to be considered a "new" system identification algorithm. The purpose of DARX is to show the similarity between DSR E and a two-step ARX algorithm. From the candidate's point of view, DARX may be easier to understand and easier to implement than DSR E. DARX has been developed only for single input / single output (SISO) systems. It is believed that DARX can be extended to multiple input / multiple output (MIMO) systems. However, such an extension has not been considered in the PhD thesis. The second ARX step of DARX slightly differs from the standard ARX form. This apparently prevents standard ARX software from being used, such as the MATLAB command `arx` of the System Identification Toolbox. This PhD thesis presents a rewriting of the second ARX step of DARX, such that standard ARX software can be used also for this ARX step.

At the SINTEF CZ plant, the crucible containing the molten silicon is heated by a heating element. The heating element power is manipulated using a triode for alternating current (TRIAC). There is a dynamic relationship between the TRIAC input signal (control system output) and the actual (measured) heating element power. The PhD thesis presents system identification of this dynamics. Initially, linear model structures were used. The results of this work revealed that the dynamics is somewhat nonlinear. The candidate therefore used the Hammerstein-Wiener model structure. As no independent dataset was available for model validation, significant effort was put into reducing the number of model parameters, in order to avoid overfitting. A good compromise between model fit and few parameters was achieved in a Hammerstein model. This model has only one more parameter than a linear output error (OE) model. Still, the Hammerstein model reduces the mean squared ballistic simulation error by a factor of six compared to the linear OE model. The Hammerstein model was extended by adding a noise model. This extended Hammerstein model reduced the mean squared one-

step-ahead prediction error with 42% compared to a linear ARMAX model. The extended Hammerstein model has one more parameter than the ARMAX model.

This PhD thesis also presents work on adaptive (recursive) system identification of the dynamics described in the previous paragraph. Three ARMAX models, each having four model parameters, are compared in terms of the mean squared one-step-ahead prediction error. The first model is non-adaptive. The second model is adaptive in all four model parameters. This model is referred to as the adaptive ARMAX model. The third model has adaptive gain, while the pole, the zero, and the noise model are fixed. This model is referred to as the adaptive gain model. The choice of making only the gain adaptive is justified from the results of the system identification work described in the previous paragraph. The adaptive ARMAX model performs better than the non-adaptive model for some choices of the forgetting factor. However, the adaptive ARMAX model's performance is very sensitive to the choice of the forgetting factor. An even more serious issue of this model is that the parameter adaptation switches the model between being stable and unstable. This issue makes the model useless for many applications. The adaptive gain model outperforms the non-adaptive model and the adaptive ARMAX model. The adaptive gain model is also very robust to the choice of the forgetting factor. As the model's pole is fixed, the model will always remain stable.

As explained above, at the SINTEF CZ plant the crucible containing the molten silicon is heated by a heating element, which power is manipulated by a TRIAC. Tight control of the silicon temperature is essential for producing high crystal quality. Initially, the temperature was controlled by a single-loop PID controller, using the TRIAC as controlling element. During experiments at the CZ process, process disturbances were observed at the heating element power. That is, there are responses in the power that can not be explained by the TRIAC. These power disturbances influence the silicon temperature. This PhD thesis suggests a cascade control strategy, which quick and effectively rejects the power disturbances. The cascade inner loop (slave control loop) controls the heating element power, using the TRIAC as controlling element. The cascade outer loop (master control loop) controls the silicon temperature. The temperature controller sets the reference (setpoint) to the power controller. The power controller's ability to quick and effectively reject the power disturbances is demonstrated on the SINTEF CZ process.

At the SINTEF CZ process, the pyrometer used for temperature control measures the temperature of a graphite material, not the temperature of the molten silicon. This pyrometer is referred to as the graphite pyrometer. Hence, using the graphite pyrometer for temperature control, the temperature of the graphite material is actually controlled, not the temperature of the molten silicon. During a CZ batch, another pyrometer was tested. This pyrometer is able to measure the

temperature directly in the molten silicon. This pyrometer is referred to as the silicon pyrometer. The output signal of the silicon pyrometer seems reasonable based on the melting point of silicon, and based on the signal's response to steps in the heating element power. Unfortunately, this signal has much high-frequency measurement noise. The noise can be attenuated using a traditional lowpass filter. However, such filtering will cause phase lag over the filter, which is unfortunate for the temperature control. This PhD thesis presents a sensor fusion algorithm which estimates the temperature of the molten silicon. The algorithm takes the silicon pyrometer and the graphite pyrometer as inputs. For a given lowpass filter cut-off frequency, the algorithm's temperature estimate has the same amount of measurement noise as a traditional lowpass filter, but significantly less phase lag. The sensor fusion algorithm works well within a limited temperature range. For the algorithm to handle larger temperature changes properly, more work must be done to understand and model the CZ process' nonlinear character.

The results presented above have been published in two journal articles and five conference papers. The conference papers are included in the conferences' respective proceedings.

Bibliography

Alic, M., Hauge, T. A., and Lie, B. (2009). Developing a simple Modelica library for simuluation of the Xstrata Nikkelverk copper production plant. In *50th International Conference of Scandinavian Simulation Society (SIMS 2009)*, Fredericia, Denmark.

Brown, R. G. and Hwang, P. Y. C. (1997). *Introduction to random signals and applied Kalman filtering - 3rd edition.* John Wiley & Sons Inc.

Chen, C. (1999). *Linear system theory and design, 3rd edition.* Oxford University Press, USA.

Di Ruscio, D. (1996). Combined deterministic and stochastic system identification and realization: DSR - a subspace approach based on observations. *Modeling, Identification and Control*, 17:193–230.

Di Ruscio, D. (2003). Subspace system identification of the Kalman filter. *Modeling, Identification and Control*, 24:125–157.

Di Ruscio, D. (2005). *DSR Toolbox for MATLAB.* Telemark University College, Porsgrunn, Norway.

Di Ruscio, D. (2008). Subspace system identification of the Kalman filter: Open and closed loop systems. In *The 6th international conference on Computing, Communication and Control Technologies (CCCT 2008)*, Orlando, Florida.

Ding, F. and Chen, T. (2005). Identification of Hammerstein nonlinear ARMAX systems. *Automatica*, 41:1479–1489.

Esbensen, K. H. (2002). *Multivariate data analysis - in practice, 5th edition.* CAMO Process AS, Oslo, Norway.

Gerald, C. F. and Wheatley, P. O. (2004). *Applied numerical analysis, 7th edition.* Pearson Education.

Gevelber, M., Wilson, D., and Duanmu, N. (2001). Modelling requirements for development of an advanced Czochralski control system. *Journal of Crystal Growth*, 230:217–223.

Hauge, T. A. (2003). *Roll-out of model based control with application to paper machines*. PhD thesis, Norwegian University of Science and Technology / Telemark University College, Trondheim / Porsgrunn, Norway.

Hauge, T. A. (2007). Cu process introduction. Xstrata Nikkelverk, Kristiansand, Norway.

Haugen, F. (1994). *Regulering av dynamiske systemer, bind 1*. Tapir Academic Press, Trondheim, Norway.

Haugen, F. (2001). *Regulering av dynamiske systemer, bind 2*. Tapir Academic Press, Trondheim, Norway.

Haugen, F. (2004). *Dynamic systems - modeling, analysis and simulation*. Tapir Academic Press, Trondheim, Norway.

Haugen, F. (2010). Comparing PI tuning methods in a real benchmark temperature control system. *Modeling, Identification and Control*, 31:79–91.

Hurle, D. T. J. (1993). Dynamics, stability and control of Czochralski growth. *Journal of Crystal Growth*, 128:15–25.

Irizarry-Rivera, R. and Seider, W. D. (1997a). Model-predictive control of the Czochralski crystallization process Part I. Conduction-dominated melt. *Journal of Crystal Growth*, 178:593–611.

Irizarry-Rivera, R. and Seider, W. D. (1997b). Model-predictive control of the Czochralski crystallization process Part II. Reduced-order convection model. *Journal of Crystal Growth*, 178:612–633.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45.

Katayama, T. (2005). *Subspace methods for system identification*. Springer-Verlag.

Komperød, M. (2008). System identification: Topics in data preprocessing and model realization, with real-life applications from the process industry. Master's thesis, Telemark University College, Porsgrunn, Norway.

Lan, C. W. (2004). Recent progress of crystal growth modeling and growth control. *Chemical Engineering Science*, 59:1437–1457.

Lee, K., Lee, D., Park, J., and Lee, M. (2005). MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 3:252–257.

Lie, B. and Hauge, T. A. (2008). Modeling of an industrial copper leaching and electrowinning process, with validation against experimental data. In *The 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008)*, Oslo, Norway.

Ljung, L. (1982). Aspects on the system identification problem. *Signal Processing*, 4:445–456.

Ljung, L. (1994). System identification in a MIC perspective. *Modeling, Identification and Control*, 15:153–159.

Ljung, L. (1999). *System identification - theory for the user, 2nd edition*. Prentice-Hall, Upper Saddle River, New Jersey.

Ljung, L. (2002). Recursive identification algorithms. *Circuits Systems Signal Processing*, 21:57–68.

Ljung, L. (2009). *System identification toolbox 7 - user's guide*. The MathWorks, Inc.

Ljung, L. and Glover, K. (1981). Frequency domain versus time domain methods in system identification. *Automatica*, 17:71–86.

Lyons, R. G. (2011). *Understanding digital signal processing - 3rd edition*. Prentice Hall.

Nilsen, G. W. (2006). *Topics in Open and Closed Loop Subspace System Identification: Finite Data-Based Methods*. PhD thesis, Norwegian University of Science and Technology / Telemark University College, Trondheim / Porsgrunn, Norway.

Park, J. S., Seo, M., Oh, H. J., and Jung, J. H. (2008). Silicon ingot diameter modeling in Czochralski process and its dynamic simulation. *Korean J. Chem. Eng.*, 25:623–630.

Rugh, W. J. (1996). *Linear system theory*. Prentice Hall, Upper Saddle River, New Jersey.

Seborg, D. E., Edgar, T. F., and Mellichamp, D. A. (2004). *Process dynamics and control, 2nd edition*. John Wiley and Sons, Inc., New Jersey, USA.

Skogestad, S. (2004). Simple analytic rules for model reduction and PID controller tuning. *Modeling, Identification and Control*, 25:85–120.

Skogestad, S. and Postlethwaite, I. (2005). *Multivariable feedback control - Analysis and design, 2nd edition*. John Wiley and Sons Ltd., England.

Sælid, S., Egeland, O., and Foss, B. (1985). A solution to the blow-up problem in adaptive controllers. *Modeling, Identification and Control*, 6:39–56.

Strang, G. (2003). *Introduction to linear algebra, 3rd edition*. Wellesley-Cambridge Press, USA.

Strang, G. (2006). *Linear algebra and its applications, 4th edition*. Thomson Brooks/Cole.

Van Overschee, P. and De Moor, B. (1995). A unifying theorem for three subspace system identification algorithms. *Automatica*, 31:1853–1864.

Wheeler, A. J. and Ganji, A. R. (2004). *Introduction to engineering experimentation, 2nd edition*. Pearson Education.

Wikipedia (2011). World energy consumption. [Online; accessed 13-June-2011].

Part II

Published Papers

Paper A

Preprocessing of Experimental Data for Use in Model Building and Model Validation

The candidate presented this paper at the 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008). The paper is also included in the conference's proceedings. The conference was held October 7th-8th 2008 at Oslo University College, Oslo, Norway. The paper was drafted during the candidate's master thesis (Komperød (2008)). The paper was finished and presented at the conference during the candidate's PhD study. The method for outlier detection presented in this paper is also applied in Paper D. After the paper, on page 59, a few considerations of the paper are discussed.

PREPROCESSING OF EXPERIMENTAL DATA FOR USE IN MODEL BUILDING AND MODEL VALIDATION

Magnus Komperød[1], **Tor Anders Hauge**[2], **Bernt Lie**[3]

[1]Østfold University College, Faculty of Engineering
Halden, Norway

[2]Xstrata Nikkelverk
Kristiansand, Norway

[3]Telemark University College, Faculty of Technology,
Department of Electrical Engineering,
Information Technology, and Cybernetics
Porsgrunn, Norway

Bernt.Lie@hit.no (Bernt Lie)

Abstract

A method for outlier detection based on the residual of system identification is presented in [1]. An alternative approach to this method is derived in this paper; outliers may cause large innovation processes (absolute values). Hence, such outliers may be detected by searching for samples having large innovation processes. In [2] it is proved that the innovation process can be identified directly from system input and system output data, without relying on models. This method is therefore well suited for outlier detection. In [3] it is shown that the innovation process identified by the method of [2] is identical to the residual of an ARX identification. Hence, the method to be derived in this paper can mathematically be proved to be a special case of the method presented in [1]: If an ARX algorithm is chosen for the system identification step in the method of [1], then the method of [1] and the method to be presented in this paper become mathematically identical. The latter method is tested on experimental data from the copper refining process of Xstrata Nikkelverk, Kristiansand, Norway. The MATLAB command `delayest` is used to estimate time delays between system inputs and system outputs in a dataset. A simulation study of `delayest` shows that the command is sensitive to: (i) The specified model order. (ii) Stochastic elements in the dataset. (iii) Whether all time delays are estimated during one execution of `delayest`, or whether only one time delay is estimated for each execution. `delayest` is also tested on experimental data from Xstrata Nikkelverk. These tests confirm that `delayest` is sensitive to the factors listed above.

Keywords

Data preprocessing; Outlier detection; Time delay estimation; Time delay identification.

1 Introduction

Modeling of dynamic systems is a most important part of today's science and engineering. Dynamic models serve many purposes. For example: (i) Training of process operators, pilots, and astronauts. (ii) Exploring systems in a different time scale than physical time. (iii) Testing systems by simulations before they are manufactured. For example ships, airplanes, missiles, and sub-sea oil installations. (iv) Model-based control, such as LQG control, model-based predictive control (MPC), linear and nonlinear decouplers, Smith predictors, etc.

In those cases where the systems to be modeled already exist and it is possible to log data from the systems, it may be desirable to include such experimental data in the modeling work. For mechanistic (first principle) modeling, experimental data may be used for parameter estimation. For empirical modeling (black-box modeling), including system identification, the models are built directly from experimental data. Experimental data is also essential with respect to model validation.

Experimental data logged from real-life systems, such as process industry, often requires preprocessing before they can be used for model building and model validation. This paper considers two sorts of data preprocessing that are frequently required on experimental data:

1. Outlier detection: In [1], a method for outlier de-

tection is presented. This method is based on the residuals of system identification: Samples having particularly large residuals (one-step-ahead prediction errors) should be inspected further as they may be associated with outliers. An alternative approach to this method for outlier detection will be derived in this paper.

2. Time delay estimation: The MATLAB command `delayest` is included in the MATLAB System Identification Toolbox. This command is used to estimate time delays between system inputs and system outputs in a dataset. A simulation study of `delayest` is presented in this paper. A suggestion for improvement that will make `delayest` less sensitive to the specified model order is also presented.

Both with respect to outlier detection and with respect to time delay estimation, real-life examples from the copper refining process at Xstrata Nikkelverk, Kristiansand, Norway, will be presented. Three process inputs, u^1, u^2, and u^3, and one process output, y, are considered in the examples. u^1 is the mass flow from the roasting furnace to the copper leaching process. u^2 is a recirculation flow from after the electro winning back to the copper leaching process. u^3 is the flow of sulphuric acid (H_2SO_4) to the copper leaching process. y is the concentration of sulphuric acid before the electro winning.

2 Notation and Definitions

The inputs to a system are collected in the input column vector, $u \in \mathbb{R}^{r\times 1}$, where r is the number of inputs. The outputs from a system are collected in the output column vector, $y \in \mathbb{R}^{m\times 1}$, where m is the number of outputs. A sub-script to these vectors refers to the sampling number. A super-script refers to the input or output number. For example u_k^2 refers to input no. 2 at sample no. k.

The symbol τ refers to true time delay (number of samples). τ is in general unknown and is subject to estimation. A super-script to τ, for example τ^2, refers to the time delay from input u^2 to the output y. $\hat{\tau}$ refers to time delay estimates. $\bar{\tau}$ refers to a time delay given as argument to a system identification algorithm. $\bar{\tau}$ may or may not be equal to the actual time delay, τ.

Def. 1 (Innovation Process)**.** The system output, y_k, may be decomposed into two components: (i) The component of y_k that can be predicted from previous inputs, $u_{-\infty}, \ldots, u_{k-1}$, and previous outputs, $y_{-\infty}, \ldots, y_{k-1}$, assuming no model errors. This predictable component of y_k is referred to as $\bar{y}_k$. (ii) The complement of $\bar{y}_k$, i.e. the component of y_k that can *not* be predicted from previous inputs and previous outputs. This is referred to as the innovation process, ε_k. Hence, $\varepsilon_k = y_k - \bar{y}_k$.

For bi-proper systems, i.e. systems having direct feed-through from the input, u_k, to the output, y_k, the current input, u_k, may also be included in the prediction of y_k. For simplicity, only strictly proper systems, i.e. systems without direct feed-through from u_k to y_k, will be considered in this paper.

The symbol ε is used for the true innovation process, which in general is unknown. The symbol ϵ is used for the identified innovation process. The identified innovation process is in general not exactly identical to the true innovation process.

Def. 2 (Orthogonal Projection)**.** The orthogonal projection of G onto H, G/H, is defined as in Eq. (1) [2].

$$G/H \stackrel{\text{def}}{=} GH^T(HH^T)^\dagger H \tag{1}$$

The super-script † refers to the Moore-Penrose pseudo-inverse.

Def. 3 (Complement of Orthogonal Projection)**.** The complement of the orthogonal projection of G onto H, $GH^\perp$, is defined by Eq. (2) [2].

$$GH^\perp \stackrel{\text{def}}{=} G - G/H \stackrel{\text{def}}{=} G - GH^T(HH^T)^\dagger H \tag{2}$$

Def. 4 (ARMAX Model Form)**.** Eq. (3) defines the general form of ARMAX models [1].

$$A(q)y_k = B(q)u_k + C(q)\varepsilon_k \tag{3}$$

In Eq. (3), q is the time-shift operator of the Z-transform, i.e. $q^{-1}\, y_k = y_{k-1}$. The symbol q is commonly used within the subject of system identification. The symbol z is used in many other contexts. $A(q)$, $B(q)$, and $C(q)$ are polynomials. n_A, n_B, and n_C are the number of coefficients in the polynomials that in general are different from 1. The $A(q)$ polynomial and the $C(q)$ polynomial are monic polynomials, i.e. the coefficient of their highest order term is 1.

Def. 5 (ARX Model Form)**.** Eq. (4) defines the general form of ARX models [1].

$$A(q)y_k = B(q)u_k + \varepsilon_k \tag{4}$$

$A(q)$ is a monic polynomial.

3 Outlier Detection

Outliers may be data that for one sample, or for a few samples, go to unlikely values, and then back to normal values. Such outliers can usually be seen by plotting the dataset. However, as shown in [1], outliers may also be data that are *not* unlikely values, but rather average values. These data can for example be outliers because the derivative of the data changes unlikely fast, i.e. the second derivative has unlikely large absolute value.

In [1, Example 14.1] the following method for outlier detection is suggested: (i) Identify an empirical model based on the dataset using a system identification algorithm. In [1, Example 14.1], a 3rd order ARMAX

model is used. (ii) Run a one-step-ahead prediction simulation using the empirical model and the dataset. (iii) Plot the one-step-ahead prediction errors from the simulation to detect outliers: Large prediction errors (absolute values) may be caused by outliers.

3.1 An Alternative Approach to a Method for Outlier Detection Presented in [1]

This subsection derives an alternative approach to the method for outlier detection presented in [1]. This alternative approach is based on sub-space system identification, in particular the work presented in [2].

Most real-life systems do have some innovation process, ε. The innovation process is typically caused by unmeasured process disturbances and / or measurement noise. Assume that there is a measurement error in one of the variables used to compute $\bar{y}_k$ or in y_k that is significantly larger than the normal measurement noise. This is likely to cause a mismatch between y_k and $\bar{y}_k$ that is also significantly larger than normal. According to Def. 1: Assuming that the model used to compute $\bar{y}_k$ has no modeling error, then the mismatch between y_k and $\bar{y}_k$ is the innovation process, ε_k. Hence, measurement errors may be detected by looking for samples having unusually large innovation processes (absolute values).

In [2] it is proved that for linear time-invariant (LTI) systems, the innovation process, ε_k, can be identified directly from previous inputs, $u_{-\infty}, \ldots, u_{k-1}$, and previous outputs, $y_{-\infty}, \ldots, y_{k-1}$, without relying on models. As infinite number of preceding samples can not be used in any practical problems, only the J preceding samples are used, where J is a user-specified parameter. The identified innovation process, ϵ_k, is the complement of the orthogonal projection of current outputs, y_k, onto inputs and outputs from the J preceding samples, $u_{k-J}, \ldots, u_{k-1}$ and $y_{k-J}, \ldots, y_{k-1}$ [2]. This can mathematically be written as Eq. (5). The matrices of Eq. (5) are as presented in Eq. (6) to Eq. (8). K is the number of columns of the matrices of Eq. (6) to Eq. (8) [2, 4, 3]. (Eq. (5) to Eq. (8) are obtained by choosing $L = 1$ and $g = 0$ in the derivation of [2], as shown in [5, 4, 3].)

$$\epsilon_{J|1} = Y_{J|1} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^{\perp} \tag{5}$$

$$Y_{J|1} = \begin{bmatrix} y_J & y_{J+1} & \ldots & y_{J+K-1} \end{bmatrix} \in \mathbb{R}^{m \times K} \tag{6}$$

$$\epsilon_{J|1} = \begin{bmatrix} \epsilon_J & \epsilon_{J+1} & \ldots & \epsilon_{J+K-1} \end{bmatrix} \in \mathbb{R}^{m \times K} \tag{7}$$

$$\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} = \begin{bmatrix} u_0 & u_1 & \ldots & u_{K-1} \\ u_1 & u_2 & \ldots & u_K \\ \vdots & \vdots & \ddots & \vdots \\ u_{J-1} & u_J & \ldots & u_{K+J-2} \\ y_0 & y_1 & \ldots & y_{K-1} \\ y_1 & y_2 & \ldots & y_K \\ \vdots & \vdots & \ddots & \vdots \\ y_{J-1} & y_J & \ldots & y_{K+J-2} \end{bmatrix} \in \mathbb{R}^{(r+m)J \times K} \tag{8}$$

The method for identifying the innovation process given by Eq. (5) to Eq. (8) is equivalent to the first step of the DSR E subspace system identification algorithm [5, 4]. The DSR E algorithm is presented in [5, 4]. DSR E should not be confused with the DSR algorithm, although DSR and DSR E are closely related.

The identified innovation process, ϵ_k, is plotted for all timesteps, k (except for the first J timesteps, which are used for initialization purposes). The upper subplot of Fig. 1 shows an ϵ plot. Timesteps having particularly large innovation processes (absolute values) can easily be identified as peaks in the ϵ plot. These timesteps and the surrounding timesteps are candidates for being outliers and should be subject to further inspection.

In [3] it is proved that the innovation process identified by the method of [2], i.e. Eq. (5) to Eq. (8), is mathematically identical to the residual from identification of a strictly proper ARX model where $n_A = n_B = J$. This residual can also be expressed as the one-step-ahead prediction error when simulating the identified ARX model on its own training dataset. Hence, the method for outlier detection based on the identified innovation process, as presented above, is a special case of the method explained in [1, Example 14.1]: If choosing a strictly proper ARX model where $n_A = n_B = J$ in the system identification step of [1, Example 14.1], then the two methods become mathematically identical.

With respect to the method of [1] and the method de-

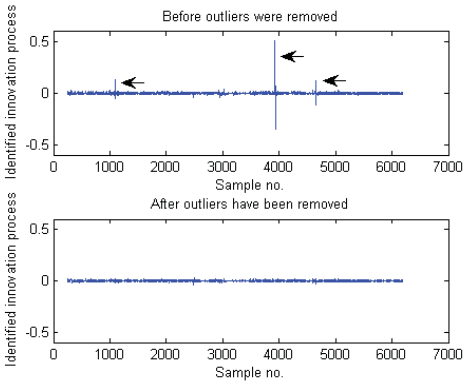


Fig. 1 The innovation process identified by Eq. (5) to Eq. (8) before (upper subplot) and after (lower subplot) outliers have been removed.

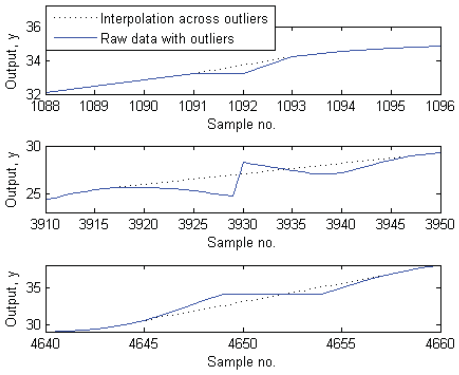


Fig. 2 The process output, y, zoomed in at the samples marked by arrows in Fig. 1.

rived above, it is most important to be aware that these methods identify samples having large identified innovation processes / prediction errors. One should *not* conclude that samples having large identified innovation processes in general are *equivalent* to samples associated with outliers: (i) Large innovation processes may be caused by other factors than outliers, for example large, sudden process disturbances. In other words; large innovation process at a sample indicates that this sample and the surrounding samples should be objects to further inspection. Whether it actually is an outlier, and if so, how to handle it, is a decision to be taken by the human data analyzer (or by other methods). (ii) The identified innovation process may be significantly different from the true innovation process. For example, the method of [1] and the method derived above assume linear time-invariant (LTI) systems (unless a nonlinear system identification method is used in the method of [1]). Strongly nonlinear systems may cause large deviations between the true innovation process and the identified innovation process. (iii) There may also be outliers that do not cause particularly large innovation process and are therefore not detected by these methods.

3.2 Outlier Detection on Industrial Data

This subsection presents a real-life industrial example of the method for outlier detection presented in Subsec. 3.1. The example is based on experimental data from the copper refining process at Xstrata Nikkelverk, Kristiansand, Norway. This process is briefly explained in Sec. 1.

The innovation process identified in a dataset from Xstrata Nikkelverk is shown in the upper subplot of Fig. 1. The innovation process was identified using Eq. (5) to Eq. (8). The three largest peaks of the ϵ plot, indicated by arrows, will be considered in this example. The exact sample numbers of these peaks must be identified. The zoom options of MATLAB figures may be used for this purpose.

The subplots of Fig. 2 show the output, y, zoomed in at the samples indicated by arrows in Fig. 1. The solid lines in Fig. 2 show y before outliers were removed. The samples indicated by arrows in Fig. 1 are all at samples where the derivative of y changes very fast, i.e. the second derivative has high absolute value. These data were classified as outliers and were removed by linear interpolations across them. These interpolations are shown by dotted lines in Fig. 2. After these outliers were removed, the innovation process was re-identified using Eq. (5) to Eq. (8). The new ϵ plot is shown in the lower subplot of Fig. 1. The three peaks marked by arrows in the upper subplot are no longer present. Hence, it is reasonable to conclude that these peaks were caused by the outliers shown in Fig. 2.

Consider criterion V defined by Eq. (9).

$$V \stackrel{\text{def}}{=} \frac{1}{N} \sum_{k=1}^{N} {\epsilon_k}^2 \tag{9}$$

In Eq. (9), N is the number of samples in the dataset (minus the number of samples used for initialization purposes and to compensate for time delays). Criterion V is a commonly used model fit criterion within the subject of system identification as ϵ is identical to the one-step-ahead prediction error. The value of V was reduced by more than 80% when removing the outliers. Hence, removing the outliers resulted in significantly better model fit.

4 Time Delay Estimation

When building models using system identification, time delays from system inputs to system outputs may cause problems if not handled properly. If the time delays are not compensated for, a large number of model states may be necessary to model the dynamics of the time delays. Too many model states are unfortunate because the identified models may be over-fitted. Over-fitted

models may perform very well on their respective training datasets, but such models will in general perform poorly on independent validation datasets. If the time delays are known, they may be compensated for by shifting the input dataseries with respect to the output dataseries.

This section considers the MATLAB command `delayest`, which is included in the MATLAB System Identification Toolbox [6]. This command serves the purpose of estimating time delays from the system inputs, u, to a system output, y, based on a dataset. The command can only handle single output systems, i.e. $y \in \mathbb{R}^{1\times 1}$. Quoting [6]:

The `delayest` command estimates the time delay in a dynamic system by estimating a low-order, discrete-time ARX model with a range of delays, and then choosing the delay that corresponding to the best fit.

Subsec. 4.1 presents a simulation study of `delayest`. This subsection also includes a suggestion for improvement of `delayest`. In Subsec. 4.2, `delayest` is tested on experimental data from the copper refining process at Xstrata Nikkelverk; the estimates of `delayest` are compared to the estimates of [7], which are based on process knowledge.

The following properties of the time delay estimates made by `delayest` will be considered: (i) The accuracy of the estimates, i.e. how close the estimates are to the actual time delays (or to the estimates of [7]). (ii) Whether the time delay estimates are sensitive to the model order specified to `delayest`. (iii) Whether the time delays estimates are sensitive to stochastic elements in the dataset. (iv) Whether the time delay estimates are sensitive to whether the system is considered as one 3×1 system or as three 1×1 systems. In the 3×1 case, all three inputs, u^1, u^2 and u^3, are provided to `delayest` in one single execution of the command. `delayest` then estimates $\hat{\tau}^1$, $\hat{\tau}^2$ and $\hat{\tau}^3$ by making ARX models based on u^1, u^2 and u^3, and of course y. In the 1×1 case, only one input is provided to `delayest` at each execution. Hence, three executions are needed to estimate the three time delays. The 1×1 case is to be preferred with respect to computational efficiency [3].

4.1 Simulation Study of Time Delay Estimation

In the experiments presented in this subsection, two models are used: (i) A single input, single output (SISO) ARMAX model on the form of Eq. (10). This model is referred to as model no. 1. (ii) A multiple input, single output (MISO) ARMAX model on the form of Eq. (11). This model is referred to as model no. 2.

$$A(q)y_k = B^1(q)u_k^1 + C(q)\varepsilon_k \tag{10}$$

$$A(q)y_k = B^1(q)u_k^1 + B^2(q)u_k^2 + B^3(q)u_k^3 + C(q)\varepsilon_k \tag{11}$$

For the polynomials of Eq. (10) and Eq. (11), $n = n_A = n_{B^1} = n_{B^2} = n_{B^3} = n_C = 6$. Both models are asymptotically stable.

The experiment to be presented in the following aims to explore the accuracy of `delayest` and which factors the command are sensitive to. `delayest` is tested on two datasets generated by simulations based on model no. 2. The input data used in the simulations, u^1, u^2, and u^3, are based on experimental data from Xstrata Nikkelverk. One dataset was generated as no innovation process, ε, was applied to the simulation. This dataset is referred to as the deterministic case. The other dataset was generated as a white noise sequence was applied for simulating the innovation process, ε. This dataset is referred to as the stochastic case. Except for the innovation process, the datasets were generated under identical condition. After the simulations, the input dataseries, u^1, u^2 and u^3, are shifted with respect to the output, y, so that $\tau^1 = 20$, $\tau^2 = 30$, and $\tau^3 = 50$. The deterministic and the stochastic datasets are then provided to `delayest` for estimation of the time delays, τ^1, τ^2, and τ^3.

`delayest` requires the user to specify intervals of possible values for $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$. `delayest` will only consider these intervals when estimating time delays. In the simulation study presented here, these intervals are set to ± 10 the true time delays: $\hat{\tau}^1$ is specified to be in the interval $[10, 30]$, $\hat{\tau}^2$ is specified to be in the interval $[20, 40]$, and $\hat{\tau}^3$ is specified to be in the interval $[40, 60]$. `delayest` also requires the user to specify $\bar{n}_A$, $\bar{n}_{B^1}$, $\bar{n}_{B^2}$, and $\bar{n}_{B^3}$ of the ARX models to be identified during execution of the command. The bar symbol is here used to avoid confusion with the corresponding values of the ARMAX models, i.e. Eq. (10) and Eq. (11). In this simulation study, these values are always chosen so that $\bar{n}_A = \bar{n}_{B^1} = \bar{n}_{B^2} = \bar{n}_{B^3}$. This value will be referred to as $\bar{n}$.

Fig. 3 to Fig. 6 show the estimates of `delayest`, $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$, plotted as functions of $\bar{n}$. Fig. 3 shows the deterministic 3×1 case, Fig. 4 shows the stochastic 3×1 case, Fig. 5 shows the deterministic 1×1 case, and Fig. 6 shows the stochastic 1×1 case. These simulations show that the estimates of `delayest` are sensitive to:

1. The value of $\bar{n}$. If `delayest` was not sensitive to the value of $\bar{n}$, then the curves in Fig. 3 to Fig. 6 would have been horizontal lines.
2. Stochastic elements in the dataset. If `delayest` was not sensitive to stochastic elements, then the curves of Fig. 3 and Fig. 4 would have been identical. Likewise for Fig. 5 and Fig. 6.
3. Whether the 3×1 approach or the 1×1 approach is used. If `delayest` was not sensitive to this, the curves of Fig. 3 and Fig. 5 would have been identical. Likewise for Fig. 4 and Fig. 6.

In the simulations presented in Fig. 3 to Fig. 6, `delayest` estimates τ^1, τ^2, *and* τ^3 exactly correct

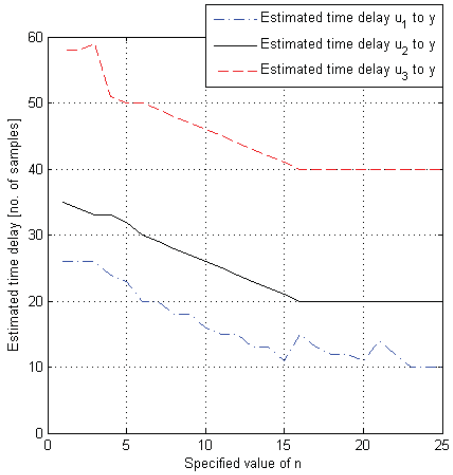


Fig. 3 Time delay estimates, $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$, plotted as functions of $\bar{n}$ for the deterministic 3×1 case.

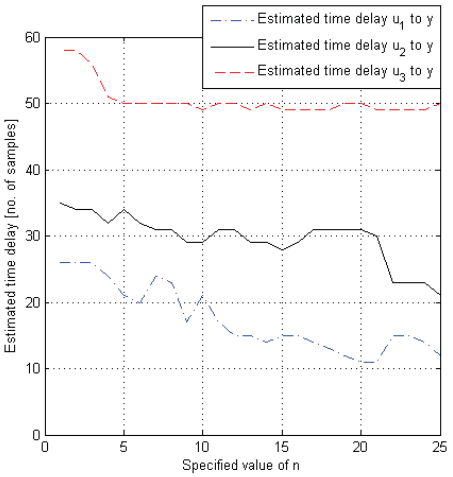


Fig. 5 Time delay estimates, $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$, plotted as functions of $\bar{n}$ for the deterministic 1×1 case.

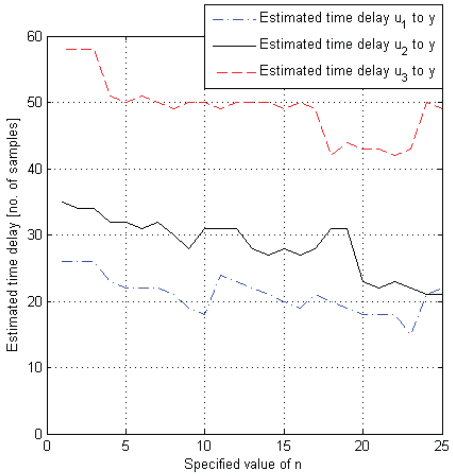


Fig. 4 Time delay estimates, $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$, plotted as functions of $\bar{n}$ for the stochastic 3×1 case.

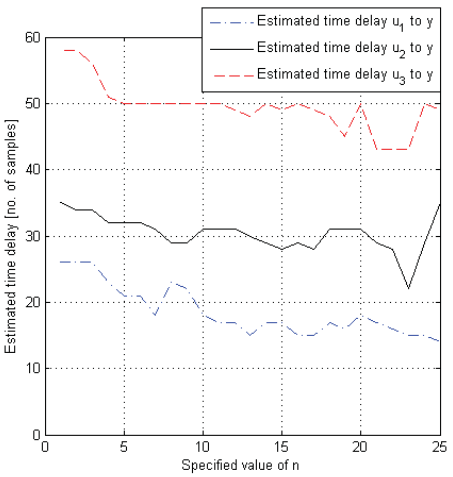


Fig. 6 Time delay estimates, $\hat{\tau}^1$, $\hat{\tau}^2$, and $\hat{\tau}^3$, plotted as functions of $\bar{n}$ for the stochastic 1×1 case.

only when the following three criteria are met: (i) The correct polynomial orders are specified, i.e. $\bar{n} = 6$. (ii) Stochastic elements are *not* present in the dataset, i.e. $\varepsilon = 0$. (iii) The 3×1 approach is used. In any other cases, `delayest` fails to make exactly correct estimates for all three time delays.

In particular the deterministic 3×1 case, i.e. Fig. 3, and to some extend also the other figures show that there is a *systematic* error present: The time delay estimates tend to decrease as $\bar{n}$ increases. In the deterministic 3×1 case, $\hat{\tau}^2$ and $\hat{\tau}^3$ decrease as $\bar{n}$ increase from $\bar{n} = 6$ until $\hat{\tau}^2$ and $\hat{\tau}^3$ saturate at the lower ends of their specified intervals, i.e. at $\bar{n} = 16$. Over the interval $\bar{n} \in [6, 16]$, $\bar{n}$ has increased by 10 and $\hat{\tau}^2$ and $\hat{\tau}^3$ have decreased by 10.

In the following, a suggestion for improvement of `delayest` is derived. This improvement makes `delayest` less sensitive for the value of $\bar{n}$. The improvement works only for deterministic, linear time-invariant (LTI) systems. The derivation presented here is based on a single input, single output (SISO) system. In [3] it is shown that this improvement can easily be extended to multiple input, single output (MISO) systems.

Based on model no. 1, a deterministic dataset was generated using an input sequence, u^1, from a dataset of Xstrata Nikkelverk. No innovation process, ε, was applied to the simulation. The input dataserie, u^1, was shifted with respect to the output dataserie, y, so that $\tau^1 = 20$.

Please note the difference between the symbols ε and ϵ. ε is the innovation process generated by a random generator. In this derivation $\varepsilon = 0$. ϵ is the one-step-ahead prediction error. In the following, ϵ refers to the one-step-ahead prediction error as a model is simulated on its own training dataset.

For a given dataset and a given system identification algorithm, in this case ARX, the one-step-ahead prediction error, ϵ, and inherently the model fit criterion, V, as defined in Eq. (9), are functions of the parameters specified to the ARX system identification algorithm: (i) The specified time delay, $\bar{\tau}^1$, and (ii) the specified values of $\bar{n}_A$ and $\bar{n}_{B^1}$. In this derivation, $\bar{n}_A$ and $\bar{n}_{B^1}$ are always chosen so that $\bar{n}_A = \bar{n}_{B^1}$. This value will be referred to as $\bar{n}$. Hence, $\epsilon = \epsilon\left(\bar{\tau}^1, \bar{n}\right)$ and $V = V\left(\bar{\tau}^1, \bar{n}\right)$.

Fig. 7 shows V plotted as function of $\bar{\tau}^1$ for $\bar{n} = 5$, for $\bar{n} = 6$, and for $\bar{n} = 12$. Please note that: (i) For $\bar{n} = 5$, V does not reach zero for any value of $\bar{\tau}^1$. (ii) For $\bar{n} = 6$, i.e. the same value as for model no. 1: V does reach zero ($< 10^{-28}$) for $\bar{\tau}^1 = \tau^1 = 20$, i.e. for the correct time delay, but not for any $\bar{\tau}^1 \neq \tau^1$. (iii) For $\bar{n} = 12$, V is zero ($< 10^{-28}$) for $\bar{\tau}^1 \in [14, 20]$. Values below 10^{-28} are considered as zero as it is assumed that these values differ from zero only due to numerical reasons.

Further simulations were run, which are *not* presented in this paper. These simulations indicate that the observations presented in Fig. 7 can be generalized to the following two statements; Statement (i): For $\bar{n} < n$, V does not reach 0 for any $\bar{\tau}^1$. Statement (ii): For $\bar{n} \geq n$, $V = 0$ for any $\bar{\tau}^1 \in [\tau^1 - (\bar{n} - n), \tau^1]$.

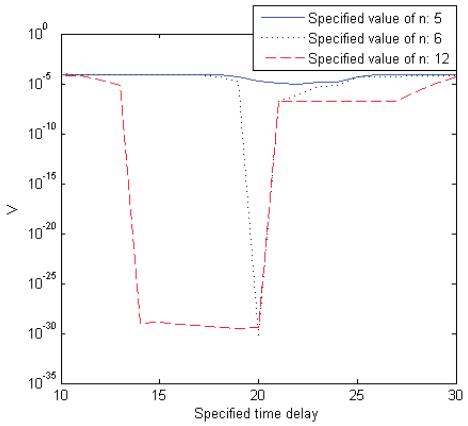


Fig. 7 V plotted as function of $\bar{\tau}^1$ for $\bar{n} = 5$, for $\bar{n} = 6$, and for $\bar{n} = 12$. The Y axis is logarithmic.

Statement (i) has a trivial explanation: As the polynomials of the ARX model have lower order than the polynomials of model no. 1, which was used to generate the dataset, the ARX model does not have enough poles nor enough zeros to perfectly explain the dynamics of model no. 1.

Statement (ii) can be explained by considering the ARX regression matrix. For simplicity, a deterministic SISO system were $n = n_A = n_B = 2$ is used for illustration. The time delay of the system is $\tau = 20$ samples. Hence, the system is on the form of Eq. (12).

$$y_k = -a_1 y_{k-1} - a_2 y_{k-2} + b_1 u_{k-20} + b_2 u_{k-21} \quad (12)$$

Now assume that a dataset is generated by applying an input signal, u, to the system of Eq. (12). Further assume that u is *not* a periodic signal. The generated dataset is now to be used for system identification of the system presented in Eq. (12). If the value of n is known, but the time delay, τ, is unknown, the ARX regression problem is on the form of Eq. (13). For simplicity, only one row of the regression matrix is shown.

$$y_k = \begin{bmatrix} -y_{k-1} & -y_{k-2} & u_{k-\bar{\tau}} & u_{k-\bar{\tau}-1} \end{bmatrix} \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \bar{b}_1 \\ \bar{b}_2 \end{bmatrix} \quad (13)$$

In Eq. (13), $\bar{\tau}$ is a time delay specified to the ARX system identification algorithm. For this ARX regression

problem to have no residual, the columns u_{k-20} and u_{k-21} must be included in the regression matrix. This is true if, and only if, $\bar{\tau}$ is chosen identical to the true time delay, i.e. 20 samples. For any $\bar{\tau} \neq 20$, there will be residuals. Hence, the true time delay can be identified by considering the residual for various values of $\bar{\tau}$. This is identical to the curve representing $\bar{n} = 6$ shown in Fig. 7.

If also the value of n is unknown, an arbitrary $\bar{n} \geq n$ is chosen. Assume $\bar{n} = 4$ is chosen. The ARX regression problem is now on the form of Eq. (14).

$$y_k = \begin{bmatrix} -y_{k-1} & -y_{k-2} & -y_{k-3} & -y_{k-4} & u_{k-\bar{\tau}} & u_{k-\bar{\tau}-1} & u_{k-\bar{\tau}-2} & u_{k-\bar{\tau}-3} \end{bmatrix} \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \bar{a}_3 \\ \bar{a}_4 \\ \bar{b}_1 \\ \bar{b}_2 \\ \bar{b}_3 \\ \bar{b}_4 \end{bmatrix} \tag{14}$$

Also in the case of Eq. (14), the columns u_{k-20} and u_{k-21} must be included in the regression matrix to avoid residuals. However, in this case, this is achieved for three different values of $\bar{\tau}$:

$$\begin{aligned} \bar{\tau} = 20 \quad &\Rightarrow \quad [u_{k-\bar{\tau}}, u_{k-\bar{\tau}-1}] = [u_{k-20}, u_{k-21}] \\ \bar{\tau} = 19 \quad &\Rightarrow \quad [u_{k-\bar{\tau}-1}, u_{k-\bar{\tau}-2}] = [u_{k-20}, u_{k-21}] \\ \bar{\tau} = 18 \quad &\Rightarrow \quad [u_{k-\bar{\tau}-2}, u_{k-\bar{\tau}-3}] = [u_{k-20}, u_{k-21}] \end{aligned}$$

This is identical to the results presented in Fig. 7: The residual is zero for $\bar{\tau} \in [\tau - (\bar{n} - n), \tau]$. In the case of Eq. (14) the residual is zero for $\bar{\tau} \in [20 - (4 - 2), 20]$, i.e. $\bar{\tau} \in [18, 20]$. The highest value of $\bar{\tau}$ that gives no residual is the correct time delay.

Statement (ii) can be used to identify the value of τ^1: As $V = 0$ for $\bar{\tau}^1 \in [\tau^1 - (\bar{n} - n), \tau^1]$, τ^1 is given by the highest values of $\bar{\tau}^1$ giving $V = 0$. This value can be read from Fig. 7: For both $\bar{n} = 6$ and for $\bar{n} = 12$, the highest value of $\bar{\tau}$ giving $V = 0$ is $\bar{\tau} = 20$, which is the correct time delay. Statement (ii) can not be used for $\bar{n} = 5 < n$ because the statement assumes $\bar{n} \geq n$.

Statement (ii) can also be used to identify the value of n: Define $\bar{\tau}^1_{min}(\bar{n})$ as the lowest $\bar{\tau}^1$ giving $V = 0$. According to statement (ii), $\bar{\tau}^1_{min}(\bar{n})$ is given by $\bar{\tau}^1_{min}(\bar{n}) = \tau^1 - (\bar{n} - n)$. $\bar{\tau}^1_{min}(\bar{n})$ can be read from the plot shown in Fig. 7. Hence, as τ^1, $\bar{\tau}^1_{min}(\bar{n})$, and $\bar{n}$ are known, n can be calculated by $n = \bar{\tau}^1_{min}(\bar{n}) - \tau^1 + \bar{n}$.

For the curve representing $\bar{n} = 6$ in Fig. 7: $\tau^1 = 20$ (see two paragraphs above), $\bar{\tau}^1_{min}(\bar{n}) = 20$ (read from Fig. 7), and $\bar{n} = 6$ (specified). Hence, $n = \bar{\tau}^1_{min}(\bar{n}) - \tau^1 + \bar{n} = 20 - 20 + 6 = 6$, which is the correct value.

For the curve representing $\bar{n} = 12$ in Fig. 7: $\tau^1 = 20$ (see three paragraphs above), $\bar{\tau}^1_{min}(\bar{n}) = 14$ (read from Fig. 7), and $\bar{n} = 12$ (specified). Hence, $n = \bar{\tau}^1_{min}(\bar{n}) - \tau^1 + \bar{n} = 14 - 20 + 12 = 6$, which is the correct value.

In the derivation presented above, it has been assumed that $n_A = n_B$. The method presented above actually identifies n_B, i.e. if the assumption $n_A = n_B$ is violated, the identified value of n is equal to n_B and different from n_A. This can be seen from the explanation given by Eq. (12) to Eq. (14): The value of $\bar{n}_A$ does not influence for which value of $\bar{\tau}$ there is no residual. Of course $\bar{n}_A$ must be chosen so that $\bar{n}_A \geq n_A$, otherwise there will always be residuals regardless of the value of $\bar{\tau}$.

4.2 Time Delay Estimation on Industrial Data

The `delayest` command has been tested on a dataset from the copper refining process at Xstrata Nikkelverk. This process is briefly explained in Sec. 1. As the actual system order is unknown, `delayest` has been tested for four different values of $\bar{n}$: $\bar{n} = 3$, $\bar{n} = 5$, $\bar{n} = 10$, and $\bar{n} = 20$, where $\bar{n} = \bar{n}_A = \bar{n}_{B^1} = \bar{n}_{B^2} = \bar{n}_{B^3}$. For each of the $\bar{n}$ values, both the 3×1 case and the 1×1 case were tested. The time delays, τ^1, τ^2, and τ^3, were all specified to be within the interval $[15, 110]$.

In [7] it is provided rough time delay estimates based on process knowledge. In Fig. 8, the estimates of `delayest` are compared to the estimates of [7]. The figure shows that: (i) There are large variations within the estimates of `delayest`. This confirms that `delayest` is sensitive to the specified values of $\bar{n}$ and to whether the 3×1 approach or the 1×1 approach is used. (ii) There are in general large deviations between the estimates of [7] and the estimates of `delayest`.

5 Conclusions

Outliers in a dataset may cause innovation processes, which are significantly larger (absolute value) than usual. Hence, searching for samples having large innovation processes may be used as a method for outlier detection. In [2] it is proved that for linear time-invariant (LTI) systems the innovation process can be identified directly from input and output data, without relying on models. This method is therefore well suited for outlier detection. In [3] it is shown that the innovation process

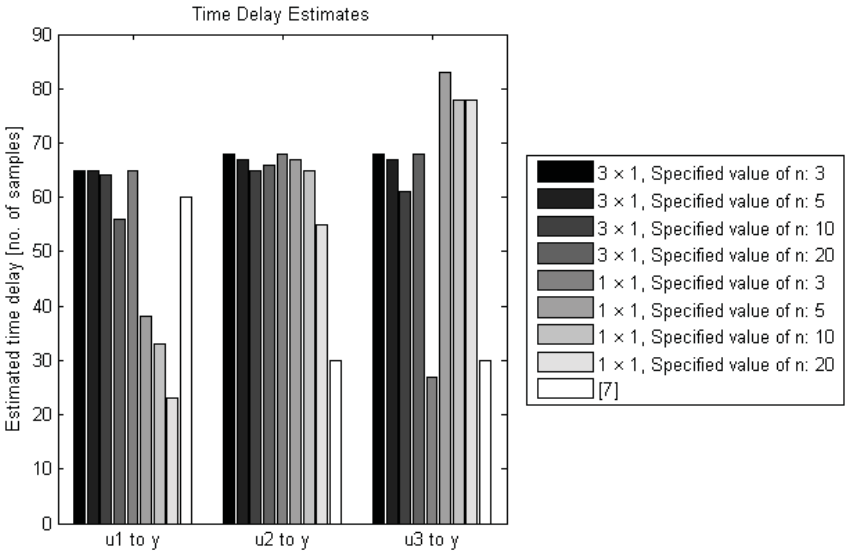


Fig. 8 Time delay estimates of `delayest` and of [7].

identified by the method of [2] is mathematically identical to the residual of an ARX identification of which $n_A = n_B = J$. It then follows that using the method of [2] for outlier detection is a special case of a method for outlier detection presented in [1, Example 14.1].

The method for outlier detection based on [2] has been tested on industrial data from Xstrata Nikkelverk, Kristiansand, Norway. The method detected three outliers that would not have been detected by plotting the raw data. The detected data are outliers because the derivative of the output changes unlikely fast, i.e. the second derivative of the output has unlikely large absolute values.

The MATLAB command `delayest` is used to estimate time delays between system inputs and system outputs in datasets. The command has been tested in a simulation study. The results from this study show that `delayest` is sensitive to: (i) The specified model order. (ii) Stochastic elements in the dataset. (iii) Whether all time delays are estimated during one execution of `delayest`, or whether only one time delay is estimated for each execution. The simulation study also indicates that `delayest` has a *systematic* error: The time delay estimates tend to decrease as the system order increases. In simulation of a 3 input, 1 output system, `delayest` estimated the time delays from the three inputs to the output exactly correct only when the following conditions were met: (i) The correct system order was specified. (ii) No stochastic elements were present. (iii) All three time delays were estimated during one execution of `delayest`.

`delayest` was also tested on industrial data from Xstrata Nikkelverk, Kristiansand, Norway. These tests confirmed that `delayest` is sensitive to: (i) The specified model order. (ii) Whether all time delays are estimated during one execution of `delayest`, or whether only one time delay is estimated for each execution. Whether `delayest` is sensitive to stochastic elements in the dataset could not be tested on the industrial data. In general, the estimates of `delayest` deviated significantly from estimates based on process knowledge.

In this paper, a suggestion to improvement of `delayest` is presented. This improvement makes `delayest` less sensitive for unknown system order. This improvement works only for deterministic LTI systems.

6 Acknowledgments

The authors are most grateful to Xstrata Nikkelverk, Kristiansand, Norway, for providing the datasets from the copper refining process and for allowing these data to be used in this paper.

7 References

[1] L. Ljung. *System Identification - Theory for the User, 2nd Edition*. Prentice-Hall, Upper Saddle River, New Jersey, 1999.

[2] D. Di Ruscio. Subspace system identification of the Kalman filter. *Modeling, Identification and Control*, 24:125–157, 2003.

[3] M. Komperød. *System Identification: Topics in Data Preprocessing and Model Realization, with Real-Life Applications from the Process Industry,*

Master Thesis. Telemark University College, Porsgrunn, Norway, 2008.

[4] GW. Nilsen. *Topics in Open and Closed Loop Subspace System Identification: Finite Data-Based Methods, Ph.D. Thesis*. Norwegian University of Science and Technology / Telemark University College, Trondheim / Porsgrunn, Norway, 2006.

[5] D. Di Ruscio. Subspace system identification of the Kalman filter: Open and closed loop systems. In *The 6th international conference on Computing, Communication and Control Technologies: CCCT 2008*, Orlando, Florida, June 29th - July 2rd 2008.

[6] The MathWorks, Inc. *The System Identification Toolbox*, 2008.

[7] TA. Hauge. E-mail communication, dated Mars 13, 2008.

Considerations of Paper A

In Section 4.1 of the paper, it is pointed out that there is a *systematic* error in the time delay estimates. The time delay estimates tend to decrease as the specified model order increases. However, this systematic error seems to be valid only for deterministic systems. At Figures 4 and 6, which represent stochastic systems, the negative correlation between the time delay estimates and the model order seems to be weak. In Figure 8, which considers logged process data from the process industry, the negative correlation seems to be present from u^1 to y and from u^2 to y, but not from u^3 to y.

The paper uses the notation $\bar{y}$ for prediction of the process output y. As the notation $\hat{y}$ is common in systems and control engineering, it is unfortunate not to use this notation.

Paper B

Empirical Modeling: Approximating the DSR E Sub-Space System Identification Algorithm by a Two-Step ARX Algorithm

The candidate presented this paper at the 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008). The paper is also included in the conference's proceedings. The conference was held October 7th-8th 2008 at Oslo University College, Oslo, Norway. The paper was drafted during the candidate's master thesis (Komperød (2008)). The paper was finished and presented at the conference during the candidate's PhD study. After the paper, on page 73, a few considerations of the paper are discussed.

EMPIRICAL MODELING: APPROXIMATING THE DSR E SUB-SPACE SYSTEM IDENTIFICATION ALGORITHM BY A TWO-STEP ARX ALGORITHM

Magnus Komperød[1], Tor Anders Hauge[2], David Di Ruscio[3], Bernt Lie[3]

[1]Østfold University College, Faculty of Engineering
Halden, Norway

[2]Xstrata Nikkelverk
Kristiansand, Norway

[3]Telemark University College, Faculty of Technology,
Department of Electrical Engineering,
Information Technology and Cybernetics
Porsgrunn, Norway

Bernt.Lie@hit.no (Bernt Lie)

Abstract

DSR E is a sub-space system identification algorithm for solving deterministic / stochastic linear time-invariant (LTI) system identification problems. DSR E consists of two steps: (i) The innovation process is identified. (ii) The deterministic / stochastic system identification problem is reduced to a deterministic problem by considering the identified innovation process as a deterministic input. In the DSR E algorithm presented in [1], this deterministic problem is solved using a deterministic sub-space system identification method. ARX is a system identification algorithm based on the ordinary least squares (OLS) method. In addition to the basic, single-step ARX algorithm, there are various multi-step ARX algorithms. In this paper it is shown that the first step of DSR E is mathematically identical to the single-step ARX algorithm. The second step, which is a deterministic problem, may also be solved by single-step ARX. Hence, each of the two steps of the DSR E algorithm may be replaced by single-step ARX, allowing DSR E to be approximated by a two-step ARX algorithm. DSR E and its two-step ARX approximation are compared by modeling a section of the copper refining process at Xstrata Nikkelverk, Kristiansand, Norway.

Keywords

ARX; DSR E; Sub-space system identification; System identification.

1 Introduction

Modeling of dynamic systems is a most important part of today's science and engineering. Dynamic models serve many purposes, for example: (i) Training of process operators, pilots and astronauts. (ii) Exploring systems in a different time scale than physical time. (iii) Testing systems by simulations before they are manufactured, for example ships, airplanes, missiles, and sub-sea oil installations. (iv) Model-based control, such as LQG control, model-based predictive control (MPC), linear and nonlinear decouplers, Smith predictors, etc.

One of the most commonly used approaches for modeling dynamic systems is to develop models from equations of science. This is referred to as mechanistic modeling or first principle modeling. Another approach that may be based on laws of science and / or knowledge of the systems to be modeled is linguistic modeling (fuzzy modeling). Empirical modeling is another commonly used approach for developing dynamic as well as static models: Models are developed directly from observations of the systems. This is also referred to as black-box modeling. Empirical modeling used to build *dynamic* models is referred to as system identification. System identification is commonly used for developing models for model-based control. System identification may be used for systems that are too complex to be modeled by mechanistic modeling and where parameters in the mechanistic models are unknown.

A general introduction to system identification is given in [2]. Both linear and nonlinear system identification are considered. Also practical issues are discussed,

such as experiment design and data preprocessing. The DSR sub-space system identification algorithm is presented in [3]. In [1, 4] the DSR E sub-space system identification algorithm for use in closed loops is presented. DSR and DSR E have been developed by David Di Ruscio. Simulations comparing DSR E to other system identification algorithms, including N4SID, DSR, and the MATLAB implementation `pem` (Prediction Error Method), are presented in [1].

The main contribution of this paper is to show that the DSR E algorithm can be approximated by a two-step ARX algorithm. This approximation will in this paper be referred to as DARX.

The mathematical derivation of the DSR E algorithm, and inherently also the DARX algorithm, requires $N \rightarrow \infty$ and $J \rightarrow \infty$. Here N is the number of samples and J is a parameter to the DSR E and DARX algorithms. The J parameter is the order of the model to be identified in the first step of the DARX algorithm. These requirements can not be met in any practical system identification problems. A study of how finite values of N and J influence the first step of DSR E and DARX is presented in this paper.

2 Notation and Definitions

The inputs to a system are collected in the input column vector, $u \in \mathbb{R}^{r\times 1}$, where r is the number of inputs. The outputs from a system are collected in the output column vector, $y \in \mathbb{R}^{m\times 1}$, where m is the number of outputs. A sub-script to these vectors, for example y_k, refers to the sampling number.

The limit notation of Eq. (1) is simplified as shown in Eq. (2).

$$\lim_{x\rightarrow z, y\rightarrow z} f(x,y) \tag{1}$$

$$\lim_{x,y\rightarrow z} f(x,y) \tag{2}$$

Def. 1 (Innovation Process)**.** The system output, y_k, may be decomposed into two components: (i) The component of y_k that can be predicted from previous inputs, $u_{-\infty}, \ldots, u_{k-1}$, and previous outputs, $y_{-\infty}, \ldots, y_{k-1}$, assuming no model errors. For bi-proper systems, i.e. systems having direct feed-through from the input, u_k, to the output, y_k, the current input, u_k, is also included in the prediction of y_k. This predictable component of y_k is referred to as $\bar{y}_k$. (ii) The complement of $\bar{y}_k$, i.e. the component of y_k that can *not* be predicted from previous inputs and previous outputs. This is referred to as the innovation process, ε_k. Hence, $\varepsilon_k = y_k - \bar{y}_k$.

The symbol ε is used for the true innovation process, which in general is unknown. The symbol ϵ is used for the identified innovation process. The identified innovation process is in general not exactly identical to the true innovation process.

Def. 2 (State Space Model Form)**.** The discrete state space model form used by the DSR E algorithm is as shown in Eq. (3) and Eq. (4) [5].

$$\begin{aligned} \bar{x}_{k+1} &= \tilde{A}\bar{x}_k + \tilde{B}u_k + \tilde{C}e_k & (3)\\ y_k &= \tilde{D}\bar{x}_k + \tilde{E}u_k + \tilde{F}e_k & (4) \end{aligned}$$

In Eq. (3) and Eq. (4) the tilde symbol is used to avoid confusion with the polynomials of ARMAX and ARX models. $\bar{x} \in \mathbb{R}^{n\times 1}$ is the estimate of the system state vector, x, where n is the number of system states. $e \in \mathbb{R}^{m\times 1}$ is white noise with covariance matrix $E(e_k e_k^T) = I_m$. For invertible $\tilde{F}$, the system can be written on innovation form as presented in Eq. (5) and Eq. (6) [5].

$$\begin{aligned} \bar{x}_{k+1} &= \tilde{A}\bar{x}_k + \tilde{B}u_k + \tilde{K}\varepsilon_k & (5)\\ y_k &= \tilde{D}\bar{x}_k + \tilde{E}u_k + \varepsilon_k & (6) \end{aligned}$$

In Eq. (6), $\varepsilon_k = \tilde{F}e_k$ is the innovation process and $\tilde{K} = \tilde{C}\tilde{F}^{-1}$ is the Kalman filter gain matrix. In this paper, only strictly proper systems will be considered, i.e. $\tilde{E} = 0_{m\times r}$.

Def. 3 (ARMAX Model Form)**.** Eq. (7) defines the general form of ARMAX models [2].

$$A(q)y_k = B(q)u_k + C(q)\varepsilon_k \tag{7}$$

In Eq. (7), q is the time-shift operator of the Z-transform, i.e. $q^{-1}\, y_k = y_{k-1}$. Symbol q is commonly used within the subject of system identification. Symbol z is used in many other contexts. $A(q)$, $B(q)$, and $C(q)$ are polynomials. n_A, n_B, and n_C are the number of coefficients in these polynomials that in general are different from 1. The $A(q)$ and $C(q)$ polynomials are monic polynomials, i.e. the coefficient of their highest order term is 1.

Def. 4 (ARX Model Form)**.** Eq. (8) defines the general form of ARX models [2].

$$A(q)y_k = B(q)u_k + \varepsilon_k \tag{8}$$

The $A(q)$ polynomial is monic.

Def. 5 (Orthogonal Projection)**.** The orthogonal projection of matrix G onto matrix H, G/H, is defined as in Eq. (9) [5].

$$G/H \stackrel{\text{def}}{=} GH^T(HH^T)^\dagger H \tag{9}$$

In Eq. (9), the super-script $\dagger$ refers to the Moore-Penrose pseudo-inverse.

Def. 6 (Complement of Orthogonal Projection)**.** The complement of the orthogonal projection of matrix G onto matrix H, $GH^\perp$, is defined as in Eq. (10) [5].

$$GH^\perp \stackrel{\text{def}}{=} G - G/H \stackrel{\text{def}}{=} G - GH^T(HH^T)^\dagger H \tag{10}$$

Def. 7 (Hankel Matrix)**.** Let $s_t \in \mathbb{R}^{n_r \times n_c}$ be a matrix of data sampled at timestep t. The Hankel matrix $S_{t_0|L}$, organizing timeseries of s_t starting at timestep t_0, i.e. $s_{t_0}, s_{t_0+1}, \ldots$, is defined as in Eq. (11).

$$S_{t_0|L} \stackrel{\text{def}}{=} \begin{bmatrix} s_{t_0} & s_{t_0+1} & \cdots & s_{t_0+K-1} \\ s_{t_0+1} & s_{t_0+2} & \cdots & s_{t_0+K} \\ \vdots & \vdots & \ddots & \vdots \\ s_{t_0+L-1} & s_{t_0+L} & \cdots & s_{t_0+L+K-2} \end{bmatrix} \in \mathbb{R}^{Ln_r \times Kn_c} \tag{11}$$

In Eq. (11), L is the number of block rows in $S_{t_0|L}$ and K is the number of block columns in $S_{t_0|L}$ [5].

Def. 8 (Lower Block Triangular Toeplitz Matrix for the Quadruple $(\tilde{D}, \tilde{A}, \tilde{C}, \tilde{F})$)**.** The lower block triangular Toeplitz matrix for the quadruple $(\tilde{D}, \tilde{A}, \tilde{C}, \tilde{F})$ is defined as in Eq. (12) [5].

$$H_L^s \stackrel{\text{def}}{=} \begin{bmatrix} \tilde{F} & 0_{m\times m} & 0_{m\times m} & \cdots & 0_{m\times m} \\ \tilde{D}\tilde{C} & \tilde{F} & 0_{m\times m} & \cdots & 0_{m\times m} \\ \tilde{D}\tilde{A}\tilde{C} & \tilde{D}\tilde{C} & \tilde{F} & \cdots & 0_{m\times m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \tilde{D}\tilde{A}^{L-2}\tilde{C} & \tilde{D}\tilde{A}^{L-3}\tilde{C} & \tilde{D}\tilde{A}^{L-4}\tilde{C} & \cdots & \tilde{F} \end{bmatrix} \in \mathbb{R}^{Lm\times Lm} \tag{12}$$

In Eq. (12), L is the number of block rows and block columns in H_L^s.

3 The DSR E Algorithm

This section gives a brief derivation of the DSR E subspace system identification algorithm. A comprehensive presentation of DSR E is provided by [1]. DSR E is also presented in [4].

In [5] it is proved that for linear time-invariant (LTI) systems the innovation process, ε_k, can be identified directly from previous inputs, $u_{-\infty}, \ldots, u_{k-1}$, and previous outputs, $y_{-\infty}, \ldots, y_{k-1}$, without relying on models. The DSR E algorithm presented in [1, 4] is based on this proof. A MATLAB implementation of the DSR E algorithm is available in the DSR Toolbox for MATLAB ([6]). The DSR E algorithm consists of two steps:

1. The innovation process, ε_k, is identified by orthogonal projection of the current output, y_k, onto inputs and outputs from the J preceding samples, i.e. $u_{k-J}, \ldots, u_{k-1}$ and $y_{k-J}, \ldots, y_{k-1}$. Here J is a parameter to the DSR E algorithm. The complement of this orthogonal projection is the identified innovation process, ϵ_k [1, 4]. Please refer to Subsec. 3.1 for details.
2. The identified innovation process, ϵ_k, is considered as a known deterministic input. Hence, the deterministic / stochastic system identification problem is reduced to a deterministic system identification problem [1, 4]. In the DSR E algorithm presented in [1], this deterministic problem is solved by a deterministic sub-space system identification algorithm.

The the following parameters are most important with respect to the DSR E algorithm and its derivation [5, 1, 4]:

1. L - the number of block rows in the Toeplitz matrices and some of the Hankel matrices to be used in the second step of the DSR E algorithm, i.e. L has the same meaning as in Eq. (11) and Eq. (12).
2. g - if the system is strictly proper, i.e. $E = 0_{m\times r}$, then g is set to 0. Otherwise g is set to 1.
3. J - the number of preceding inputs and outputs used to identify the innovation process. Please refer to Subsec. 3.1 for details.

In addition to parameters L, g, and J, the model order, n, of the model to be identified by DSR E has to be specified to the DSR E implementation of the DSR Toolbox for MATLAB ([6]).

3.1 Step 1 of the DSR E Algorithm

Let $U_{0|J}$, $U_{J|L+g-1}$, $Y_{0|J}$, $Y_{J|L}$, and $E_{J|L}$ be Hankel matrices according to Def. 7. The input vector $u \in \mathbb{R}^{r\times 1}$ is the block elements of the matrices $U_{0|J}$ and $U_{J|L+g-1}$. The output vector $y \in \mathbb{R}^{m\times 1}$ is the block elements of the matrices $Y_{0|J}$ and $Y_{J|L}$. The white noise vector $e \in \mathbb{R}^{m\times 1}$ is the block elements of matrix $E_{J|L}$. In [5], Eq. (13) is proved. This proof will not be repeated in here.

$$\begin{aligned} \lim_{J,K\to\infty} Y_{J|L} - Y_{J|L} / \begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} \\ = \lim_{J,K\to\infty} H_L^s E_{J|L} \end{aligned} \tag{13}$$

Please note that the mathematical derivation of Eq. (13) requires that $J \to \infty$ and $K \to \infty$, where K is the number of columns in the Hankel matrices of Eq. (13). In Sec. 7 the consequences of finite values of J and

K are considered. Choosing $L = 1$ and $g = 0$ gives Eq. (14) [1, 4].

$$\begin{bmatrix} U_{J|L+g-1} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \begin{bmatrix} U_{J|0} \\ U_{0|J} \\ Y_{0|J} \end{bmatrix} = \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} \quad (14)$$

The equality between the middle and the rightmost terms of Eq. (14) is due to $U_{J|0} \in \mathbb{R}^{0\times K}$. For $L = 1$ the Toeplitz matrix of Def. 8 reduces to $H_1^s \in \mathbb{R}^{m\times m} \Rightarrow H_1^s = \tilde{F}$. Hence, Eq. (13) can be written as Eq. (15) [1, 4].

$$\begin{aligned} \lim_{J,K\to\infty} Y_{J|1} - Y_{J|1}/\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} &= \lim_{J,K\to\infty} \tilde{F}E_{J|1} \quad (15)\\ &= \lim_{J,K\to\infty} \tilde{F}\begin{bmatrix} e_J & e_{J+1} & \dots & e_{J+K-1} \end{bmatrix} \\ &= \lim_{J,K\to\infty} \begin{bmatrix} \varepsilon_J & \varepsilon_{J+1} & \dots & \varepsilon_{J+K-1} \end{bmatrix} \\ &= \lim_{J,K\to\infty} \varepsilon_{J|1} \end{aligned}$$

In Eq. (15), $\varepsilon_{J|1}$ is the Hankel matrix of which block elements are the innovation process $\varepsilon \in \mathbb{R}^{m\times 1}$.

3.2 Step 2 of the DSR E Algorithm

As the innovation process is identified in Eq. (15), the deterministic / stochastic system identification problem of Eq. (5) and Eq. (6) reduces to a deterministic problem. For strictly proper systems, i.e. $\tilde{E} = 0_{m\times r}$, the system identification problem is now on the form of Eq. (16) and Eq. (17) [1, 4].

$$x_{k+1} = \tilde{A}x_k + \begin{bmatrix} \tilde{B} & \tilde{K} \end{bmatrix}\begin{bmatrix} u_k \\ \epsilon_k \end{bmatrix} \quad (16)$$

$$y_k - \epsilon_k = \tilde{D}x_k \quad (17)$$

Step 2 of the DSR E algorithm as presented in [1] is to solve the deterministic system identification problem of Eq. (16) and Eq. (17) using a deterministic sub-space system identification algorithm.

4 Relating Orthogonal Projection to the Ordinary Least Squares Method

The key to understand that the first step of the DSR E algorithm is mathematically identical to the single-step ARX algorithm is the relation between orthogonal projection (Def. 5) and the ordinary least squares (OLS) method. Consider the linear regression problem of Eq. (18).

$$Y = XB + E \quad (18)$$

In Eq. (18), the elements of $X \in \mathbb{R}^{N\times b}$ and $Y \in \mathbb{R}^{N\times a}$ are known data. The regression matrix $B \in \mathbb{R}^{b\times a}$ is to be identified. $E \in \mathbb{R}^{N\times a}$ is the residual of the linear regression. Assume that $N \geq b$ and that $\mathrm{rank}(X) = b$. Then $\mathrm{rank}(X^TX) = b$, which is full rank. For a quadratic matrix of full rank, i.e. an invertible matrix, the Moore-Penrose pseudo inverse is equivalent to the inverse, i.e. $(X^TX)^\dagger = (X^TX)^{-1}$. Transposing Eq. (18) gives Eq. (19).

$$Y^T = B^TX^T + E^T \quad (19)$$

Solving the linear regression problem of Eq. (19) using OLS gives Eq. (20).

$$B^T = Y^TX(X^TX)^{-1} \quad (20)$$

Inserting Eq. (20) into Eq. (19) gives Eq. (21).

$$Y^T = \underbrace{Y^TX(X^TX)^{-1}}_{B}X^T + E^T \quad (21)$$

Because $(X^TX)^\dagger = (X^TX)^{-1}$, the first term on the right hand side of Eq. (21) is identical to the right hand side of Eq. (9), where $Y^T = G$ and $X^T = H$. Hence, according to Def. 5, Eq. (21) can be written as Eq. (22).

$$Y^T = Y^T/X^T + E^T \quad (22)$$

Further, using Def. 6 gives Eq. (23).

$$Y^TX^{T\perp} = Y^T - Y^T/X^T = E^T \quad (23)$$

Conclusions: (i) The orthogonal projection Y^T/X^T is equivalent to the part of Y that can be explained by an OLS regression of Y onto X. (ii) The complement of the orthogonal projection, $Y^TX^{T\perp}$, is equivalent to the residual of this OLS regression, E.

5 The ARX Algorithm

The term ARX may refer to (i) the model form of Eq. (8) or (ii) a system identification algorithm used to identify models on the form of Eq. (8). This section gives a brief introduction to the ARX algorithm.

For single input, single output (SISO), strictly proper systems, Eq. (8) can be written on the form of Eq. (24). It has here been used that $q^{-1}y_k = y_{k-1}$, where q is the time shift operator of the Z transform. Eq. (24) can be rewritten as Eq. (25).

$$y_k + a_1 y_{k-1} + a_2 y_{k-2} + \ldots + a_{n_A} y_{k-n_A} = b_1 u_{k-1} + b_2 u_{k-2} + \ldots + b_{n_B} u_{k-n_B} + \varepsilon_k \tag{24}$$

$$y_k = -a_1 y_{k-1} - a_2 y_{k-2} - \ldots - a_{n_A} y_{k-n_A} + b_1 u_{k-1} + b_2 u_{k-2} + \ldots + b_{n_B} u_{k-n_B} + \varepsilon_k \tag{25}$$

The polynomial coefficients, $a_1, a_2, \ldots, a_{n_A}$ and $b_1, b_2, \ldots, b_{n_B}$, can be estimated by stacking timeseries of y and u in the Y and X matrices of Eq. (18) as shown in Eq. (26), and then solve this OLS problem with respect to the ARX parameter vector, θ, which corresponds to the regression matrix, B, of Eq. (18).

$$\underbrace{\begin{bmatrix} \vdots \\ y_k \\ y_{k+1} \\ y_{k+2} \\ \vdots \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -y_{k-1} & -y_{k-2} & \cdots & -y_{k-n_A} & u_{k-1} & u_{k-2} & \cdots & u_{k-n_B} \\ -y_k & -y_{k-1} & \cdots & -y_{k-n_A+1} & u_k & u_{k-1} & \cdots & u_{k-n_B+1} \\ -y_{k+1} & -y_k & \cdots & -y_{k-n_A+2} & u_{k+1} & u_k & \cdots & u_{k-n_B+2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n_A} \\ b_1 \\ b_2 \\ \vdots \\ b_{n_B} \end{bmatrix}}_{\theta} + \underbrace{\begin{bmatrix} \vdots \\ \varepsilon_k \\ \varepsilon_{k+1} \\ \varepsilon_{k+2} \\ \vdots \end{bmatrix}}_{E} \tag{26}$$

In Eq. (26), $Y \in \mathbb{R}^{P\times 1}$, $X \in \mathbb{R}^{P\times(n_A+n_B)}$, $\theta \in \mathbb{R}^{(n_A+n_B)\times 1}$, and $E \in \mathbb{R}^{P\times 1}$, where P is the number of rows in Y, X, and E.

The ARX system identification problem of Eq. (26) can be generalized to multiple input, multiple output (MIMO) systems as shown in Eq. (27).

$$\underbrace{\begin{bmatrix} \vdots \\ y_k^T \\ y_{k+1}^T \\ y_{k+2}^T \\ \vdots \end{bmatrix}}_{Y} \tag{27}$$

$$= \underbrace{\begin{bmatrix} \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -y_{k-1}^T & -y_{k-2}^T & \cdots & -y_{k-n_A}^T & u_{k-1}^T & u_{k-2}^T & \cdots & u_{k-n_B}^T \\ -y_k^T & -y_{k-1}^T & \cdots & -y_{k-n_A+1}^T & u_k^T & u_{k-1}^T & \cdots & u_{k-n_B+1}^T \\ -y_{k+1}^T & -y_k^T & \cdots & -y_{k-n_A+2}^T & u_{k+1}^T & u_k^T & \cdots & u_{k-n_B+2}^T \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}}_{X} \Theta + \underbrace{\begin{bmatrix} \vdots \\ \varepsilon_k^T \\ \varepsilon_{k+1}^T \\ \varepsilon_{k+2}^T \\ \vdots \end{bmatrix}}_{E}$$

In Eq. (27), $Y \in \mathbb{R}^{P\times m}$, $X \in \mathbb{R}^{P\times(mn_A+rn_B)}$, $\Theta \in \mathbb{R}^{(mn_A+rn_B)\times m}$, and $E \in \mathbb{R}^{P\times m}$, where P is the number of rows in Y, X, and E. The notation Θ is used for the ARX parameter matrix, which is to be identified by the OLS method.

Solving the linear regression problem of Eq. (26) or Eq. (27) with respect to θ or Θ, respectively, is the system identification algorithm referred to as single-step ARX. There also exist various multi-step ARX algorithms. These algorithms involve solving several single-step ARX problems. A two-step ARX identification algorithm is derived in Sec. 6.

6 Approximating the DSR E Algorithm Using a Two-Step ARX Algorithm

This section shows that the DSR E sub-space system identification algorithm can be approximated by a two-step ARX algorithm.

6.1 Approximating Step 1 of the DSR E Algorithm

Consider Eq. (15), using the final right hand side term: As the requirements $J \to \infty$ and $K \to \infty$ can not be met in any practical system identification problems, the limit notation is removed and the true innovation process, ε, is replaced by the identified (in general not exact) innovation process, ϵ. Further, Eq. (15) is rearranged by: (i) The last term on the left hand side is rewritten using Def. 5 and moved to the right hand side. (ii) The equation is transposed. Eq. (15) is then written as Eq. (28). The underbraces of Eq. (28) is on the form of Eq. (18), where the regression matrix, B, is replaced by the ARX parameter matrix, Θ. The structures of the regression matrices of Eq. (28) are shown in Eq. (29) and Eq. (30).

$$\underbrace{Y_{J|1}^T}_{Y} = \underbrace{\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^T}_{X} \underbrace{\left(\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^T \right)^{-1} \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix} Y_{J|1}^T}_{\Theta} + \underbrace{\epsilon_{J|1}^T}_{E} \tag{28}$$

$$Y_{J|1}^T = \begin{bmatrix} y_J^T \\ y_{J+1}^T \\ \vdots \\ y_{J+K-1}^T \end{bmatrix} \in \mathbb{R}^{K\times m} \tag{29}$$

$$\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^T = \begin{bmatrix} u_0^T, & u_1^T, & \ldots, & u_{J-1}^T, & y_0^T, & y_1^T, & \ldots, & y_{J-1}^T \\ u_1^T, & u_2^T, & \ldots, & u_J^T, & y_1^T, & y_2^T, & \ldots, & y_J^T \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ u_{K-1}^T, & u_K^T, & \ldots, & u_{K+J-2}^T, & y_{K-1}^T, & y_K^T, & \ldots, & y_{K+J-2}^T \end{bmatrix} \in \mathbb{R}^{K\times(r+m)J} \tag{30}$$

The right hand sides of Eq. (29) and Eq. (30) are recognized as the matrices for linear regression of a strictly proper ARX model where $n_A = n_B = J$: Choosing $n_A = n_B = J$ in Eq. (27) gives that Y and X as underbraced in Eq. (27) are identical to Eq. (29) and Eq. (30) respectively. Comparing X as underbraced in Eq. (27) to Eq. (30) shows that: (i) The arrangement (order) of the block columns is different and (ii) the signs (plus or minus) of the y block elements are different. However, this will not affect the orthogonal projection as it does not affect the information available in each row of the matrices. Hence, Eq. (28) is identical to a strictly proper ARX model written on OLS regression form. This regression can be written as Eq. (31).

$$Y_{J|1}^T = \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^T \Theta + \epsilon_{J|1}^T \tag{31}$$

The main point of this derivation is obtained by rewriting Eq. (31) as Eq. (32).

$$\epsilon_{J|1}^T = Y_{J|1}^T - \begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}^T \Theta \tag{32}$$

Eq. (32) proves that the innovation process identified by the DSR E method, ϵ, is mathematically identical to the residual of a strictly proper ARX model where $n_A = n_B = J$. This residual can also be expressed as the one-step-ahead prediction errors in a simulation running the ARX model on its own training dataset. It has now been proved that the first step of the DSR E algorithm can be replaced by the single-step ARX algorithm.

6.2 Approximating Step 2 of the DSR E Algorithm

Similar to the DSR E algorithm, the DARX algorithm considers the identified innovation process, ϵ, as a known deterministic input. The deterministic / stochastic system identification problem has then been reduced to a deterministic system identification problem. This deterministic system can be written on the ARMAX form, Eq. (7), replacing the true (but unknown) innovation process, ε, by the identified innovation process, ϵ, which in general is not exactly identical to the true innovation process. The model is then on the form of Eq. (33). Assume for simplicity that a single input, single output (SISO) system is to be modeled using a strictly proper ARMAX model of which $n_A = n_B = n_C = n$. Hence, the ARMAX polynomials are given by Eq. (34) to Eq. (36).

$$A(q)y_k = B(q)u_k + C(q)\epsilon_k \tag{33}$$

$$A(q) = 1 + a_1q^{-1} + \ldots + a_nq^{-n} \tag{34}$$

$$B(q) = b_1q^{-1} + \ldots + b_nq^{-n} \tag{35}$$

$$C(q) = 1 + c_1q^{-1} + \ldots + c_nq^{-n} \tag{36}$$

Inserting Eq. (34) to Eq. (36) into Eq. (33) and using that $q^{-1}y_k = y_{k-1}$ gives Eq. (37). Eq. (37) can be rewritten as Eq. (38). Writing Eq. (38) on linear regression form gives Eq. (39). The number of equations in this linear regression problem is $K - n$, where K is the number of rows in Eq. (29) and Eq. (30).

$$y_k + a_1y_{k-1} + \ldots + a_ny_{k-n} = b_1u_{k-1} + \ldots + b_nu_{k-n} + \epsilon_k + c_1\epsilon_{k-1} + \ldots + c_n\epsilon_{k-n} \tag{37}$$

$$y_k - \epsilon_k = -a_1y_{k-1} - \ldots - a_ny_{k-n} + b_1u_{k-1} + \ldots + b_nu_{k-n} + c_1\epsilon_{k-1} + \ldots + c_n\epsilon_{k-n} \tag{38}$$

$$\begin{bmatrix} y_{J+n} & - & \epsilon_{J+n} \\ y_{J+n+1} & - & \epsilon_{J+n+1} \\ & \vdots & \\ y_{K+J-1} & - & \epsilon_{K+J-1} \end{bmatrix} \tag{39}$$

$$= \begin{bmatrix} -y_{J+n-1}, & \dots, & -y_J, & u_{J+n-1}, & \dots, & u_J, & \epsilon_{J+n-1}, & \dots, & \epsilon_J \\ -y_{J+n}, & \dots, & -y_{J+1}, & u_{J+n}, & \dots, & u_{J+1}, & \epsilon_{J+n}, & \dots, & \epsilon_{J+1} \\ \vdots & \ddots, & \vdots & \vdots & \ddots, & \vdots & \vdots & \ddots, & \vdots \\ -y_{K+J-2}, & \dots, & -y_{K+J-n-1}, & u_{K+J-2}, & \dots, & u_{K+J-n-1}, & \epsilon_{K+J-2}, & \dots, & \epsilon_{K+J-n-1} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_n \\ c_1 \\ \vdots \\ c_n \end{bmatrix}$$

The ARX regression problem of Eq. (39) identifies not only the $A(q)$ and $B(q)$ polynomials, but also the $C(q)$ polynomial. Hence, the identified model is an ARMAX model on the form of Eq. (33). An ARMAX model can be converted to a state space model on the form of Eq. (5) and Eq. (6) and vice versa. Hence, the ARMAX model identified by solving the ARX regression problem of Eq. (39) can be converted to a state space model as generated by the DSR E algorithm presented in [1].

It has now been shown that: (i) The innovation process identified by the first step of the DSR E algorithm, ϵ, is mathematically identical to the residual of an ARX identification where $n_A = n_B = J$. (ii) The second step of the DSR E algorithm, as presented in [1], can be replaced by ARX identification. From (i) and (ii) it is concluded that the DSR E algorithm can be approximated by a two-step ARX algorithm. The model identified by this approximation is an ARMAX model.

7 The Influence of Number of Samples, N, and the Parameter J

Consider Eq. (29) and Eq. (30): The lowest sample index used is 0 (in u_0 and y_0) and the highest sample index used is $J+K-1$ (in y_{J+K-1}). Hence, the number of samples, N, used to identify the innovation process is given by $N = K + J$.

As the mathematical derivation from [5] requires that $J \to \infty$ and $K \to \infty$, it is also implicitly requires that $N \to \infty$. These requirements can not be met in any practical system identification problems. In practical problems the number of samples, N, is given by the dataset. It is then a consideration to choose a proper value of J. K is then given by $K = N - J$.

Consider a single input, single output (SISO) system. Then the OLS regression problem of Eq. (28) has $2J+K$ unknown values: (i) A SISO ARX model where $n_A = n_B = J$ has $2J$ unknown parameters, i.e. J coefficients in the $A(q)$ polynomial and J coefficients in the $B(q)$ polynomial. (ii) Unknown innovation processes for K samples. The number of equations in the OLS problem is K. By choosing J too large, the identified innovation process, ϵ, will be smaller (absolute value) than the true innovation process, ε. This can be illustrated by an extreme choice of J: Choosing $J = N/3$. Then the ARX model will have $2J = 2N/3$ parameters. The number of equations in the OLS regression problem will be $K = N - J = N - N/3 = 2N/3$. Assuming that these $2N/3$ equations are linearly independent, the number of parameters to be identified in the OLS regression is equal to the number of linearly independent equations. Hence, the OLS problem is reduced to a deterministic set of linear equations. Then the residual, i.e. the identified innovation process, ϵ, will be zero regardless of the true innovation process, ε. On the other hand, choosing J too low will also conflict the derivation of [5].

In order to quantify the fit of the identified innovation process, ϵ, to the true innovation process, ε, the fit criterion $W(N, J)$ of Eq. (40) has been defined.

$$W(N, J) \stackrel{\text{def}}{=} \frac{1}{N-J} \sum_{k=J+1}^{N} (\varepsilon_k - \epsilon_k(N, J))^2 \tag{40}$$

Two datasets were generated by simulations using a SISO ARMAX model where $n_A = n_B = n_C = 6$. During the simulations, a pseudo random binary signal (PRBS) was applied to the deterministic input, u. A uniformly distributed random number sequence was applied for simulating of the innovation process, ε. Different amplitude of the innovation process, ε, was applied for dataset no. 2 compared to dataset no. 1. Otherwise the datasets were generated using identical conditions.

For both datasets, $W(N, J)$ was plotted as function of J for $N = 350$, $N = 1,000$, $N = 3,000$, and $N = 10,000$, i.e. total eight J versus $W(N, J)$ plots. These plots are shown in Fig. 1 and Fig. 2. In Fig. 1 the innovation process, ε, is uniformly distributed in the interval $[-0.05, 0.05]$. In Fig. 2 the innovation process is

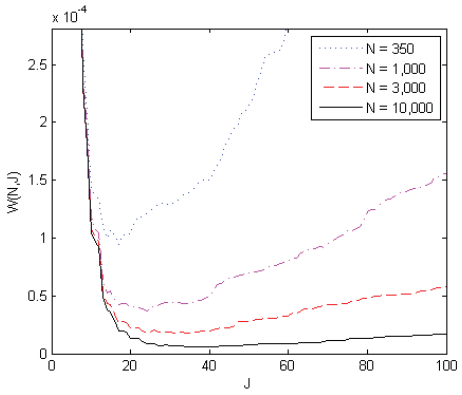


Fig. 1 The figure shows $W(N, J)$ plotted as function of J for $N = 350$, $N = 1,000$, $N = 3,000$ and $N = 10,000$. The innovation process, ε, is uniformly distributed in the interval $[-0.05, 0.05]$.

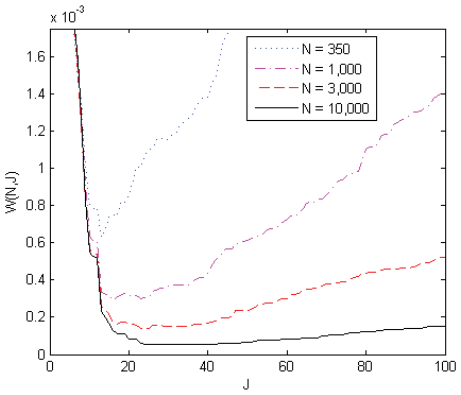


Fig. 2 The figure shows $W(N, J)$ plotted as function of J for $N = 350$, $N = 1,000$, $N = 3,000$ and $N = 10,000$. The innovation process, ε, is uniformly distributed in the interval $[-0.15, 0.15]$.

uniformly distributed in the interval $[-0.15, 0.15]$. In other words: In Fig. 2 the amplitude of the innovation process is three times higher than in Fig. 1. Please note that the figures have different scaling of their respective Y-axes.

Based on the plots shown in Fig. 1 and Fig. 2, it seems reasonable to draw the following conclusions:

1. Increased value of N gives better match between ε and ϵ for the optimal choice of J, i.e. the value of J giving the lowest $W(N, J)$. This is to be expected because the derivation from [5] assumes $N \to \infty$.
2. When ignoring some high frequency variations ("noise") on the curves, it seems (but can not be stated) that: (i) There is exact one minimum on each curve. (ii) The curves are strictly increasing as moving away from these minimums. This seems reasonable according to the discussion for choice of J above.
3. As N increases the optimal choice of J, i.e. the value of J giving the minimum of $W(N, J)$, also increases. This is reasonable: As N increases, the number of equations in the OLS problem of Eq. (28) also increases. Hence, the number of coefficients in the ARX model, $2J$, may increase without over-fitting the model.
4. As the amplitude of the innovation process, ε, increases, the fit criterion, $W(N, J)$, also increases. This is reasonable: When ε and ϵ in general have larger values, also the difference between these values will be larger.

8 Comparing DSR E and DARX on Industrial Data

The DSR E and DARX algorithms have been compared on experimental data from the copper refining process of Xstrata Nikkelverk, Kristiansand, Norway. A single input, single output (SISO) system was modeled using DSR E and DARX. The input, u, is the mass flow from the roasting furnace to the copper leaching process. The output, y, is the concentration of sulphuric acid, H_2SO_4, in the flow from the copper leaching process to the electro winning. Before system identification, the input and output dataseries were preprocessed by (i) removing outliers, (ii) subtracting mean value, (iii) dividing by standard deviation, and (iv) compensating for the time delay form the input, u, to the output, y, by shifting the input dataseries with respect to the output dataseries. For DSR E, the parameters were chosen as $L = n = 20$, $g = 0$, and $J = 30$. For DARX, the parameters were chosen as $J = 30$ and $n_A = n_B = n_C = 20$. The DSR E implementation of the DSR Toolbox for MATLAB ([6]) was used for identifying the DSR E model. For identifying the DARX model, a MATLAB implementation of the DARX algorithm was written.

Based on the models identified by DSR E and DARX, ballistic simulations were run. Fig. 3 compares these simulations to the actual measured sulphuric acid concentration. The ballistic simulations fit the measured data poorly. This is to be expected because there are several other factors influencing the concentration of sulphuric acid that are not included in the models. By comparing the identified models to the model properties expected based on process knowledge, it has been verified that the modeled transfer functions have correct signs (in this case minus) and correct stability properties (in this case integrating).

The main point of this experiment is to show that DSR E and DARX give identical models: Fig. 3 shows that the model responses are identical when applying the mass flow signal, u, to the model inputs. Even though the values of the ballistic simulations vary within approximately ± 1.3, the maximum difference between the

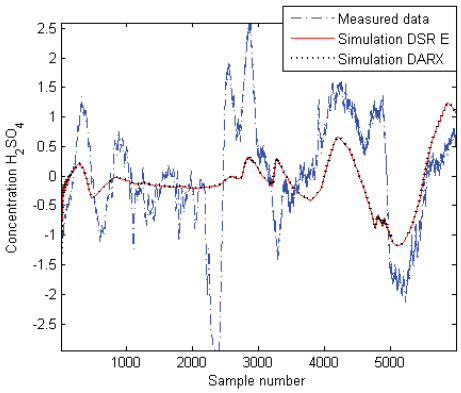


Fig. 3 The figure compares ballistic simulations using the models identified by DSR E and DARX. The actual measured sulphuric acid concentration, after being preprocessed, is also shown.

DSR E based simulation and the DARX based simulation is only 3.3×10^{-11}. This is a very strong indication that the models are identical.

To verify that the models actually are identical, the pole / zero plots and the step responses were also compared. Both the deterministic models, i.e. the transfer functions from u to y, and the noise models, i.e. the transfer functions from ε to y, were compared by pole / zero plots and step response plots. These plots are *not* shown in this paper. To the resolution of the plots, it was not possible to distinguish the models, neither by the pole / zero plots nor by the step response plots. It is then concluded that the models are identical beyond the model accuracy that can be expected from such modeling techniques.

9 Conclusions

The DSR E sub-space system identification algorithm can be approximated by a two-step ARX algorithm. The first step of the DSR E algorithm is mathematically identical to the identification of a strictly proper ARX model of which $n_A = n_B = J$: The innovation process identified by DSR E is identical to the residual of the ARX identification. During the second step of the DSR E algorithm the deterministic / stochastic system identification problem is reduced to a deterministic problem by considering the identified innovation process as a known deterministic input. In the DSR E algorithm, as presented in [1], this deterministic problem is solved by a deterministic sub-space system identification algorithm. However, this problem may instead be solved using the single-step ARX algorithm. Hence, each of the two steps of the DSR E algorithm may be replaced by the single-step ARX algorithm. This allows DSR E to be approximated by a two-step ARX algorithm.

The mathematical derivation of DSR E requires that $N \to \infty$ and $J \to \infty$, where N is the number of samples and J is as presented above. These requirements can not be met in any practical system identification problems. Simulations presented in this paper show that the innovation process is identified more accurately as N increase. The optimal choice of J, i.e. the value of J that gives the most accurate identification, increases as N increases.

The DSR E algorithm and its two-step ARX approximation were compared by modeling a section of the copper refining process at Xstrata Nikkelverk, Kristiansand, Norway. The model identified by the DSR E algorithm and the model identified by the approximation are identical beyond the model accuracy that can be expected from such modeling techniques.

10 Acknowledgments

The authors are most grateful to Xstrata Nikkelverk, Kristiansand, Norway for providing datasets from the copper refining process and for allowing these data to be used in this paper.

11 References

[1] GW. Nilsen. *Topics in Open and Closed Loop Subspace System Identification: Finite Data-Based Methods, Ph.D. Thesis*. Norwegian University of Science and Technology / Telemark University College, Trondheim / Porsgrunn, Norway, 2006.

[2] L. Ljung. *System Identification - Theory for the User, 2nd Edition*. Prentice-Hall, Upper Saddle River, New Jersey, 1999.

[3] D. Di Ruscio. Combined deterministic and stochastic system identification and realization: DSR - a subspace approach based on observations. *Modeling, Identification and Control*, 17, 1996.

[4] D. Di Ruscio. Subspace system identification of the Kalman filter: Open and closed loop systems. In *The 6th international conference on Computing, Communication and Control Technologies: CCCT 2008*, Orlando, Florida, June 29th - July 2rd 2008.

[5] D. Di Ruscio. Subspace system identification of the Kalman filter. *Modeling, Identification and Control*, 24:125–157, 2003.

[6] D. Di Ruscio. *DSR Toolbox for MATLAB*. Telemark University College, Porsgrunn, Norway, 2005.

Considerations of Paper B

The two-step ARX algorithm, referred to as DARX, does not estimate the model order. This is a disadvantage compared to the DSR E algorithm presented in Nilsen (2006) (reference [1] in the paper's reference list).

The DARX algorithm has been developed only for single input / single output (SISO) systems. This is also a disadvantage compared to DSR E. It is believed that the DARX algorithm can be extended to multiple input / multiple output (MIMO) systems. However, the candidate has not done any research on this topic. Unfortunately, the paper does not make it clear that the DARX algorithm has been developed only for SISO systems. On the contrary, some notation and verbal formulations *may* give the impression that the DARX algorithm has been developed also for MIMO systems. This is very unfortunate.

In Section 1 of the paper, it is written that system identification may be used for systems that are too complex for mechanistic (first principle) modeling. This is an imprecise formulation. It is meant that black-box modeling can be used when the process' input-output dynamics is relatively simple, even if the process' physics is very complex. This topic is discussed in Section 4.2 of this PhD thesis.

The paper uses the notation $\bar{y}$ for prediction of the process output y. As the notation $\hat{y}$ is common in systems and control engineering, it is unfortunate not to use this notation. The same issue applies for $\bar{x}$ versus $\hat{x}$. The symbol ε is used for the true, but unknown, innovation process, and the symbol ϵ is used for the identified innovation process. It would be better to use the notation $\hat{\varepsilon}$ for the identified innovation process, in order to consequently use hat-notation ($\hat{}$) for predicted and estimated values. It complicates the notation to use the symbol B for the parameter matrix in Equation (18), for later replacing it by the symbol Θ in (28). It would be better to use Θ both places.

In Section 6.1 of the paper, it is poorly explained *why* (32) proves that the innovation process identified by DSR E is identical to the residual of a strictly proper ARX model with $n_A = n_B = J$. Replacing Θ in (32) with the notation indicated by braces in (28), and expanding (15) using Def. 5, show that (15) and (32) are on the same form. In (15) the true innovation process, $\varepsilon_{J|1}$, depends on $J \to \infty$ and $K \to \infty$. These demands can not be meet as they require infinite number of data samples. The identified innovation process, $\epsilon_{J|1}$, is found from (15) as J and K are assigned finite values. That is, $\varepsilon_{J|1}$ is replaced by $\epsilon_{J|1}$ for finite J and K. As $\varepsilon_{J|1}$ is replaced by $\epsilon_{J|1}$, (15) and (32) become equal ((32) is written as the transposed of (15)). The main point is that $\varepsilon_{J|1}$ in (15) is the residual after projecting $Y_{J|1}$ onto the row space of $\begin{bmatrix} U_{0|J} \\ Y_{0|J} \end{bmatrix}$. In (32), $\epsilon_{J|1}$ is identified as the residual of the exact same projection (written as the transposed of (15)).

The candidate has used the term "timeseries" somewhat erroneously in this

paper compared to how the term is used in Ljung (1999). The paper uses the term for datasets logged from processes or made by dynamic simulations, while Ljung (1999) uses the term for "outputs of dynamical systems whose external stimuli are not observed".

Paper C

Solution to an Implementation Issue for a Two-Step ARX Algorithm, with Application to the Czochralski Crystallization Process

The candidate presented this paper at the 50th International Conference of Scandinavian Simulation Society (SIMS 2009). The paper is also included in the conference's proceedings. The conference was held October 7th-8th 2009 at DONG Energy, Fredericia, Denmark. The basic idea of this paper was first presented in the candidate's master thesis (Komperød (2008)). The paper was written and presented at the conference during the candidate's PhD study. The paper strongly depends on Paper B of this PhD thesis. After the paper, on page 85, a few considerations of the paper are discussed.

SOLUTION TO AN IMPLEMENTATION ISSUE FOR A TWO-STEP ARX ALGORITHM, WITH APPLICATION TO THE CZOCHRALSKI CRYSTALLIZATION PROCESS

Magnus Komperød
Østfold University College
Faculty of Engineering
1757 Halden
Norway

John Atle Bones
SINTEF Materials and Chemistry
Department of Metallurgy
7465 Trondheim
Norway

Bernt Lie *
Telemark University College
Faculty of Technology
3901 Porsgrunn
Norway

ABSTRACT

DSR E is a system identification algorithm for identification of linear time-invariant systems. DARX is a two-step ARX algorithm which approximates DSR E. The second step of DARX slightly differs from the standard ARX form. Apparently this prevents standard ARX software from being used, requiring a tailor-made implementation of ARX. This paper presents a rewriting of the second step of DARX. Applying this rewriting standard ARX software can be used also for the second step of DARX, simplifying the overall implementation of DARX. The rewriting is validated using timeseries logged at a real-life Czochralski crystallization process.

Keywords: Applied linear algebra, ARX, Czochralski crystallization process, DSR E, System identification.

NOMENCLATURE

$A(z)$	Polynomial in z^{-1}, $A(z) = 1 + \sum_{i=1}^{n_A} a_i z^{-i}$.
$B(z)$	Polynomial in z^{-1}, $B(z) = \sum_{i=1}^{n_B} b_i z^{-i}$.
$C(z)$	Polynomial in z^{-1}, $C(z) = 1 + \sum_{i=1}^{n_C} c_i z^{-i}$.
e	Actual ("true") innovation process.
f	Identified innovation process.
k	Timestep index.
P	Control signal for heating power [%].
T	Raw signal from temperature sensor [V].
u	Deterministic system input.
x_k	Value of x at timestep k, for $x \in \{e, f, u, y\}$.
y	Measured system output.
z	The timeshift operator of the Z-transform, $z^{-1}\, x_k \stackrel{\text{def}}{=} x_{k-1}$.

Text written in `teletype font` refers to MATLAB commands, for example `arx`.

*Corresponding author: Phone: +47 35 57 51 69 Fax: +47 35 57 52 50 E-mail: bernt.lie@hit.no

INTRODUCTION

Modeling of dynamic systems is a most important part of today's science and engineering. Empirical modeling is a commonly used modeling approach

which is based on observations of the systems to be modeled. Empirical modeling used for developing *dynamic* models is referred to as system identification. System identification is commonly used for developing models for model-based control, predictors, and state estimators.

ARX is an algorithm for system identification of linear, time-invariant (LTI) systems. ARX is a simple, commonly used algorithm, which is based on the ordinary least squares method (OLS). Although ARX identifies good models for many systems, it has significant disadvantages compared to other, more advanced methods. DSR E is a more advanced algorithm for system identification of LTI systems. The DSR E algorithm is a two-step algorithm [1, 4].

Reference [2] shows that the DSR E algorithm can be approximated by a two-step ARX algorithm. This approximation of the DSR E algorithm is referred to as the DARX algorithm. Unfortunately, the second ARX step of DARX is slightly different than the standard ARX form. This apparently requires an implementation of a tailor-made ARX algorithm, instead of using standard ARX software, such as the MATLAB command `arx`. Implementing a tailor-made ARX algorithm will increase the complexity of the otherwise simple implementation of DARX.

The main contribution of this paper is to present a rewriting of the second step of DARX. This rewriting allows standard ARX software to be used for both steps of DARX. Hence the implementation of DARX is significantly simplified. Only strictly proper, single input, single output (SISO) systems will be considered in this paper. Although not tested, it is believed that the results presented in this paper can be applied directly to biproper systems and multiple input systems as well. With respect to multiple output systems, the general challenges caused by a complex internal structure and a large number of parameters must be handled. Multiple output systems have not been considered by the authors.

The Czochralski (CZ) crystallization process is used to grow monocrystals (single crystals). Among the most important applications of the CZ process is production of monocrystalline silicon. Monocrystalline silicon is used for production of solar cells and in computers and electronics.

The rewriting of the second step of DARX is validated using timeseries logged at a real-life CZ process.

INNOVATION PROCESS

The actual ("true") innovation process, e_k, is the part of the measured system output, y_k, that can *not* be explained from previous system inputs, $u_{-\infty},\ldots,u_{k-1}$, and previous system outputs, $y_{-\infty},\ldots,y_{k-1}$, under the assumption of no model errors. For biproper systems, the current input, u_k, is also included in the explanation of y_k. The actual innovation process is typically caused by non-measured process disturbances and measurement noise.

The identified innovation process, f_k, is an estimate of the actual innovation process, e_k. The differences between the actual innovation process and the identified innovation process are typically caused by imperfect models and limited number of previous inputs and previous outputs being available.

A known input which has no measurement error is referred to as a deterministic input. A system having only deterministic inputs is referred to as a deterministic system. A system having both deterministic inputs and nonzero innovation process is referred to as a deterministic / stochastic system.

ARX AND ARMAX

The term ARX refers to a dynamic model on the form of Eq. 1.

$$A(z)\,y_k = B(z)\,u_k + f_k \tag{1}$$

The term ARX also refers to a system identification algorithm that identifies models on the form of Eq. 1. This algorithm is derived as follows: Using the definitions of $A(z)$, $B(z)$, and z from the nomenclature, Eq. 1 can be written as Eq. 2.

$$\begin{aligned} y_k \;=\; & -a_1\,y_{k-1} - \ldots - a_{n_A}\,y_{k-n_A} \\ & +b_1\,u_{k-1} + \ldots + b_{n_B}\,u_{k-n_B} + f_k \end{aligned} \tag{2}$$

Define the column vector $\mathbf{u}_k$ as in Eq. 3.

$$\mathbf{u}_k \stackrel{\text{def}}{=} \begin{bmatrix} u_k & u_{k+1} & \ldots \end{bmatrix}^T \tag{3}$$

Here k is the timestep index of the first element in the vector. The number of elements in the vector will be discussed shortly. The column vectors $\mathbf{f}_k$ and $\mathbf{y}_k$ are defined similarly for f and y, respectively. Define the regression matrix $\mathbf{R}_k$ and the parameter vector $\mathbf{p}$ as in Eq. 4 and Eq. 5.

$$\mathbf{R}_k \stackrel{\text{def}}{=} \begin{bmatrix} -\mathbf{y}_{k-1} & \cdots & -\mathbf{y}_{k-n_A} & \mathbf{u}_{k-1} & \cdots & \mathbf{u}_{k-n_B} \end{bmatrix} \quad (4)$$

$$\mathbf{p} \stackrel{\text{def}}{=} \begin{bmatrix} a_1 & \cdots & a_{n_A} & b_1 & \cdots & b_{n_B} \end{bmatrix}^T \quad (5)$$

Multiple instances of Eq. 2, for increasing values of k, can be stacked in a column vector and then written in the form of Eq. 6.

$$\mathbf{y}_k = \mathbf{R}_k \, \mathbf{p} + \mathbf{f}_k \quad (6)$$

The ith row of Eq. 6 is equivalent to Eq. 2 for $k := k + i - 1$. Assuming that the first timestep in a timeseries has index 1, the lowest value for k is $k = \max(n_A, n_B) + 1$. The number of rows in Eq. 6 is limited by the number of samples in the timeseries.

The ARX algorithm is to solve Eq. 6 with respect to $\mathbf{p}$ using the ordinary least squares (OLS) method, where $\mathbf{f}_k$ is considered the residual. Hence the model identified by the ARX algorithm is given by Eq. 1 or, equivalently, by Eq. 2, where the model parameters are given by the parameter vector $\mathbf{p}$ computed from Eq. 7.

$$\mathbf{p} = (\mathbf{R}_k^T \, \mathbf{R}_k)^{-1} \, \mathbf{R}_k^T \, \mathbf{y}_k \quad (7)$$

The term ARMAX refers to a dynamic model on the form of Eq. 8. The DARX algorithm uses the ARX algorithm twice to identify a model on the ARMAX form.

$$A(z) \, y_k = B(z) \, u_k + C(z) \, f_k \quad (8)$$

DSR E AND DARX

DSR E is a system identification algorithm for identification of deterministic / stochastic linear time-invariant (LTI) systems. Two somewhat different versions of DSR E are derived in references [1] and [4], respectively.

The DSR E algorithm consists of two steps. Step 1: The identification problem is reduced from a deterministic / stochastic problem to a deterministic problem. The purpose of this reduction is that identification of a deterministic system is a much simpler problem. The stochastic part of the problem is eliminated by identifying the innovation process, f, and then consider the identified innovation process as a deterministic input. Step 2: Identify the deterministic system having u and f as deterministic inputs, and y as output. For a detailed explanation of DSR E, please refer to references [1, 4].

Reference [2] shows that the first step of DSR E is mathematically identical to the ARX algorithm: The identified innovation process, f, is mathematically identical to the one-step-ahead prediction error as the identified ARX model is simulated on its own training dataset. After this first step, the problem is reduced to a deterministic identification problem. This deterministic problem can be solved using ARX. Hence each of the two steps of DSR E can be replaced by or approximated by the ARX algorithm. The total DSR E algorithm can then be approximated by a two-step ARX algorithm. This approximation is referred to as the DARX algorithm [2].

It should be emphasized that DARX is *not* a new system identification algorithm. It is an approximation of DSR E that may be easier to understand and easier to implement in a computer program. DARX is also interesting because it shows the close relationship between ARX and DSR E, even though their derivations may seem quite different.

The DARX algorithm is derived as follows: Define matrix $\mathbf{S}_k$ as in Eq. 9.

$$\mathbf{S}_k \stackrel{\text{def}}{=} \begin{bmatrix} -\mathbf{y}_{k-1} & \cdots & -\mathbf{y}_{k-J} & \mathbf{u}_{k-1} & \cdots & \mathbf{u}_{k-J} \end{bmatrix} \quad (9)$$

Matrix $\mathbf{S}_k$ is to be used during the first step of the DARX algorithm. It is identical to $\mathbf{R}_k$ of Eq. 4, with n_A and n_B set to J. Define matrix $\mathbf{M}_k$ as in Eq. 10.

$$\mathbf{M}_k = \mathbf{S}_k \, (\mathbf{S}_k^T \, \mathbf{S}_k)^{-1} \, \mathbf{S}_k^T \quad (10)$$

Matrix $\mathbf{M}_k$ is a projection matrix projecting onto the column space of $\mathbf{S}_k$. Putting Eq. 7 into Eq. 6, replacing $\mathbf{R}_k$ by $\mathbf{S}_k$, and using Eq. 10 gives Eq. 11.

$$\mathbf{y}_k = \mathbf{M}_k \, \mathbf{y}_k + \mathbf{f}_k \quad (11)$$

The term $\mathbf{M}_k \, \mathbf{y}_k$ is the projection of $\mathbf{y}_k$ onto the column space of $\mathbf{S}_k$. Equivalently: $\mathbf{M}_k \, \mathbf{y}_k$ is the part of $\mathbf{y}_k$ that can be explained from the inputs, u, and outputs, y, as organized in the matrix $\mathbf{S}_k$. The part of $\mathbf{y}_k$ that can *not* be explained from $\mathbf{S}_k$ is the residual $\mathbf{f}_k$. DARX uses $\mathbf{f}_k$, as solved from Eq. 11, as an estimate of the innovation process, i.e. the *identified* innovation process. In other words: The first step of DARX

is to compute $\mathbf{f}_k$ by solving Eq. 11 with respect to $\mathbf{f}_k$. The choice of J in Eq. 9 is most important. This is discussed in reference [2].

The second step of DARX is to develop an ARMAX model, i.e. a model on the form of Eq. 8. This model is the final model identified by DARX. Under the assumptions that the system to be identified is a perfect LTI system, and that the innovation process is perfectly identified, i.e. $f_k = e_k \,\forall\, k$, the second step of DARX is a deterministic identification problem. Even though these assumptions are not perfectly met for most real-life systems, the second step of DARX is still considered as a deterministic identification problem. During the second step of DARX, both u and f are considered deterministic inputs. The output is y.

The ARMAX model of Eq. 8 can be written as Eq. 12.

$$\begin{aligned} y_k \;=\; & -a_1\, y_{k-1} - \ldots - a_{n_A}\, y_{k-n_A} \\ & +b_1\, u_{k-1} + \ldots + b_{n_B}\, u_{k-n_B} \\ & +f_k + c_1 f_{k-1} + \ldots + c_{n_C}\, f_{k-n_C} \end{aligned} \tag{12}$$

Please notice that no coefficient is associated with f_k in Eq. 12. Moving f_k to the left side gives Eq. 13.

$$\begin{aligned} y_k - f_k \;=\; & -a_1\, y_{k-1} - \ldots - a_{n_A}\, y_{k-n_A} \\ & +b_1\, u_{k-1} + \ldots + b_{n_B}\, u_{k-n_B} \\ & +c_1 f_{k-1} + \ldots + c_{n_C}\, f_{k-n_C} \end{aligned} \tag{13}$$

In Eq. 13 all y, u, and f on the right side are associated with coefficients. Multiple instances of Eq. 13, for increasing values of k, can be stacked in a column vector and then written in the form of Eq. 14.

$$\mathbf{y}_k - \mathbf{f}_k = \mathbf{T}_k\, \mathbf{q} \tag{14}$$

The regression matrix $\mathbf{T}_k$ contains, from left to right, the column vectors $-\mathbf{y}_{k-1}$, ..., $-\mathbf{y}_{k-n_A}$, $\mathbf{u}_{k-1}$, ..., $\mathbf{u}_{k-n_B}$, $\mathbf{f}_{k-1}$, ..., $\mathbf{f}_{k-n_C}$. The parameter vector $\mathbf{q}$ is a column vector containing the elements a_1, ..., a_{n_A}, b_1, ..., b_{n_B}, c_1, ..., c_{n_C}. Using OLS the parameter vector $\mathbf{q}$ is given by Eq. 15.

$$\mathbf{q} = (\mathbf{T}_k^T\, \mathbf{T}_k)^{-1}\, \mathbf{T}_k^T\, (\mathbf{y}_k - \mathbf{f}_k) \tag{15}$$

Hence the final model identified by the DARX algorithm is an ARMAX model, where the parameters of the polynomials $A(z)$, $B(z)$, and $C(z)$ are the elements of the parameter vector $\mathbf{q}$. It is most important to understand that the definition of the innovation process requires the constant term of $C(z)$ to be 1, i.e. no coefficient can be associated with f_k in Eq. 12. How to select n_A, n_B, and n_C is a most important issue. However, this issue is beyond the scope of this paper.

IMPLEMENTATION OF DARX

When implementing DARX in a computer program, the implementation can be simplified using standard ARX software, such as the MATLAB command `arx`, instead of implementing the ARX algorithm from scratch. Using MATLAB, the first step of DARX is straight forward: An ARX model is identified using `arx`, then the innovation process, f, is identified using `pe` (prediction error).

Unfortunately, it is *not* straight forward to use `arx` to implement the second step of DARX. The main contribution of this paper is to explain the reason for this issue, and to provide a rewriting such that `arx` can be used for the second step of DARX after all. To simplify the discussion, the values $n_A = 1$, $n_B = 1$, and $n_C = 1$ will be used, unless otherwise is specified. However, the result to be presented is valid for any values of n_A, n_B, and n_C.

During the second step of DARX, u and f are considered deterministic inputs. The output is y. Using `arx` straight forward, a model will be identified by solving the linear regression problem of Eq. 16.

$$\begin{aligned} &\mathbf{y}_k = \\ &\begin{bmatrix} -\mathbf{y}_{k-1} & \mathbf{u}_{k-1} & \mathbf{f}_k & \mathbf{f}_{k-1} \end{bmatrix} \\ &\times \begin{bmatrix} a_1 & b_1 & c_0 & c_1 \end{bmatrix}^T \end{aligned} \tag{16}$$

The problem to be handled is the parameter c_0. This parameter should *not* be subject to identification, because its value must be 1 according to the definition of innovation process. The solution is to remove c_0 from the parameter vector and move the column vector $\mathbf{f}_k$ to the left side. Define $\bar{\mathbf{y}}_k \stackrel{\text{def}}{=} \mathbf{y}_k - \mathbf{f}_k \,\forall\, k$. The regression problem is then on the form of Eq. 17.

$$\begin{aligned} &\bar{\mathbf{y}}_k = \\ &\begin{bmatrix} -\mathbf{y}_{k-1} & \mathbf{u}_{k-1} & \mathbf{f}_{k-1} \end{bmatrix} \\ &\times \begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix}^T \end{aligned} \tag{17}$$

Eq. 17 violates the standard ARX form of Eq. 16, because Eq. 17 has $\bar{\mathbf{y}}_k$ on the left side, but the column in the regression matrix associated with a_1 is $-\mathbf{y}_{k-1}$.

The standard ARX form requires either $\mathbf{y}_k$ on the left side and $-\mathbf{y}_{k-1}$ associated with a_1, or $\bar{\mathbf{y}}_k$ on the left side and $-\bar{\mathbf{y}}_{k-1}$ associated with a_1. As the standard ARX form is violated, `arx` can not be used directly to identify the parameter vector of Eq. 17.

A solution to this problem is to implement from scratch a tailor-made, non-standard ARX algorithm that assumes the form of Eq. 17. However, this will complicate the implementation of DARX. The simpler solution is to rewrite Eq. 17 to the standard ARX form. This rewriting is split into two cases: Case 1 is to be applied when $n_A \leq n_C$. Case 2 is to be applied when $n_A \geq n_C$. When $n_A = n_C$, either of the cases can be applied.

Case 1: $n_A \leq n_C$

This derivation is valid for any values of n_A, n_B, and n_C which obey the constraint $n_A \leq n_C$. The values $n_A = 1$, $n_B = 1$, and $n_C = 2$ will be used to simplify the discussion. Eq. 18 is identical to Eq. 17, except that n_C is increased from 1 to 2.

$$\begin{gathered}\bar{\mathbf{y}}_k = \\ \begin{bmatrix} -\mathbf{y}_{k-1} & \mathbf{u}_{k-1} & \mathbf{f}_{k-1} & \mathbf{f}_{k-2} \end{bmatrix} \\ \times \begin{bmatrix} a_1 & b_1 & c_1 & c_2 \end{bmatrix}^T\end{gathered} \tag{18}$$

The issue is how to rewrite Eq. 18 to the standard ARX form, so that `arx` can be used to identify the parameter vector. The column of the regression matrix associated with the parameter a_1 must be $-\bar{\mathbf{y}}_{k-1}$, not $-\mathbf{y}_{k-1}$, because the left side is $\bar{\mathbf{y}}_k$, not $\mathbf{y}_k$. It must be required that the rewriting does not change the column space of the regression matrix, otherwise the parameter vector will be identified erroneously. Adding a column of a matrix to another column of the same matrix does not change the column space of the matrix. This is a basic result of linear algebra. The desired rewriting is achieved by adding the column $\mathbf{f}_{k-1}$ to the column $-\mathbf{y}_{k-1}$. This column summation changes Eq. 18 into the form of Eq. 19.

$$\begin{gathered}\bar{\mathbf{y}}_k = \\ \begin{bmatrix} -\bar{\mathbf{y}}_{k-1} & \mathbf{u}_{k-1} & \mathbf{f}_{k-1} & \mathbf{f}_{k-2} \end{bmatrix} \\ \times \begin{bmatrix} \bar{a}_1 & \bar{b}_1 & \bar{c}_1 & \bar{c}_2 \end{bmatrix}^T\end{gathered} \tag{19}$$

Eq. 19 is on the standard ARX form. Hence the parameter vector of Eq. 19 can be identified using `arx`. Three important comments regarding this derivation: (i) The output specified to `arx` should be $\bar{y} = y - f$, not y. The deterministic inputs are u and f. (ii) The columns $\mathbf{u}_k$ and $\mathbf{f}_k$ are not in the regression matrix of Eq. 19. This is equivalent to the transfer functions from u to $\bar{y}$, and from f to $\bar{y}$, having a time delay of one sample. This should be specified to `arx` by setting the time delay parameter `nk` to 1 for both inputs. (iii) The parameter vectors of Eq. 18 and Eq. 19 differ (the elements in the parameter vector of Eq. 19 have bars). The desired parameter vector of Eq. 18 is related to the identified parameter vector of Eq. 19 by $a_1 = \bar{a}_1$, $b_1 = \bar{b}_1$, $c_1 = \bar{a}_1 + \bar{c}_1$, and $c_2 = \bar{c}_2$.

This derivation can easily be generalized to any values of n_A, n_B, and n_C that obey the constraint $n_A \leq n_C$. The desired parameter vector of Eq. 18 is related to the identified parameter vector of Eq. 19 by $a_i = \bar{a}_i \,\forall\, i$, $b_i = \bar{b}_i \,\forall\, i$, $c_i = \bar{a}_i + \bar{c}_i \,\forall\, i \in \{1,\ldots,n_A\}$, and $c_i = \bar{c}_i \,\forall\, i > n_A$.

Case 2: $n_A \geq n_C$

The case $n_A \geq n_C$ is somewhat more complex. The derivation to be presented is valid for any n_A, n_B, and n_C which obey the constraint $n_A \geq n_C$. The values $n_A = 2$, $n_B = 1$, and $n_C = 1$ will be used to simplify the discussion. Eq. 20 is identical to Eq. 17, except that n_A is increased from 1 to 2.

$$\begin{gathered}\bar{\mathbf{y}}_k = \\ \begin{bmatrix} -\mathbf{y}_{k-1} & -\mathbf{y}_{k-2} & \mathbf{u}_{k-1} & \mathbf{f}_{k-1} \end{bmatrix} \\ \times \begin{bmatrix} a_1 & a_2 & b_1 & c_1 \end{bmatrix}^T\end{gathered} \tag{20}$$

For $n_A > n_C$ the rewriting presented for $n_A \leq n_C$ can not be applied. In Eq. 20 the column $-\mathbf{y}_{k-1}$ can be changed to $-\bar{\mathbf{y}}_{k-1}$ by adding the column $\mathbf{f}_{k-1}$. However, the column $-\mathbf{y}_{k-2}$ can *not* be changed to $-\bar{\mathbf{y}}_{k-2}$, because $\mathbf{f}_{k-2}$ is not in the column space of the regression matrix. In other words: Adding $\mathbf{f}_{k-2}$ would change the column space of the regression matrix. This would cause the DARX implementation to identify erroneous models.

Rewriting the $n_A \geq n_C$ case includes rearranging the columns of the regression matrix. The columns $-\mathbf{y}_{k-1}$ and $-\mathbf{y}_{k-2}$ are moved to the right, and the column $\mathbf{f}_{k-1}$ is moved to the left as shown in Eq. 21.

$$\begin{gathered}\bar{\mathbf{y}}_k = \\ \begin{bmatrix} \mathbf{f}_{k-1} & \mathbf{u}_{k-1} & -\mathbf{y}_{k-1} & -\mathbf{y}_{k-2} \end{bmatrix} \\ \times \begin{bmatrix} \bar{a}_1 & \bar{b}_1 & \bar{c}_1 & \bar{c}_2 \end{bmatrix}^T\end{gathered} \tag{21}$$

Please note the change of the parameter vector from Eq. 20 to Eq. 21. The column $\mathbf{f}_{k-1}$ can be changed

to the desired form, $-\bar{\mathbf{y}}_{k-1}$, by adding the column $-\mathbf{y}_{k-1}$. This column summation does not change the column space of the regression matrix. The regression problem is now on the form of Eq. 22.

$$\bar{\mathbf{y}}_k = \begin{bmatrix} -\bar{\mathbf{y}}_{k-1} & \mathbf{u}_{k-1} & -\mathbf{y}_{k-1} & -\mathbf{y}_{k-2} \end{bmatrix} \times \begin{bmatrix} \tilde{a}_1 & \tilde{b}_1 & \tilde{c}_1 & \tilde{c}_2 \end{bmatrix}^T \quad (22)$$

Please note the change of the parameter vector from Eq. 21 to Eq. 22. Eq. 22 is on the standard ARX form. Hence its parameter vector can be identified using `arx`. There are three important comments also for this derivation: (i) The inputs specified to `arx` are u and $-y$, and the output is $\bar{y} = y - f$. (ii) The time delay parameter `nk` of `arx` should be set to 1 for both inputs. (iii) The desired parameter vector of Eq. 20 is related to the identified parameter vector of Eq. 22 by $a_1 = \tilde{a}_1 + \tilde{c}_1$, $a_2 = \tilde{c}_2$, $b_1 = \tilde{b}_1$, and $c_1 = \tilde{a}_1$.

This derivation can easily be generalized to any values of n_A, n_B, and n_C which obey the constraint $n_A \geq n_C$. The desired parameter vector of Eq. 20 is related to the identified parameter vector of Eq. 22 by $a_i = \tilde{a}_i + \tilde{c}_i \,\forall\, i \in \{1,\ldots,n_C\}$, $a_i = \tilde{c}_i \,\forall\, i > n_C$, $b_i = \tilde{b}_i \,\forall\, i$, and $c_i = \tilde{a}_i \,\forall\, i$.

THE CZOCHRALSKI PROCESS

The Czochralski (CZ) crystallization process is named after its inventor, the Polish scientist Jan Czochralski. The CZ process is used to grow monocrystals (single crystals). Production of monocrystalline silicon is among the most important applications of the CZ process. Monocrystalline silicon is used in solar cells as well as computers and electronics.

The CZ process is a batch process. Initially multicrystalline silicon is melted in a crucible. Dopant materials, such as boron or phosphorus, may be added to the crucible. When the silicon is all molten, a seed crystal of monocrystalline silicon is dipped into the melt. As the seed crystal is slowly elevated, the molten silicon solidifies on the seed crystal. The seed crystal then grows radially and axially, while preserving its crystal structure. The produced rod of monocrystalline silicon is referred to as the ingot. The CZ process is performed in an inert atmosphere, such as an argon atmosphere. Figure 1 illustrates the principle of operation of the CZ process for production of monocrystalline silicon.

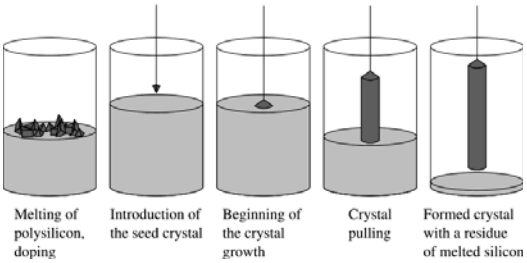


Figure 1: The Czochralski process, principle of operation. Illustration from Wikipedia.

With respect to control of the CZ process, a tight control of the ingot diameter is most important in order to minimize cutting waste. There are also several quality parameters that should be met, such as dislocation level, impurity content, and dopant distribution. Unfortunately, these quality parameters can not be measured online and can therefore not be used as control objectives for control strategies [3].

A control strategy for the CZ process is presented in reference [3]. This control strategy is based on single loop control, cascade control, and feedforward trajectories. The control strategy is thoroughly explained in reference [3] and will not be repeated here.

VALIDATION OF THE DARX REWRITING

This paper presents a rewriting of the second step of the DARX algorithm. This rewriting allows `arx` to be used also for the second step of DARX, canceling the need for implementing a tailor-made ARX algorithm. Hence the rewriting simplifies the overall implementation of DARX.

In this section the DARX rewriting is validated using timeseries logged at a real-life CZ process owned and operated by SINTEF Materials and Chemistry in Trondheim, Norway. At this CZ process the temperature of the crucible is controlled by adjusting the electric power to a heating element using TRIACs. The temperature is measured using a pyrometer. The DARX rewriting is validated by identifying the transfer function from the control signal of the TRIACs, P, to the voltage output signal from the pyrometer, T. The timeseries used for identification are shown in Figure 2. These timeseries were logged during heating of the crucible before the actual crystal pulling started, because it then could be

done larger excitations in P than during the crystal pulling.

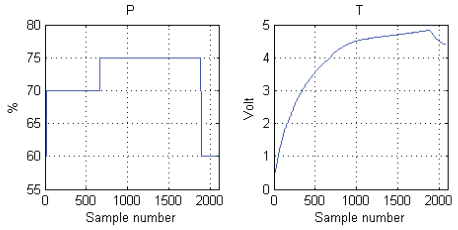


Figure 2: Timeseries P and T used for identification of the transfer function from P to T.

In this validation four implementations of DARX, referred to as A, B, C, and D, will be compared. Implementation A uses a tailor-made, non-standard ARX algorithm to handle the second step of DARX. In other words: Implementation A handles the form of Eq. 17 directly. The main disadvantage of this approach is that implementing the tailor-made ARX algorithm significantly complicates the implementation of DARX. Implementations B, C, and D use the rewriting presented in this paper and `arx` to handle the second step of DARX. Implementation B implements only case 1, i.e. $n_A \leq n_C$. Implementation C implements only case 2, i.e. $n_A \geq n_C$. Implementation D calls implementation B as a sub-routine if $n_A \leq n_C$, otherwise implementation D calls implementation C. Implementations A, B, C, and D all use `arx` for the first step of DARX.

The validation of the DARX rewriting consists of three tests. In each test the transfer function from the control signal P to the measurement signal T is identified based on the timeseries shown in Figure 2. The same timeseries are used in all three tests. In each test four models are identified, one for each of the implementations A to D. Implementation A is known to work correctly and is used as a reference for validating the other implementations.

In test #1 the values $n_A = 1$, $n_B = 1$, and $n_C = 2$ are used. Hence the parameters a_1, b_1, c_1, and c_2 are to be identified. The identified parameters are shown in Figure 3. Implementations A, B, and D identify the parameters identically. Hence B and D identify the parameters correctly as A is known to be correct. This was expected, because B is implemented to handle the constraint $n_A \leq n_C$, which is obeyed, and D calls B as a sub-routine when this constraint is obeyed. Implementation C identifies the parameters erroneously. This is because C assumes the constraint $n_A \geq n_C$, which is violated.

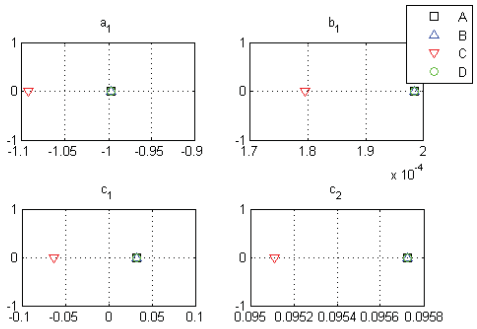


Figure 3: Parameters a_1, b_1, c_1, and c_2 as identified in test #1.

Test #2 uses the values $n_A = 2$, $n_B = 1$, and $n_C = 1$. The parameters a_1, a_2, b_1, and c_1 are then to be identified. The identified parameters are shown in Figure 4. In this test C and D identify the parameters correctly, while B does not. This is because B assumes the constraint $n_A \leq n_C$, which is violated in this test.

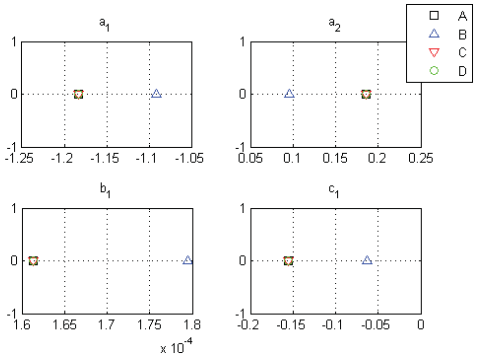


Figure 4: Parameters a_1, a_2, b_1, and c_1 as identified in test #2.

In test #3, the values $n_A = 2$, $n_B = 1$, and $n_C = 2$ are used. Hence the parameters a_1, a_2, b_1, c_1, and c_2 are to be identified. In this test, all implementations identify the parameters correctly, because the constraints $n_A \leq n_C$ and $n_A \geq n_C$ both are obeyed. This result is *not* illustrated graphically.

In all three tests the parameter J, which was intro-

duced in Eq. 9, was set to $J = 10$. This parameter does not influence the validation result because it is only used during step 1 of DARX, while the rewriting only considers step 2.

Summing up the tests shows that implementation B, which is based on case 1, identifies the parameters correctly if and only if the constraint $n_A \leq n_C$ is obeyed. Implementation C, which is based on case 2, identifies the parameters correctly if and only if the constraint $n_A \geq n_C$ is obeyed. Hence the results of the validation back up the mathematical derivation of the rewriting. It is then concluded that the rewriting works correctly.

CONCLUSION

DARX is a two-step ARX algorithm which approximates the DSR E system identification algorithm. The first step of DARX is on the standard ARX form, allowing standard ARX software, such as the MATLAB command `arx`, to be used directly. The second step of DARX slightly differs from the standard ARX form. This issue can be handled by implementing from scratch a tailor-made, non-standard ARX algorithm. However, this approach significantly complicates the otherwise simple implementation of DARX.

The main contribution of this paper is to present a rewriting of the second step of DARX. Using this rewriting standard ARX software can be used also for the second step of DARX. Hence the need for a tailor-made ARX algorithm is canceled.

The Czochralski (CZ) crystallization process is used to grow monocrystals (single crystals). Production of monocrystalline silicon is among the most important applications of the CZ process. Monocrystalline silicon is used in solar cells, computers, and electronics.

The rewriting of the second step of DARX has been validated by comparing four implementations of DARX. These implementations were tested by identifying a transfer function in the CZ process based on timeseries logged at a real-life CZ process. One implementation uses a tailor-made, non-standard ARX algorithm to handle the second step of DARX. The other three implementations use the rewriting presented in this paper. As the results of this validation back up the mathematical derivation, it is concluded that the rewriting is correct.

ACKNOWLEDGMENTS

The authors are most grateful to SINTEF Materials and Chemistry in Trondheim, Norway, for allowing timeseries logged at a CZ process owned and operated by the organization to be used in this paper. The authors are also very grateful to Eivind Johannes Øvrelid at SINTEF Materials and Chemistry for sharing his knowledge and experience of the CZ process. Beathe Furenes at Prediktor AS is acknowledged for her participation and contribution to the CZ batch of which the timeseries used in this paper were logged.

The main author is most grateful for the financial support of his PhD study from NorSun AS, Østfold Energi AS, and the Norwegian Research Council. The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is also very much appreciated.

REFERENCES

[1] D. Di Ruscio. Subspace system identification of the Kalman filter: Open and closed loop systems. In *The 6th international conference on Computing, Communication and Control Technologies (CCCT 2008)*, Orlando, Florida, June 29th - July 2nd 2008.

[2] M. Komperød, TA. Hauge, D. Di Ruscio, and B. Lie. Empirical modeling: Approximating the DSR E sub-space system identification algorithm by a two-step ARX algorithm. In *The 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008)*, October 7th - 8th 2008.

[3] K. Lee, D. Lee, J. Park, and M. Lee. MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 3:252–257, 2005.

[4] GW. Nilsen. *Topics in Open and Closed Loop Subspace System Identification: Finite Data-Based Methods*. PhD thesis, Norwegian University of Science and Technology / Telemark University College, Trondheim / Porsgrunn, Norway, 2006.

Considerations of Paper C

This paper uses the notation e for the true, but unknown, innovation process, and the notation f for the identified innovation process. This notation differs from the notation used in Paper B, where the symbols ε and ϵ, respectively, were used. It is unfortunate not to be consistent with the notation. In this paper, the symbols $\mathbf{p}$ and $\mathbf{q}$ are used for two different parameter vectors. This is also an unfortunate notation as θ is usually used for parameter vectors in the system identification literature.

The candidate has used the term "timeseries" somewhat erroneously in this paper compared to how the term is used in Ljung (1999). The paper uses the term for datasets logged from processes or made by dynamic simulations, while Ljung (1999) uses the term for "outputs of dynamical systems whose external stimuli are not observed".

Paper D

Empirical Modeling of Heating Element Power for the Czochralski Crystallization Process

This article was published in the open-access journal Modeling, Identification and Control (MIC) in 2010. The research presented in this article is based on logged process data from an experiment at the Czochralski (CZ) crystallization process at SINTEF Materials and Chemistry in Trondheim, Norway. The candidate did not participate in this experiment. The candidate has done all the research presented in the article and written the article during his PhD study.

This article and Paper E present modeling work on the same part of the CZ process. However, different modeling approaches are used in the two publications. In this article the method for outlier detection presented in Paper A is used. After the article, on page 105, a few considerations of the article are discussed.

PAPER D 89
Modeling, Identification and Control, Vol. 31, No. 1, 2010, pp. 19–34, ISSN 1890–1328

Empirical Modeling of Heating Element Power for the Czochralski Crystallization Process

M. Komperød [1] B. Lie [2]

[1] *Faculty of Engineering, Østfold University College, N-1757 Halden, Norway. E-mail:* magnus.komperod@hiof.no

[2] *Faculty of Technology, Telemark University College, N-3901 Porsgrunn, Norway. E-mail:* bernt.lie@hit.no

Abstract

The Czochralski (CZ) crystallization process is used to produce monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. The CZ process is a batch process, where multicrystalline silicon is melted in a crucible and later solidifies on a monocrystalline seed crystal. The crucible is heated using a heating element where the power is manipulated using a triode for alternating current (TRIAC). As the electric resistance of the heating element increases by increased temperature, there are significant dynamics from the TRIAC input signal (control system output) to the actual (measured) heating element power. The present paper focuses on empirical modeling of these dynamics. The modeling is based on a dataset logged from a real-life CZ process. Initially the dataset is preprocessed by detrending and handling outliers. Next, linear ARX, ARMAX, and output error (OE) models are identified. As the linear models do not fully explain the process' behavior, nonlinear system identification is applied. The Hammerstein-Wiener (HW) model structure is chosen. The final model identified is a Hammerstein model, i.e. a HW model with nonlinearity at the input, but not at the output. This model has only one more identified parameter than the linear OE model, but still improves the optimization criterion (mean squared ballistic simulation errors) by a factor of six. As there is no nonlinearity at the output, the dynamics from the prediction error to the model output are linear, which allows a noise model to be added. Comparison of a Hammerstein model with noise model and the linear ARMAX model, both optimized for mean squared one-step-ahead prediction errors, shows that this optimization criterion is 42% lower for the Hammerstein model. Minimizing the number of parameters to be identified has been an important consideration throughout the modeling work.

Keywords: Czochralski Crystallization Process, Empirical Modeling, Hammerstein-Wiener Model, Heating Element, Nonlinear System Identification.

1 Introduction

The Czochralski (CZ) crystallization process was invented by the Polish scientist Jan Czochralski in 1916. The process is used to convert multicrystalline materials into monocrystalline materials, i.e. materials that have homogeneous crystal structures. Among the most important applications is production of monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Solar cells based on monocrystalline wafers have higher efficiency than those based on multicrystalline wafers. Hence, the CZ process is an important part of the solar cell industry.

The CZ process is a batch process. During the process multicrystalline silicon is melted in a crucible. The crucible is heated using a heating element. Tight control of the crucible temperature is most important for achieving high crystal quality. The heating element

doi:10.4173/mic.2010.1.2

power is manipulated using a triode for alternating current (TRIAC). As the electric resistance of the heating element increases with the temperature, there is a dynamic, nonlinear relationship between the TRIAC input signal and the heating element power.

System identification is the science of developing *dynamic* models based on observations of the process or system to be modeled. The identified models explain the process outputs as mathematical functions of the process inputs.

The contribution of this paper is to model the dynamics from the TRIAC input signal to the heating element power using system identification. The modeling work is based on a dataset logged at a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway. Initially, linear system identification is used. The identified linear models reveal that the process is nonlinear. Next the process is modeled using nonlinear system identification. Deciding a model structure that explains the process behavior using few parameters is emphasized. Also, providing the identification algorithm good initial values for the parameters to be identified is considered.

As the identified model provides a mathematical description of how the heating element power responds to the TRIAC input signal, the model serves several purposes. The most intuitive application is to simulate the power for specified sequences of the TRIAC input signal. The model is also most useful for analyzing the process' dynamic properties. Process models may be used to tune PID controllers. As the process is nonlinear, it may be desirable to use gain scheduling. The nonlinear process model contains information of which PID parameters to use for each interval in the gain scheduling scheme. If more advanced model-based control strategies are to be applied, such as model predictive control (MPC), process models are most important. In case of a power sensor failure, the model may be used to simulate the power for the purpose of process monitoring and control.

The literature of system identification is extensive. Among the most well-known books is Ljung (1999). This book gives a comprehensive introduction to system identification. Both theoretical and practical aspects are covered. Ljung (1999) also serves as an overview of the system identification literature.

Lee et al. (2005) presents an approach for batch-to-batch optimization of the CZ process, which includes model-based control. The paper includes two simple dynamic models empirically developed from step responses. However, these models cover different parts of the CZ process than the present paper. Except for Lee et al. (2005), the authors of the present paper have not been successful in finding any publications that present results within empirical modeling of the CZ process. Several papers with focus towards mechanistic (first principle) modeling of the CZ process have been found. However, none of these papers seem to have validated the mechanistic models against datasets from real-life processes.

The authors have searched for literature covering modeling of heating element dynamics in general. Unfortunately, no interesting results were found.

2 Notations and Definitions

Table 1 presents the notations used in this paper. A variable with subscript k refers to the variable's value at timestep k. For example P_k refers to the heating element power at timestep k. Subscript "nom" refers to the variable's nominal value. The operating point $(s_{\text{nom}}, P_{\text{nom}})$ is defined to be a steady state operating point.

Polynomials to be used in linear polynomial models are defined as

$$A(q) \stackrel{\text{def}}{=} 1 + \sum_{i=1}^{n_a} a_i q^{-i}, \tag{1}$$

$$B(q) \stackrel{\text{def}}{=} \sum_{i=1}^{n_b} b_i q^{-i}, \tag{2}$$

$$C(q) \stackrel{\text{def}}{=} 1 + \sum_{i=1}^{n_c} c_i q^{-i}, \tag{3}$$

$$F(q) \stackrel{\text{def}}{=} 1 + \sum_{i=1}^{n_f} f_i q^{-i}. \tag{4}$$

The parameters a_i, b_i, c_i, and f_i are to be identified using system identification. The constants n_a, n_b, n_c, and n_f are to be specified to the system identification algorithm. There is a time delay of one sample in the dataset to be used in this paper. For processes without any time delay, the lower summation limit of $B(q)$ should be 0 instead of 1. The $A(q)$, $C(q)$, and $F(q)$ polynomials are defined as above in any case.

The linear polynomial models that will be considered in this paper are based on the ARX model structure

$$A(q)y_k = B(q)u_k + e_k, \tag{5}$$

the ARMAX model structure

$$A(q)y_k = B(q)u_k + C(q)e_k, \tag{6}$$

and the output error (OE) model structure

Table 1: Notations used in this paper.

$e(\theta)$	One-step-ahead prediction error.
k	Index referring to sample number in the dataset.
N	The total number of samples in the dataset.
P	Actual (measured) power to the heating element [kW].
P_{nom}	Nominal power to heating element [kW] (see main text for explanation).
ΔP	Actual (measured) power to the heating element [kW] as deviation from the nominal power, i.e. $\Delta P = P - P_{\text{nom}}$.
q	The time-shift operator defined by $x_{k+1} = qx_k$ and $x_{k-1} = q^{-1}x_k$.
s	TRIAC input signal [%] (output from control system).
s_{nom}	Nominal TRIAC input signal [%] (see main text for explanation).
Δs	TRIAC input signal [%] as deviation from the nominal input signal, i.e. $\Delta s = s - s_{\text{nom}}$.
t	Time [s], relative to beginning of the CZ batch.
u	System input for a general system.
$V(\theta)$	Criterion to be optimized when using the system identification method PEM. This criterion is based on ballistic simulation errors. Please refer to Section 4 for further explanation.
$W(\theta)$	Criterion to be optimized when using the system identification method PEM. This criterion is based on one-step-ahead prediction errors. Please refer to Section 4 for further explanation.
y	System output for a general system.
$\varepsilon(\theta)$	Ballistic simulation error.
θ	A vector containing the parameters to be identified using PEM. Please refer to Section 4 for further explanation.

$$y_k = \frac{B(q)}{F(q)}u_k + \varepsilon_k. \tag{7}$$

The term noise model refers to the dynamics from the one-step-ahead prediction error, e, to the system output, y. Solving (5) and (6) with respect to y_k shows that the noise models of ARX and ARMAX are $1/A(q)$ and $C(q)/A(q)$, respectively. The OE model has no noise model. Model structures having noise models, such as ARX and ARMAX, are suitable for n-step-ahead predictions, because the noise models correct the model states when there are errors between the measured process outputs and the simulated model outputs. Model structures having no noise model, like OE, are only suitable for ballistic simulations, not n-step-ahead predictions.

The term input-output model will be used to refer to the dynamics from the system input, u, to the system output, y. With respect to the input-output model, $A(q)$ of the ARX and ARMAX model structures is equivalent to $F(q)$ of the OE model structure.

Text written in `teletype font` refers to MATLAB commands, for example `pem`.

3 The Czochralski Crystallization Process

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials. The process considered in this paper converts multicrystalline silicon into monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Monocrystalline silicon wafers give solar cells with higher efficiency than multicrystalline silicon wafers.

The CZ process is a batch process of which main steps are illustrated in Figure 1. (i) Initially multicrystalline silicon is melted in a crucible. (ii) When the silicon is molten, the tip of a seed crystal is dipped into the melt. The seed crystal is monocrystalline and has the crystal structure that is to be produced. (iii) When the tip of the seed crystal begins to melt, the crystal is slowly elevated. As the crystal is lifted, the molten silicon solidifies on the crystal. (iv) The crystal grows radially and axially. The produced crystal is referred to as an ingot. The crucible temperature and the vertical pulling speed are used to control the ingot diameter. Stable growing conditions are necessary to produce high crystal quality. (v) As the final ingot length is reached, the crystal growth is terminated by slowly decreasing the crystal diameter to zero. During the entire batch process, the crucible is rotated in one

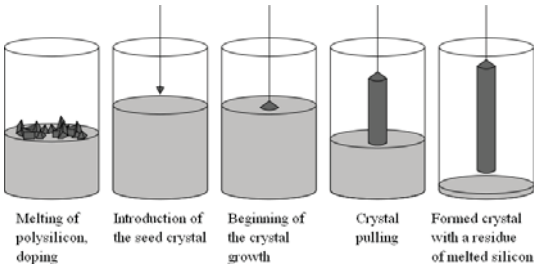


Figure 1: The main batch steps of the CZ process. Illustration from Wikipedia (the illustration is released to public domain by the copyright holder).

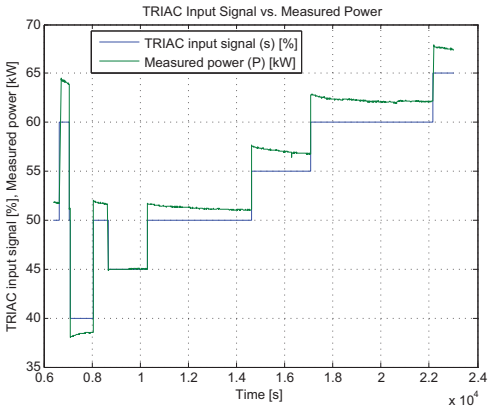


Figure 2: The dataset to be used for system identification.

direction, and the seed crystal is rotated in the other direction.

SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a real-life CZ process. At this plant the crucible is heated using a cylinder-shaped heating element, which encircles the crucible. The heating element power is manipulated using a TRIAC. This paper considers empirical modeling of the dynamics from the TRIAC input signal, s, to the heating element power, P, based on a dataset logged from this plant.

The dataset that is used for empirical modeling is shown in Figure 2. The experiment was done without the crucible mounted in the process. The figure shows that increased s gives instantaneously increased P. However, as the temperature increases, the electric resistance of the heating element increases, causing decreased P. As shown later in this paper, these dynamics are nonlinear.

The authors have access to only one dataset that is considered to be suitable for the modeling work. This dataset will be used for model identification as well as model validation. The authors have chosen to use the entire dataset for both identification and validation. The reason for not splitting the dataset into an identification section and a validation section is that s tends to increase by time throughout the dataset. Hence, splitting the dataset in two halves will leave a small range of excitations in each half. As concluded in Section 6, the process is almost linear close to an operating point. Therefore, a large range of excitations is desirable to provide the identification algorithm sufficient information about the process' nonlinearity.

Figure 2 shows that P responds in a similar way for any step in s. This gives confidence in that the responses of P are caused by excitations of s, not by process disturbances nor measurement noise. However, as the models are not validated on independent data, it is most important to limit the number of parameters to be identified in order to avoid overfitting. The sampling time of the dataset is 2 seconds.

4 Prediction Error Method

System identification is the science of developing *dynamic* models based on observations of system inputs and system outputs. There exist many different system identification algorithms based on various mathematical approaches. This paper will be restricted to the prediction error method (PEM). PEM is one of the most used system identification approaches. When identifying a model using PEM, the identification algorithm is provided a model structure with one or more unknown parameters to be identified. The unknown parameters are stacked in a parameter vector θ. For example when identifying an ARMAX model with $n_a = 1$, $n_b = 2$, and $n_c = 1$ the parameter vector is

$$\theta = \begin{bmatrix} a_1 \\ b_1 \\ b_2 \\ c_1 \end{bmatrix}. \tag{8}$$

PEM computes the parameter vector that gives the least difference between the real process output and the simulated model output. What exactly is meant by "least difference" must be specified by a mathematical optimization criterion. This paper will consider two different criteria:

$$V(\theta) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{k=1}^{N}\varepsilon_k(\theta)^2, \tag{9}$$

$$W(\theta) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{k=1}^{N}e_k(\theta)^2. \tag{10}$$

When optimizing for ballistic simulations, $V(\theta)$ is used. When optimizing for one-step-ahead predictions, $W(\theta)$ is used. In this paper, unless otherwise specified, models with noise models are optimized with respect to $W(\theta)$, and models without noise models are optimized with respect to $V(\theta)$.

PEM can be used to estimate parameters in both linear and nonlinear model structures. If the model structure is developed based on mechanistic (first principle) modeling of the process, the model is referred to as a grey-box model, as it combines the principles of black-box modeling with process knowledge.

From a mathematical point of view, PEM is an optimization problem of which $V(\theta)$ or $W(\theta)$ is to be minimized with respect to θ. In most cases this optimization problem must be solved iteratively. A poor initial value of θ may cause the optimization algorithm to be trapped in a local minimum. It is therefore desirable for the model developer to find good initial values. ARX models can be identified using the ordinary least squares (OLS) method and are therefore not at the risk of being trapped in a local minimum.

Using system identification, models can in principle be developed without having any knowledge of the process at all, except for datasets of logged process inputs and outputs. However, in practice, basic knowledge of the process greatly helps the model developer during data preprocessing, model identification, and model validation.

When developing models using system identification, one should be aware of the limitation that the model may perform poorly outside the ranges of input and output values of the calibration and validation datasets. Even if the model structure has been developed using mechanistic (first principle) modeling, the model structure is often based on assumptions or simplifications that may not hold for larger ranges of input and output values. Also the estimation error of a parameter (the difference between the "true" parameter value and the estimated parameter value) may have larger impact for input and output values outside the calibration and validation ranges.

5 Data Preprocessing

Datasets logged from real-life processes often need preprocessing before they can be used for empirical modeling and model validation. The dataset to be used in this paper has been preprocessed in two ways: (i) data detrending and (ii) outlier detection.

5.1 Data Detrending

The dataset to be used in this paper has been detrended by subtracting a steady state operating point, $(s_{\text{nom}}, P_{\text{nom}})$, from the raw data. The detrended dataset, $\Delta s = s - s_{\text{nom}}$ and $\Delta P = P - P_{\text{nom}}$, then refers to deviations from the steady state operating point. In particular when identifying linear models, subtracting a steady state operating point is desirable. The resulting linear model is then equivalent to a linearization (first order Taylor expansion) around the operating point.

If a steady state operating point is not known to the model developer, a commonly used approximation is the sample means, i.e. $s_{\text{nom}} = \frac{1}{N}\sum_{k=1}^{N} s_k$ and $P_{\text{nom}} = \frac{1}{N}\sum_{k=1}^{N} P_k$. For this particular dataset, Figure 2 shows that the process is very close to steady state in the time interval $t \in [8700, 10300]$. As the dataset happens to reveal a steady state operating point, this operating point is used for detrending, instead of the sample means. The steady state values found are $s_{\text{nom}} = 45\%$ and $P_{\text{nom}} = 45\text{kW}$. Please note that it is a coincidence that s_{nom} and P_{nom} happen to have the same value in the steady state operating point. This is not the case in general.

5.2 Outlier Detection

The term outlier refers to a sample, or a segment of samples, in the dataset that is not representative for the process' behavior. It is therefore desirable to exclude the outliers from the dataset during model identification and model validation.

It is not trivial to decide which samples that are outliers. It may be intuitive to look for extreme values, i.e. samples of which values are significantly higher or lower than the other samples in the dataset. However, this approach is not reliable. A sample of extreme value may be real, while a non-extreme value may be an outlier. The latter case is shown for real-life industrial data in Komperød et al. (2008).

Deciding whether a sample is an outlier or not is to a large degree a subjective choice which requires practical knowledge of the process to be modeled. However, mathematical algorithms may be used to identify samples that are candidates for being outliers. An approach for outlier detection presented in Komperød et al. (2008) was used at the dataset shown in Figure 2: A high order ARX model with $n_a = 10$ and $n_b = 10$ was identified. Figure 3 shows the residual plot (one-step-ahead prediction errors), e, of this model. In general, samples having large residuals (absolute values)

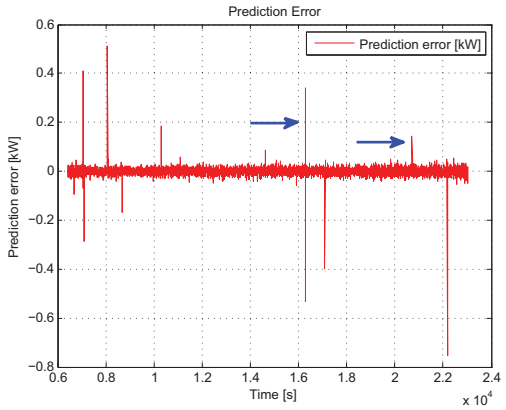


Figure 3: Residual plot of an ARX model with $n_a = 10$ and $n_b = 10$.

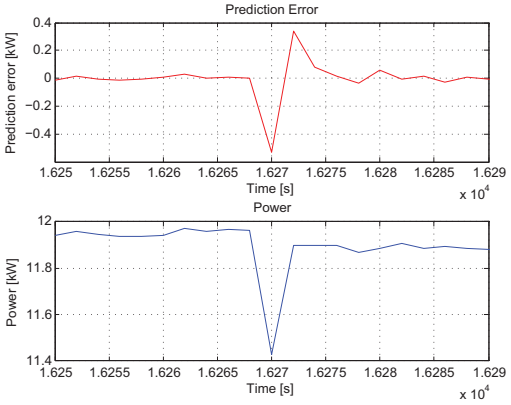


Figure 4: Residuals, e, (upper) and measured power, ΔP, (lower) zoomed in at the left-most arrow in Figure 3.

and their neighbor samples are candidates for being outliers. There are several large residuals in Figure 3. Plotting the residuals in the same plot as the heating element power, P, reveals that most of the large residuals correspond to steps in the heating element power (this plot is not shown). Hence, these residuals are most likely caused by model imperfection, rather than outliers. Only two large residuals do not correspond to power steps. These residuals are marked by blue arrows in Figure 3.

Figure 4 and Figure 5 show the residuals, e, and the measured power, ΔP, zoomed in close to the residuals marked by the left-most arrow and the right-most arrow in Figure 3, respectively. In Figure 4 there is one sample in ΔP (lower subplot), which value is significantly lower than the neighbor samples. This sample value can not be explained by the TRIAC input signal, s. The reason for this sample value is not known for sure. A reasonable explanation is voltage variations on the power grid. A voltage drop of short duration can occur for example when starting an electric motor. Please note from the lower subplot in Figure 4 that the power does not completely restore after the power drop. This observation strengthens the assumption that a larger load on the power grid was activated. The outlier is handled by replacing the extreme value sample by a linear interpolation between the two neighboring samples.

The residual in the upper subplot of Figure 5 is of a different nature than the one in Figure 4. In Figure 5 there are positive residuals for two to three samples, but no negative residual. The lower subplot shows that the power is lifted to a higher level and does *not* go back to the initial level. A reasonable explanation is that a

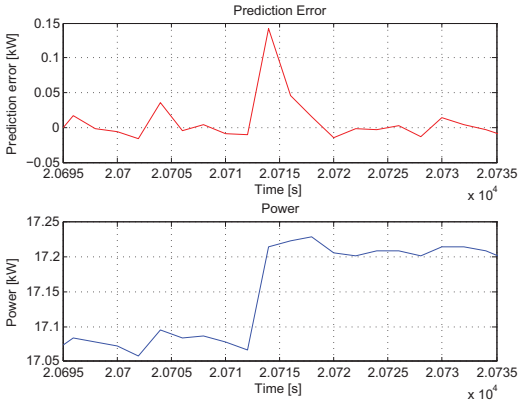


Figure 5: Residuals, e, (upper) and measured power, ΔP, (lower) zoomed in at the right-most arrow in Figure 3.

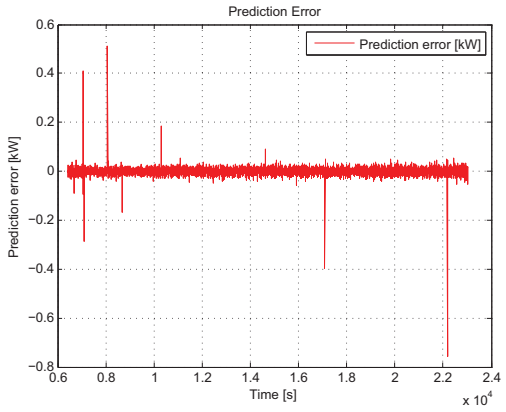


Figure 6: Residual plot after handling outliers.

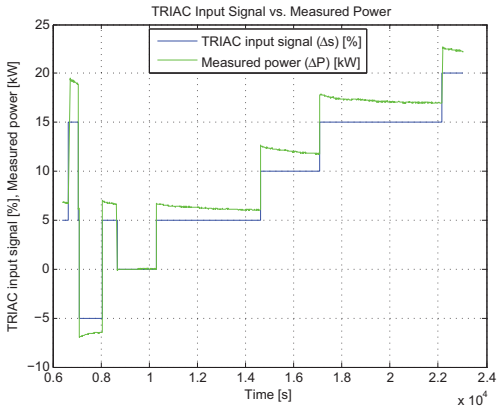


Figure 7: The dataset after being preprocessed. It is now ready to be used for linear and nonlinear system identification.

large load at the power grid was shut down, increasing the grid voltage. This outlier is handled by subtracting 0.15kW from $\Delta P(t) \,\forall\, t \geq 20714$s.

After the outliers were handled, a new ARX model of the same polynomial orders was identified. The residual plot of this model is shown in Figure 6. The figure shows that the residuals marked by blue arrows in Figure 3 are no longer present.

Figure 7 shows the dataset after being preprocessed. This dataset will be used for linear and nonlinear system identification in Section 6 and Section 7, respectively.

6 Linear System Identification

For many processes it is not trivial to decide whether or not the process can be approximated by a linearization within the operating range of interest. It therefore makes sense to try linear models at first, as these models are simpler than nonlinear models. Even if the process is somewhat nonlinear, a linear model may be useful for understanding the process' basic dynamics, such as model order and dominant time constant. Understanding these basic dynamics may be helpful if developing more complex, nonlinear models later.

In this section linear ARX, ARMAX, and OE models are identified. These three model structures have identical input-output model structures, but different noise models. The dataset to be used for system identification is the dataset shown in Figure 7. Before identification, the dataset has been preprocessed as explained in Section 5.

6.1 Deciding Polynomial Orders

A most important choice is how to select the polynomial orders n_a, n_b, n_c, and n_f in ARX, ARMAX, and OE models. In particular when there is no independent dataset available for model validation, the model developer should limit the number of parameters to identify in order to avoid overfitting. The polynomial orders to be used in this paper will now be derived based on the plot in Figure 7. For each positive step in the TRIAC input signal, Δs, the heating element power, ΔP, does an instantaneous positive step and then slowly decreases. The decrement will be approximated as a first order response with negative gain. There is a time delay of one sample from Δs to ΔP. The dynamics from Δs to ΔP are then given on the form

$$\Delta P_k = \left(\underbrace{\tilde{b}_2 q^{-1}}_{\text{step}} + \underbrace{\frac{-\tilde{b}_1 q^{-1}}{1 + a_1 q^{-1}}}_{\text{decrease}} \right) \Delta s_k, \qquad (11)$$

where $\tilde{b}_1$ and $\tilde{b}_2$ are temporary parameters. On the right hand side of (11), the first term represents the instantaneous step and the second term represents the subsequent decrease. Equation (11) can be rewritten as

$$\Delta P_k = \frac{\tilde{b}_2 q^{-1} + \tilde{b}_2 a_1 q^{-2} - \tilde{b}_1 q^{-1}}{1 + a_1 q^{-1}} \Delta s_k. \qquad (12)$$

Introducing $b_1 = \tilde{b}_2 - \tilde{b}_1$ and $b_2 = \tilde{b}_2 a_1$ gives

$$\Delta P_k = \frac{b_1 q^{-1} + b_2 q^{-2}}{1 + a_1 q^{-1}} \Delta s_k. \tag{13}$$

This is the standard form for input-output models of polynomial models. Hence, the chosen polynomial orders for the ARX and ARMAX models are $n_a = 1$ and $n_b = 2$. As n_c of the ARMAX model is not given, this is chosen equal to n_a, i.e. $n_c = n_a = 1$. Equivalent, for the OE model the polynomial orders are $n_b = 2$ and $n_f = 1$. An ARX model, an ARMAX, and an OE model were identified using these polynomial orders. The identifications were performed using the commands `arx`, `armax`, and `oe` of the MATLAB System Identification Toolbox. A time-delay of one sample was specified to the identification algorithms, otherwise the default settings were used.

6.2 Model Validation and Discussions

Figure 8 shows ballistic simulations of the identified ARX, ARMAX, and OE models. In general, the models give small simulation errors close to the operating point $(s_{\text{nom}}, P_{\text{nom}})$, but give a somewhat poorer fit otherwise. Notice in particular the last step at $t = 22170$s: Right before the step all the model simulations are significantly below the measured value. Right after the step all the simulations are significantly above the measured value. However, these simple linear models catch the basic dynamics of the process. The models may be usable for some applications, such as model-based PID controller tuning provided that the gain and phase margins are chosen sufficient large.

To examine whether the input-output model structure of (13) has sufficient polynomial degrees, n_a, n_b, n_c, and n_f were all increased by one to examine whether this improved the models' performance. Table 2 shows the values of the optimization criterion $V(\theta)$ of (9) for the models used in Figure 8 and for models with higher polynomial orders. The table reveals two interesting results: (i) According to $V(\theta)$, the OE models perform much better than the ARX and ARMAX models. This is to be expected as OE is optimized for $V(\theta)$ (ballistic simulation), while ARX and ARMAX are optimized for $W(\theta)$ (one-step-ahead prediction). (ii) Increasing the polynomial orders does not significantly improve the optimization criterion. Hence, it is concluded that the polynomial orders of (13) are sufficient. For the ARX and ARMAX models, increasing the polynomial orders actually gives slightly poorer performance. In the succeeding text, only the models of lower polynomial orders, i.e. rows 1, 3, and 5 of Table 2, will be considered.

Figure 9 shows the ballistic simulation errors, $\varepsilon(\theta)$, of the ARX, ARMAX, and OE models, i.e. the difference

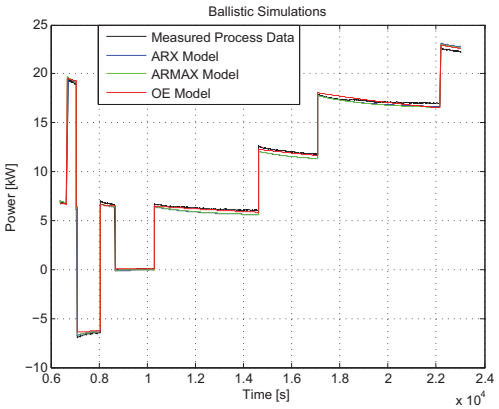


Figure 8: Ballistic simulations using the ARX model $(n_a = 1, n_b = 2)$, ARMAX model $(n_a = 1, n_b = 2, n_c = 1)$, and OE model $(n_b = 2, n_f = 1)$.

Table 2: Optimization criterion for various model structures and polynomial orders.

Model	n_a	n_b	n_c	n_f	$V(\theta)$
ARX	1	2	-	-	0.106
ARX	2	3	-	-	0.109
ARMAX	1	2	1	-	0.108
ARMAX	2	3	2	-	0.110
OE	-	2	-	1	0.043
OE	-	3	-	2	0.042

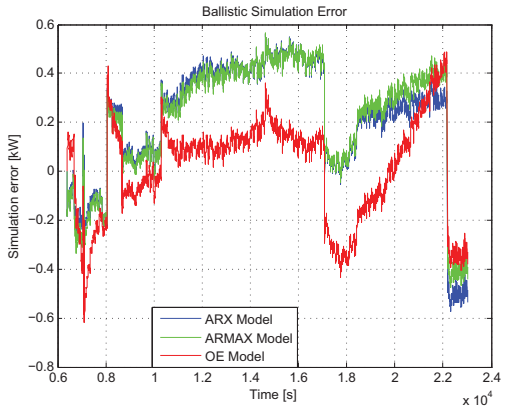


Figure 9: Ballistic simulations error using the ARX, ARMAX, and OE models.

between the measured data and the simulated data of Figure 8. A horizontal curve in Figure 9 indicates that the simulated data increase or decrease at the same rate as the measured data. An increasing curve shows that the simulated data increase too slowly or decrease too fast.

For all three models there are significant steps in Figure 9 at the steps in Δs. The most interesting result of Figure 9 is that the simulation error of the OE model is in general closer to zero than the other models, i.e. has better fit according to the criterion $V(\theta)$, but at the same time has a steeper curve in large parts of Figure 9. In particular in the time intervals $t \in [7116, 8076]$ and $t \in [17098, 22170]$ the OE model has a very steep curve compared to the other models. Hence, it seems that the ARX and ARMAX models explain the *dynamics* of the process more accurately than the OE model, but still give a poorer performance at ballistic simulation. During one-step-ahead predictions, which ARX and ARMAX are optimized for, the noise models handle the offsets that are caused by inaccurate explanation of the steps. However, as the noise models are to no avail during ballistic simulations, the good explanations of the dynamics do not fully compensate for the poorer explanations of the steps. The OE model on the other hand is optimized to balance these two considerations during ballistic simulations.

For the purpose of model-based tuning of a controller that should have a constant setpoint, i.e. constant operating point, it seems reasonable to recommend the ARX and ARMAX models over the OE model, as the ARX and ARMAX models seem to explain the dynamic more accurately.

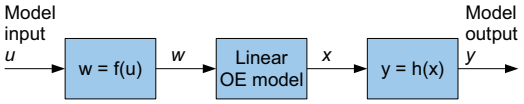


Figure 10: The Hammerstein-Wiener model structure as defined in the MATLAB System Identification Toolbox. The model structure has no noise model.

7 Nonlinear System Identification

Linear process models of the dynamics from Δs to ΔP were developed in Section 6. These models seem to explain the process dynamics well close to an operating point, but handle steps in Δs somewhat poorly.

This section will focus on nonlinear system identification of the dynamics from Δs to ΔP. The motivation for using nonlinear system identification is that the simulation errors of the linear models indicate that the process is nonlinear. The dataset to be used is shown in Figure 7. The dataset has been preprocessed as described in Section 5. As there is no independent dataset available for validation, an important consideration is to limit the number of parameters to be identified in order to avoid overfitting.

7.1 The Hammerstein-Wiener Model Structure

There are several modeling approaches and model structures that can be used for nonlinear system identification. In this paper a model structure referred to as Hammerstein-Wiener (HW) will be used. The motivations for choosing the HW model structure are: (i) The model structure is very simple and easy to understand. (ii) Linear system theory can be applied to analyze the models' stability properties.

The HW model structure is something between linear and nonlinear models. The core of the HW model is a linear model. As HW models are defined in the MATLAB System Identification Toolbox, the linear model is an OE model. The input to the OE model is processed by a static, nonlinear function $f(u)$. The output from the OE model is processed by another static, nonlinear function $h(x)$. The HW model structure is illustrated in Figure 10. A model having nonlinear processing of the input only, not the output, is referred to as a Hammerstein model. A model having nonlinear processing of the output only, not the input, is referred to as a Wiener model (Ljung, 1999, 2009).

Going from linear to nonlinear system identification gives new opportunities, but also introduces new challenges. General nonlinear model structures, such as

the HW structure, are more flexible than linear model structures. The flexibility comes with the price of several parameters to be identified and several choices where the model developer can go wrong.

For a linear OE model, the model developer's choices are the polynomial orders, i.e. n_b and n_f. When expanding the linear OE model to a HW model, the model developer must also choose which nonlinear functions $f(u)$ and $h(x)$ to use. Typically, $f(u)$ and $h(x)$ have parameters to be identified, for example saturation limits or polynomial coefficients. These parameters are added to the parameter vector θ, which was introduced in Section 4. When $f(u)$, $h(x)$, n_b, and n_f are chosen, θ is identified using PEM.

7.2 Default Settings Fail

In the MATLAB System Identification Toolbox, HW models can be identified either from the command line or by using the graphical user interface (GUI) of the toolbox. Using the command line, the model developer is forced to make explicit choices for $f(u)$, $h(x)$, n_b, and n_f. Using the GUI, default settings are provided. Deciding $f(u)$, $h(x)$, n_b, and n_f are most important, but difficult, choices. It therefore may be tempting for an inexperienced model developer to keep the default settings of the GUI.

Figure 11 shows ballistic simulation of a HW model, which was identified using the default settings for $f(u)$, $h(x)$, n_b, and n_f in the toolbox GUI. Also with respect to the optimization algorithm the default settings were used. Figure 11 speaks clearly for itself: The default settings give a very poor model. Even though this model structure has a large number of parameters to be identified and was validated at the same dataset that was used for identification, the model fit is very disappointing.

The example presented in Figure 11 illustrates what was explained above: The increased flexibility of the general nonlinear model structures also has significant disadvantages. In case of HW models, the model developer has to make clever choices for $f(u)$, $h(x)$, n_b, and n_f for the modeling work to be successful. Also, as the number of parameters to be identified increases, the risk of the optimization algorithm to be trapped in a local minimum increases significantly. Hence, the model developer may also spend some effort to provide the algorithm good initial values for the parameters to be identified.

7.3 Deciding Input and Output Nonlinearities

This subsection shows how examination of the dataset, together with the linear models developed in Section 6,

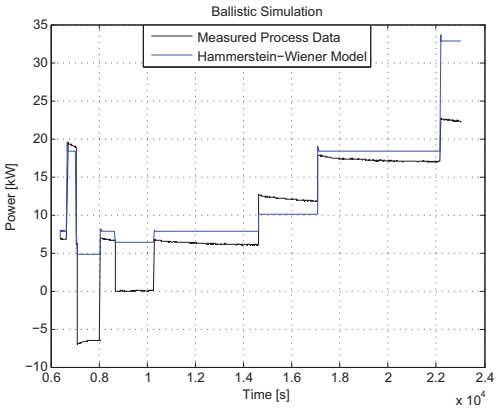


Figure 11: Ballistic simulation using the Hammerstein-Wiener model, which was identified using the default $f(u)$, $h(x)$, n_b, and n_f of the GUI in the MATLAB System Identification Toolbox.

can be used to make good choices for $f(u)$, $h(x)$, n_b, and n_f when developing a HW model for the dynamics from Δs to ΔP.

The preprocessed dataset used for system identification is shown in Figure 7. After $t = 1.0 \times 10^4$ s, four steps of Δs occur. For each step, Δs is increased by 5%. At first sight it seems that ΔP responds similarly to each step. However, more accurate examination of the dataset reveals that the magnitudes of the ΔP steps are different for each Δs step, even though all Δs steps are of the same magnitude.

To simplify notation, r [kW/%] is defined as the magnitude of a ΔP step divided by the magnitude of the corresponding Δs step. Hence, r can be considered the gain of which a step in Δs is amplified to the corresponding step in ΔP. Please note that the *instantaneous* responses in ΔP are considered, not the steady state values. Table 3 summarizes r for each step in Δs after $t = 1.0 \times 10^4$ s. The table shows that the higher values Δs and ΔP have prior to the step, the smaller is r.

In Section 6 it was concluded that the input-output models of the ARX and ARMAX models explain the *dynamics* of the process well, even though they handle the steps in Δs somewhat poorly. It is therefore reasonable to base the linear OE block of the HW model on the input-output model of the ARX model or the ARMAX model. Hence, the polynomial orders of the linear OE block is chosen $n_b = 2$ and $n_f = 1$. For the same reason, it is also reasonable to assume that the polynomial coefficients of $A(q)$ and $B(q)$ in the ARX

Table 3: Gain, r, for each step in Δs after $t = 1.0 \times 10^4$ s.

Step Δs [%]	Step ΔP [kW]	Gain, r, [kW/%]
0.00→ 5.00	0.04→ 6.78	1.35
5.00→10.00	6.09→ 12.64	1.31
10.00→15.00	11.72→ 17.87	1.23
15.00→20.00	16.99→ 22.68	1.14

and ARMAX models are suitable as initial values for identification of $F(q)$ and $B(q)$ in the linear OE block. The ARX model is chosen for initial values, because the ARX model has the simplest model structure and can be identified using the ordinary least squares (OLS) method.

The linear OE block of the HW model can not handle the variable gain shown in Table 3. The variable gain has to be handled by the input nonlinearity $f(u)$ and/or the output nonlinearity $h(x)$. A reasonable approach is to consider the gain, r, as a function of Δs and/or ΔP. It is here chosen to consider r as a function of Δs, i.e. $r = r(\Delta s)$. This choice will lead to a Hammerstein model (the reason for this choice is explained in Subsection 7.6). That is, $f(u)$ will be used to handle the nonlinearities, and $h(x)$ will simply be $h(x) = x$, or equivalent, the $h(x)$ block in Figure 10 will be absent.

A polynomial is a reasonable choice for explaining r as a function of Δs. As there are four $(\Delta s, r)$ pairs in Table 3, a third order polynomial will give no residual. However, as it is important to keep the number of parameters low, a first order polynomial is chosen. Hence, r is written as

$$r(\Delta s) = d_2 \Delta s + d_1, \tag{14}$$

where d_1 and d_2 are the polynomial coefficients. The process' behavior at each step, ignoring the general process dynamics, can then be approximated as

$$\begin{aligned} \Delta P_k &= \Delta P_{k-1} + r(\Delta s_{k-2}) \times (\Delta s_{k-1} - \Delta s_{k-2}) \\ &= \Delta P_{k-1} + (d_2 \Delta s_{k-2} + d_1) \times (\Delta s_{k-1} - \Delta s_{k-2}) \end{aligned} \tag{15}$$

for a step that occurs in Δs at timestep $k-1$. The response in ΔP is delayed by one sample. The last term in (15) can be rewritten as

$$\begin{aligned} &(d_2 \Delta s_{k-2} + d_1) \times (\Delta s_{k-1} - \Delta s_{k-2}) \\ &= d_2 \Delta s_{k-1} \Delta s_{k-2} + d_1 \Delta s_{k-1} - d_2 {\Delta s_{k-2}}^2 - d_1 \Delta s_{k-2}. \end{aligned} \tag{16}$$

This equation gives a hint of how $f(u)$ could be chosen: The term $\Delta s_{k-1} \Delta s_{k-2}$ can not be included in $f(u)$, because $f(u)$, according to the definition of the HW model structure, should be a *static* function of u. Hence, one of these approximations must be chosen: $\Delta s_{k-1} \Delta s_{k-2} \approx {\Delta s_{k-2}}^2$ or $\Delta s_{k-1} \Delta s_{k-2} \approx {\Delta s_{k-1}}^2$. Using the first approximation, the term $d_2 {\Delta s_{k-2}}^2$ will cancel in (15). Hence, $r(\Delta s_{k-2})$ will be equal to the constant d_1, which is in conflict with the intention of choosing $r(\Delta s)$ as a function of Δs. Using the other approximation, $\Delta s_{k-1} \Delta s_{k-2} \approx {\Delta s_{k-1}}^2$, makes sense. Define $f(\Delta s)$ as

$$f(\Delta s) \stackrel{\text{def}}{=} d_2 \Delta s^2 + d_1 \Delta s. \tag{17}$$

Using this definition and this approximation, (15) can be rewritten as

$$\Delta P_k = \Delta P_{k-1} + f(\Delta s_{k-1}) - f(\Delta s_{k-2}). \tag{18}$$

Although this derivation is based on some rough approximations, it will be shown shortly that $f(\Delta s)$ as defined in (17) is a good choice for the Hammerstein model to be identified.

For the chosen polynomial orders of the linear OE block, i.e. $n_b = 2$ and $n_f = 1$, the model structure to be used in this block is on the form

$$\Delta P_k + f_1 \Delta P_{k-1} = b_1 w_{k-1} + b_2 w_{k-2}, \tag{19}$$

where w is the output from the input nonlinearity block as illustrated in Figure 10. In order to handle the process' nonlinearity, the OE model of (19) is extended to a Hammerstein model by replacing w_{k-1} and w_{k-2} by $f(\Delta s_{k-1})$ and $f(\Delta s_{k-2})$, respectively. The Hammerstein model is then written as

$$\begin{aligned} \Delta P_k + f_1 \Delta P_{k-1} &= b_1 f(\Delta s_{k-1}) + b_2 f(\Delta s_{k-2}) \\ &= b_1 (d_2 {\Delta s_{k-1}}^2 + d_1 \Delta s_{k-1}) \\ &\quad + b_2 (d_2 {\Delta s_{k-2}}^2 + d_1 \Delta s_{k-2}). \end{aligned} \tag{20}$$

Now the desired model structure for the Hammerstein model has been developed. Subsections 7.4 and 7.5 consider how to identify the model parameters.

7.4 Computing Initial Values

For most model structures the PEM method uses an iterative optimization algorithm to compute the parameter vector θ. Such algorithms are generally at the risk of being trapped in a local minimum. The model developer may try to find good initial values for the

parameters to be identified, or he can try the iterative algorithm directly, hoping it will not be trapped in a local minimum. This subsection presents a suggestion for how to estimate good initial values for the parameters of the model (20).

The first issue is to estimate d_1 and d_2. Equation (20) can be written on vector form as

$$\begin{aligned} &\Delta P_k + f_1 \Delta P_{k-1} \\ &= \begin{bmatrix} b_1 \Delta s_{k-1}{}^2 + b_2 \Delta s_{k-2}{}^2 \\ b_1 \Delta s_{k-1} + b_2 \Delta s_{k-2} \end{bmatrix}^T \begin{bmatrix} d_2 \\ d_1 \end{bmatrix}, \end{aligned} \tag{21}$$

where b_1, b_2, and f_1 are the parameters identified in the linear ARX model in Section 6 (a_1 of the ARX model corresponds to f_1 in (21)).

For each row in Table 3, a row in (21) is defined. This gives a linear, over-determined set of four equations with two unknown. Equation (21) is then solved for $[d_2 d_1]^T$ using the ordinary least squares (OLS) method. Figure 12 shows ballistic simulation of the Hammerstein model of (21) compared to the ARX model identified in Section 6. The Hammerstein model performs significantly better than the ARX model in terms of the optimization criterion $V(\theta)$. The Hammerstein model has $V(\theta) = 0.045$, while the ARX model has $V(\theta) = 0.106$. As the polynomial coefficients b_1, b_2, and f_1 (a_1 for the ARX model) are identical for the ARX model and the Hammerstein model, the improvement of the Hammerstein model can only be explained by the replacement of Δs in favor of $f(\Delta s)$ and the estimation of d_1 and d_2. The Hammerstein model gives confidence in that the chosen model structure is well suited for explaining the process' behavior.

As good estimates of d_1 and d_2 are available, the parameters b_1, b_2, and f_1 will now be re-identified. That is, d_1 and d_2 in (20) are considered fixed, and b_1, b_2, and f_1 are identified using linear system identification. During this identification step, the system input is $f(\Delta s) = d_2 \Delta s^2 + d_1 \Delta s$ and the system output is ΔP. Please note that even if linear system identification is used, the identified model is a Hammerstein model, because the input to the linear model is $f(\Delta s)$, not Δs. Figure 13 shows Hammerstein models optimized this way using ARX and OE, respectively. These optimizations give significant improvements compared to the Hammerstein model of Figure 12. The optimization criterion is $V(\theta) = 0.024$ for the ARX-optimized model and $V(\theta) = 0.008$ for the OE-optimized model.

Although the OE-optimized model gives by far the best $V(\theta)$, the ARX-optimized model has a significant advantage: As ARX models can be identified using the ordinary least squares (OLS) method, all steps in identification of the ARX-optimized model can be solved

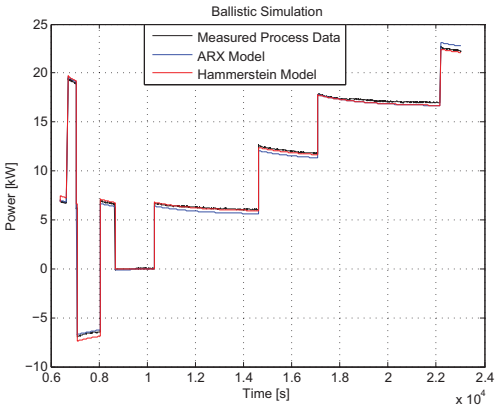


Figure 12: Ballistic simulation of the Hammerstein model of (21) after estimating d_1 and d_2. The parameters b_1, b_2, and f_1 are identical to the ARX model identified in Section 6. The model is compared to measured process data and the ARX model.

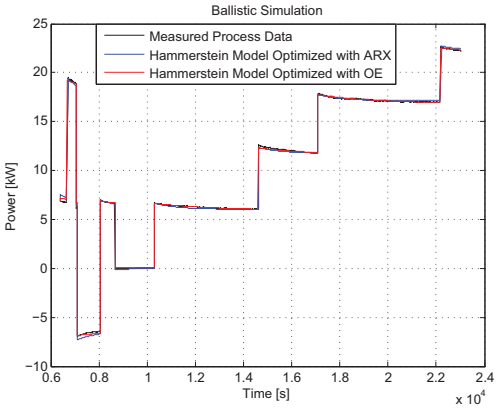


Figure 13: Ballistic simulations of Hammerstein models that are optimized with ARX and OE, respectively.

using OLS, involving no iterative optimization algorithms that are at the risk of being trapped in a local minimum. This also means that the ARX-optimized Hammerstein model can be identified by model developers not having the MATLAB System Identification Toolbox or other optimization software.

Figure 13 and the optimization criterion $V(\theta)$ show that the OE-optimized Hammerstein model fits the measured data very well. Hence, finding good initial values to be used in an iterative optimization algorithm has been successful.

7.5 Final Estimation

The final step of the nonlinear system identification work is to estimate all the parameters simultaneously using an iterative optimization algorithm. In Subsection 7.4, good initial values for the optimization algorithm were estimated.

A reasonable approach for identifying the final Hammerstein model is to use the function `nlhw` in the MATLAB System Identification Toolbox. Using this function, it can be specified that $f(\Delta s)$ should be a second order polynomial, and that there should be no output nonlinearity. Also n_b and n_f can be specified. However, to the authors' knowledge, there is no way to force the constant term of the second order polynomial in $f(\Delta s)$ to be zero. In order words: $f(\Delta s)$ will be on the form $f(\Delta s) = d_2\Delta s^2 + d_1\Delta s + d_0$, where d_0 is a constant. Hence, an additional parameter, d_0, has been introduced. This is unfortunate with respect to avoid overfitting.

A tailor-made workaround, which avoids the undesirable parameter d_0, will be presented shortly. However, the first consideration is whether the model structure of (20) has the smallest possible number of parameters to be identified. The following parameters are defined

$$\tilde{b}_2 \stackrel{\text{def}}{=} \frac{b_2}{b_1}, \tag{22}$$

$$\tilde{d}_1 \stackrel{\text{def}}{=} b_1 d_1, \tag{23}$$

$$\tilde{d}_2 \stackrel{\text{def}}{=} b_1 d_2. \tag{24}$$

Using these definitions, (20) can be rewritten as

$$\begin{aligned}\Delta P_k + f_1\Delta P_{k-1} = \tilde{d}_2\Delta {s_{k-1}}^2 + \tilde{d}_1\Delta s_{k-1} \\ + \tilde{b}_2(\tilde{d}_2\Delta {s_{k-2}}^2 + \tilde{d}_1\Delta s_{k-2}).\end{aligned} \tag{25}$$

Hence, (20) can be rewritten with one less parameter. The parameters that are to be identified are $\tilde{b}_2$, $\tilde{d}_1$, $\tilde{d}_2$, and f_1.

The simplest solution the authors have found to identify the parameters in exactly the form of (25) is to use the functions `idgrey` and `pem` in the MATLAB System Identification Toolbox. These functions allow identification of arbitrary parameters in a linear state space model. Please refer to Ljung (2009) for a detailed explanation of these functions. Equation (25) can be rewritten to a discrete time state space model as

$$\begin{aligned}\begin{bmatrix} x^1_{k+1} \\ x^2_{k+1} \end{bmatrix} &= \begin{bmatrix} -f_1 & \tilde{b}_2 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x^1_k \\ x^2_k \end{bmatrix} \\ &\quad + \begin{bmatrix} \tilde{d}_2 & \tilde{d}_1 \\ \tilde{d}_2 & \tilde{d}_1 \end{bmatrix}\begin{bmatrix} {\Delta s_k}^2 \\ \Delta s_k \end{bmatrix}, \\ \Delta P_k &= \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x^1_k \\ x^2_k \end{bmatrix} + \varepsilon_k.\end{aligned} \tag{26}$$

In (26) the ballistic simulation error, ε, is added to emphasize that the model has no noise model. When the parameters $\tilde{b}_2$, $\tilde{d}_1$, $\tilde{d}_2$, and f_1 have been identified, the model can be rewritten to a Hammerstein model with an OE model as its linear block

$$\begin{aligned}\Delta P_k + f_1\Delta P_{k-1} = f(\Delta s_{k-1}) + \tilde{b}_2 f(\Delta s_{k-2}) \\ + \varepsilon_k,\end{aligned} \tag{27}$$

where

$$f(\Delta s) = \tilde{d}_2\Delta s^2 + \tilde{d}_1\Delta s. \tag{28}$$

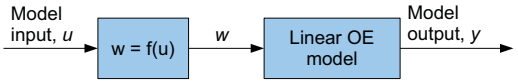


Figure 14: Block diagram of the final Hammerstein model.

Figure 14 shows a block diagram of this final Hammerstein model.

The ballistic simulation of the final model is shown in Figure 15, and the ballistic simulation error is shown in Figure 16. The optimization criterion is $V(\theta) = 0.007$. This is only a slight improvement over the OE-optimized model of Figure 13. The optimization algorithm used only three iterations to reach the final model. Hence the parameters of the OE-optimized model must have been very good initial values.

It was also tested whether poorer initial values led to the same parameters. When all initial values were set to zero, the optimization algorithm finds the same parameters as for the good initial values. Hence, in this case the work of estimating good initial values did not improve the final model. However, it is interesting

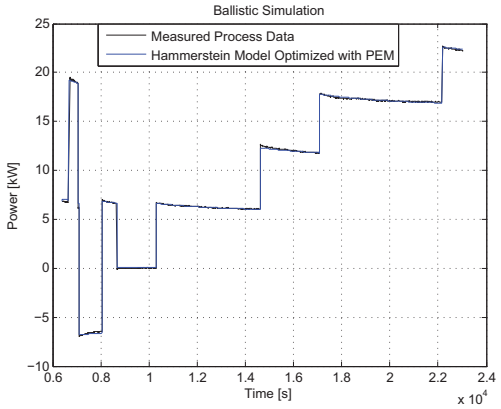


Figure 15: Ballistic simulation of the final Hammerstein model.

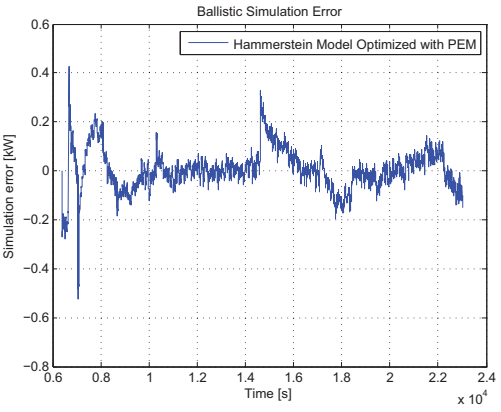


Figure 16: Ballistic simulation error of the final Hammerstein model.

to notice that the optimization algorithm used three iterations when having good initial values, and 17 iterations when having zeros as initial values.

Comparing the final Hammerstein model with the linear OE model identified in Section 6 shows that $V(\theta)$ was reduced by a factor of six when extending the linear OE model to a Hammerstein model. This extension includes only one additional parameter (the number of parameters to be identified is increased from three to four).

7.6 Noise Model

The HW model structure as defined in the MATLAB System Identification Toolbox has no noise model. This is reasonable as the output nonlinearity would significantly complicate the noise model, because there would be nonlinear dynamics from the one-step-ahead prediction error, e, to the model output, y.

The lack of noise model makes the HW model structure less suitable for some applications, such as (i) state estimation, (ii) measurement noise filtering using Kalman filter, and (iii) model predictive control (MPC).

In Subsection 7.3 the authors made the explicit choice of explaining the process nonlinearity using the input nonlinearity $f(u)$, instead of the output nonlinearity $h(x)$. This choice kept the door open to later replace the linear OE block of Figure 14 with a linear model structure having a noise model. By not applying a nonlinearity at the model output, the dynamics from the prediction error, e, to the model output, y, will be linear.

The state space model of (26) can easily be extended with a noise model

$$\begin{aligned}\begin{bmatrix} x^1_{k+1} \\ x^2_{k+1} \end{bmatrix} &= \begin{bmatrix} -a_1 & \tilde{b}_2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x^1_k \\ x^2_k \end{bmatrix} \\ &\quad + \begin{bmatrix} \tilde{d}_2 & \tilde{d}_1 \\ \tilde{d}_2 & \tilde{d}_1 \end{bmatrix} \begin{bmatrix} \Delta s_k{}^2 \\ \Delta s_k \end{bmatrix} \\ &\quad + \begin{bmatrix} k_1 \\ 0 \end{bmatrix} e_k, \\ \Delta P_k &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x^1_k \\ x^2_k \end{bmatrix} + e_k,\end{aligned} \tag{29}$$

where k_1 is the Kalman filter gain. The parameters a_1, $\tilde{b}_2$, $\tilde{d}_1$, $\tilde{d}_2$, and k_1 of (29) were identified using `idgrey` and `pem`. The parameter f_1 of (26) has been replaced by a_1, because (29) can be rewritten to a Hammerstein model with an ARMAX model as it linear block

$$\begin{aligned}\Delta P_k + a_1 \Delta P_{k-1} = f(\Delta s_{k-1}) + \tilde{b}_2 f(\Delta s_{k-2}) \\ + e_k + c_1 e_{k-1},\end{aligned} \tag{30}$$

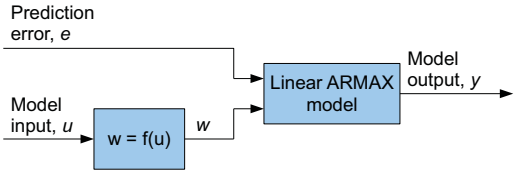


Figure 17: Block diagram of the final Hammerstein model extended with a noise model.

where

$$f(\Delta s) = \tilde{d}_2 \Delta s^2 + \tilde{d}_1 \Delta s \tag{31}$$

and

$$c_1 = a_1 + k_1. \tag{32}$$

Figure 17 shows a block diagram of the Hammerstein model of (30). This Hammerstein model has $V(\theta) = 0.032$, which is significantly poorer than the model of (27). This is because the model of (30) is optimized for $W(\theta)$ (one-step-ahead prediction), while the model of (27) is optimized for $V(\theta)$ (ballistic simulation). This result is similar to the result of Table 2, where the ARX and ARMAX models perform poorer than the linear OE model at ballistic simulation.

Figure 18 shows the one-step-ahead prediction error, e, for the linear ARMAX model identified in Section 6 with $n_a = 1$, $n_b = 2$, and $n_c = 1$ (upper subplot) and for the Hammerstein model of (30) (lower subplot). The figure shows that the largest peaks are considerably smaller for the Hammerstein model. The one-step-ahead prediction optimization criterion is $W(\theta) = 3.1 \times 10^{-4}$ for the ARMAX model and $W(\theta) = 1.8 \times 10^{-4}$ for the Hammerstein model, i.e. 42% lower for the Hammerstein model.

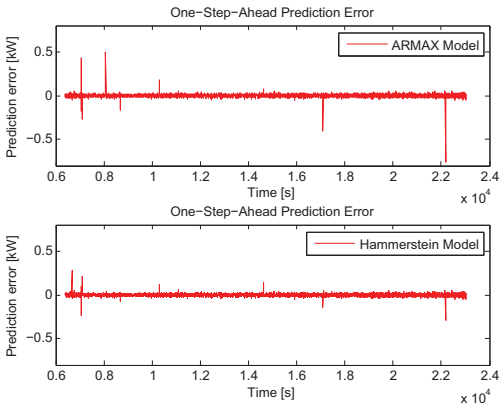


Figure 18: One-step-ahead prediction error, e, of the linear ARMAX model identified in Section 6 (upper) and the Hammerstein model of (30) (lower).

8 Model Weaknesses

The Hammerstein model of (27) gives very good model fit to the identification dataset, even with a small number of identified parameters. This strongly indicates that the identified model explains the process' true input-output behavior. If the measured power, P, was significantly influenced by random process disturbances and measurement noise, it is very unlikely that a model with few parameters would give a good fit, even to the identification dataset.

Even though the model explains the process input-output behavior well for the dataset used in this paper, it is not known whether the model will perform this good on datasets with a significantly different frequency content in the input signal. Further, it is not known whether the model will perform well on datasets of which Δs is significantly higher or lower than in the dataset used in this paper.

9 Further Research

The Hammerstein model identified in this paper is a black-box model in the sense that it explains *how* the measured power responds to the TRIAC input signal, but not *why* this happen. An interesting continuation of this work is to develop a grey-box model of the dynamics from the TRIAC input signal to the power. A grey-box model is a mechanistic model of which unknown parameters are estimated using PEM or other approaches for black-box modeling.

The modeling work presented in this paper is part of a larger modeling work, which aims to model the dynamics of the most important input / output pairs of the CZ process using black-box and grey-box modeling. Next these models are to be used for improving monitoring and control of the CZ process.

10 Conclusions

This paper considered empirical modeling of the dynamics from the TRIAC input signal to the actual (measured) heating element power at the Czochralski (CZ) crystallization process. The modeling was based

on a dataset logged from a real-life CZ process. Before modeling, the dataset was preprocessed by detrending the data and handling outliers. As no independent dataset was available for model validation, it was considered most important to limit the number of parameters to be identified.

Initially three linear polynomial models were identified; an ARX model, an ARMAX model, and an output error (OE) model. These models explain the process dynamics well close to the operating point, but perform somewhat poorer otherwise.

Because the linear models do not explain the process behavior very well over a larger range of the input signal, nonlinear system identification was used for modeling the process. The Hammerstein-Wiener (HW) model structure was chosen. At first, a model was identified using the default setting for HW models in the MATLAB System Identification Toolbox GUI. However, this model has a very poor performance, and was therefore rejected.

As the default settings of the toolbox failed, the model structure was developed by examination of the process' behavior at steps in the input signal. Also the results from linear system identification were used. The chosen model structure is a Hammerstein model, which has a second order polynomial as input nonlinearity and no nonlinearity at the output. The latter choice was done to allow a noise model to be added to the model.

The Hammerstein model has four parameters to be identified, which is only one more than the linear OE model. Still the Hammerstein model has improved the optimization criterion (mean squared ballistic simulation errors) by a factor of six compared to the linear OE model. As the Hammerstein model has good model fit despite few identified parameters, it is concluded that the modeling work has been successful.

This paper also extended the Hammerstein model with a noise model. For one-step-ahead predictions, the optimization criterion (mean squared one-step-ahead prediction errors) is reduced by 42% for the extended Hammerstein model compared to the linear ARMAX model. The extended Hammerstein model has five parameters to be identified, compared to four parameters in the ARMAX model.

Acknowledgements

The authors are most grateful to SINTEF Materials and Chemistry in Trondheim, Norway for giving access to data from the company's CZ process, and for allowing these data to be used in this paper and other publications. John Atle Bones at SINTEF is acknowledged for his decisive contributions in performing the experiments on the CZ process. Eivind Johannes Øvrelid at SINTEF is acknowledged for sharing his knowledge and experience of the CZ process.

The first author is most grateful for the financial support of his PhD study from NorSun AS, Østfold Energi AS, and the Norwegian Research Council. The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is also very much appreciated.

References

Komperød, M., Hauge, T., and Lie, B. Preprocessing of experimental data for use in model building and model validation. In *The 49th Scandinavian Conference on Simulation and Modeling (SIMS 2008)*. Oslo, Norway, 2008.

Lee, K., Lee, D., Park, J., and Lee, M. MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 2005. 3:252–257.

Ljung, L. *System Identification - Theory for the User, 2nd Edition*. Prentice-Hall, Upper Saddle River, New Jersey, 1999.

Ljung, L. *System Identification Toolbox 7 - User's Guide*. The MathWorks, Inc., 2009.

Considerations of Paper D

In the article, the process input, s, is scaled in percent (%). When considering changes in the input, a change from 45% to 50% is referred to as a change of 5%. However, the correct size is 5 percentage points.

Paper E

Adaptive System Identification of Heating Element Power for the Czochralski Crystallization Process

The candidate presented this paper at the 51st Conference on Simulation and Modelling (SIMS 2010). The paper is also included in the conference's proceedings. The conference was held October 14th-15th 2010 in Oulu, Finland. The research presented in this paper is based on logged process data from an experiment at the Czochralski (CZ) crystallization process at SINTEF Materials and Chemistry in Trondheim, Norway. The candidate did not participate in this experiment. The candidate has done all the research presented in the paper and written the paper during his PhD study.

This paper and Paper D present modeling work on the same part of the CZ process. However, different modeling approaches are used in the two publications.

SIMS 2010 Proceedings
The 51st Conference on Simulation and Modelling
14-15 October 2010 Oulu, Finland

Editor: Esko Juuso, University of Oulu

ISBN 978-952-5183-42-9
http://www.automaatioseura.fi

Adaptive System Identification of Heating Element Power for the Czochralski Crystallization Process ⋆

Magnus Komperød * **John Atle Bones** ** **Bernt Lie** ***

* *Østfold University College, Faculty of Engineering, 1757 Halden, Norway (e-mail: magnus.komperod@hiof.no).*
** *SINTEF Materials and Chemistry, Department of Metallurgy, 7465 Trondheim, Norway (e-mail: johnatle.bones@sintef.no).*
*** *Telemark University College, Faculty of Technology, 3901 Porsgrunn, Norway (e-mail: bernt.lie@hit.no).*

Abstract: The Czochralski (CZ) crystallization process is used to produce monocrystalline silicon, which is used in solar cells and electronics. This paper considers adaptive system identification of the dynamics from the triode for alternating current (TRIAC) to the actual (measured) heating element power at the CZ process. The system identification work is based on a dataset logged at a real-life CZ process.
Two approaches for adaptive system identification are considered: (i) The MATLAB command `rarmax` is used to tune all parameters of an ARMAX model. (ii) Only the gain of the model is tuned, while the pole, the zero, and the noise model are fixed. The adaptive gain can be computed using the recursive least squares method, which is simple to implement and has numerical advantages.
These adaptive system identification approaches are compared to a non-adaptive ARMAX model. The performance criterion used is the mean squared one-step-ahead prediction error. The adaptive gain model performs much better than the non-adaptive model and the adaptive model which tunes all parameters. There are also other reasons why the adaptive gain approach is favorable.

Keywords: Adaptive system identification, Czochralski crystallization process, Heating element power, Recursive ARMAX, Recursive least squares method.

1. INTRODUCTION

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials, i.e. materials that have homogeneous crystal structures. Among the most important applications is production of monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. The CZ process is a batch process.

System identification is the science of developing *dynamic* process models based on observations of the process. The identified models explain the process outputs as mathematical functions of the process inputs. Adaptive (recursive) system identification tunes the model parameters online to adapt the model to slowly varying process parameters.

The contribution of this paper is to model the heating element power of the CZ process using adaptive system identification. The modeling work is based on a dataset logged at a real-life CZ process at SINTEF Materials and Chemistry in Trondheim, Norway.

Process models serve several purposes. Process models can be used to tune PID controllers. Adaptive models can be used for adaptive controller tuning. If more advanced model-based control strategies are to be applied, such as model predictive control (MPC), process models are essential. Examining parameter variations during adaptive system identification may reveal useful information of the process.

Lan (2004) gives an introduction to crystal growth, including the CZ process, and provides an extensive number of references for further reading. The literature of system identification is extensive. Among the most well-known books is Ljung (1999). This book gives a comprehensive introduction to system identification. Lee et al. (2005) presents an approach for batch-to-batch optimization of the CZ process, which includes model-based control. The paper includes two simple dynamic models empirically developed from step responses. However, these models cover different parts of the CZ process than the present paper. Except for Lee et al. (2005), the authors of the present paper have not been successful in finding any publications that present results within empirical modeling of the CZ

⋆ This paper is based on a dataset that SINTEF Materials and Chemistry in Trondheim, Norway, most kindly has provided the authors. The first author's PhD study receives financial support from NorSun AS, Østfold Energi AS, and the Research Council of Norway.

process. The authors have searched for literature covering modeling of heating element dynamics in general. Unfortunately, no relevant results were found on this topic.

Komperød and Lie (2010) considers empirical modeling of the same input/output pair of the CZ process as the present paper. Komperød and Lie (2010) concludes that the dynamics of the heating element power are somewhat nonlinear. In that paper the nonlinearity is handled by using nonlinear, non-adaptive system identification. The present paper will handle the nonlinearity by using linear, adaptive system identification.

2. NOTATION AND DEFINITIONS

Table 1 presents the notation used in this paper.

Table 1. Notation used in this paper.

e	One-step-ahead prediction error, $e_k = y_k - \hat{y}_k$.
g	Adaptive gain.
k, t	Indices referring to sample numbers in the dataset.
N	The total number of samples in a dataset or a section of a dataset.
P	Actual (measured) power to the heating element [kW].
P_{nom}	Nominal power to the heating element [kW] (see main text for explanation).
ΔP	Actual (measured) power to the heating element [kW] as deviation from the nominal power, i.e. $\Delta P = P - P_{\text{nom}}$.
q	The time-shift operator defined by $x_{k+1} = qx_k$ and $x_{k-1} = q^{-1}x_k$.
s	Triode for alternating current (TRIAC) input signal [%] (output from control system).
s_{nom}	Nominal TRIAC input signal [%] (see main text for explanation).
Δs	TRIAC input signal [%] as deviation from the nominal input signal, i.e. $\Delta s = s - s_{\text{nom}}$.
u	System input for a general system.
W	Criterion to be optimized during system identification.
y	System output for a general system.
θ	A vector containing the parameters to be identified using system identification.
λ	Forgetting factor of adaptive system identification algorithms.

A variable with subscript k refers to the variable's value at timestep k. For example P_k refers to the heating element power at timestep k. Subscript "nom" refers to the variable's nominal value. The operating point $(s_{\text{nom}}, P_{\text{nom}})$ is defined to be a steady state operating point. Any variable with a hat-notation ($\hat{}$) refers to the one-step-ahead prediction of this variable. For example $\hat{y}_k$ refers to the one-step-ahead prediction of y_k, i.e. the prediction of y_k based on measurements available at timestep $k-1$ and (if relevant) any previous measurements. Unless otherwise specified, prediction is understood as one-step-ahead prediction.

Polynomials to be used in linear polynomial models are defined as

$$A(q) \stackrel{\text{def}}{=} 1 + \sum_{i=1}^{n_a} a_i q^{-i}, \tag{1}$$

$$B(q) \stackrel{\text{def}}{=} \sum_{i=1}^{n_b} b_i q^{-i}, \tag{2}$$

$$C(q) \stackrel{\text{def}}{=} 1 + \sum_{i=1}^{n_c} c_i q^{-i}. \tag{3}$$

The parameters a_i, b_i, and c_i are to be identified using system identification. For computational reasons, as well as to simplify notation, the parameters are stacked in a parameter vector θ. The constants n_a, n_b, and n_c are to be specified to the system identification algorithm. There is a time delay of one sample in the dataset to be used in this paper. For processes without any time delay, the lower summation limit of $B(q)$ should be 0 instead of 1. The $A(q)$ and $C(q)$ polynomials are defined as above in any case.

The linear polynomial models that will be considered in this paper are based on the ARMAX model structure

$$A(q)y_k = B(q)u_k + C(q)e_k. \tag{4}$$

The term noise model refers to the dynamics from the one-step-ahead prediction error, e, to the system output, y. Solving (4) with respect to y_k shows that the noise model of ARMAX models is $C(q)/A(q)$. Model structures having noise models are suitable for prediction, because the noise models correct the model states when there are deviations between the measured process outputs and the simulated model outputs. The term input-output model will be used to refer to the dynamics from the system input, u, to the system output, y.

The optimization criterion W will be used to compare the one-step-ahead prediction performance of various models. The criterion is defined as

$$W \stackrel{\text{def}}{=} \frac{1}{N} \sum_{k=1}^{N} {e_k}^2. \tag{5}$$

Text written in `teletype font` refers to MATLAB commands, for example `armax`.

3. THE CZOCHRALSKI CRYSTALLIZATION PROCESS

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials. The plant considered in this paper converts multicrystalline silicon into monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Monocrystalline silicon wafers give solar cells of higher efficiency than multicrystalline silicon wafers.

The CZ process is a batch process. The main batch steps are illustrated in Figure 1. (i) Initially multicrystalline silicon is melted in a crucible. (ii) When the silicon is molten, the tip of a seed crystal is dipped into the melt.

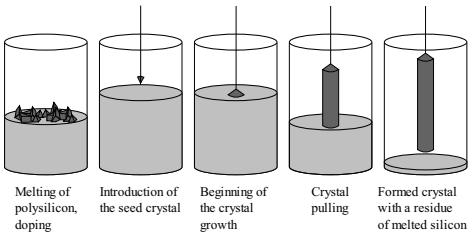


Fig. 1. The main batch steps of the CZ process. Illustration from Wikipedia (the illustration is released to public domain by the copyright holder).

The seed crystal is monocrystalline and has the crystal structure that is to be produced. (iii) When the tip of the seed crystal begins to melt, the crystal is slowly elevated. As the crystal is lifted, the molten silicon solidifies on the crystal. (iv) The crystal grows radially and axially. The produced crystal is referred to as an ingot. The crucible temperature and the vertical pulling speed are used to control the ingot diameter. Stable growing conditions are essential to produce high crystal quality. (v) As the final ingot length is reached, the crystal growth is terminated by slowly decreasing the crystal diameter until zero. During the entire batch process, the crucible is rotated in one direction, and the seed crystal is rotated in the opposite direction. An introduction to crystal growth, including the CZ process, is given in Lan (2004). The publication also provides an extensive number of references for further reading.

SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a real-life CZ process. At this plant the crucible is heated using a cylinder-shaped heating element, which encircles the crucible. The heating element power, P, is manipulated using a triode for alternating current (TRIAC). This paper considers adaptive system identification of the dynamics from the TRIAC input signal, s, to the heating element power, P, based on a dataset logged from this plant.

The dataset that is used for system identification is shown in Figure 2. The experiment was done without the crucible mounted in the process. The figure shows that increased s gives instantaneously increased P. However, P then slowly decreases. It is assumed that this decrease is caused by increased electric resistance due to increased temperature in the heating element. The dataset has 8311 samples. The sample time is 2 seconds. In this paper, sample number (not seconds) will be used to refer to time-progress.

Before system identification the dataset is preprocessed by subtracting a steady-state operating point. Figure 2 happens to reveal that ($s_{\text{nom}} = 45\%, P_{\text{nom}} = 45\text{kW}$) is a steady state operating point. Hence, the data used for system identification is $\Delta s_k = s_k - 45\%$, $\Delta P_k = P_k - 45\text{kW}\ \forall\ k$. The dataset is split into two sections: The first 2000 samples will be referred to as the initialization set, and the remaining samples (sample 2001 to 8311) will be referred to as the validation set.

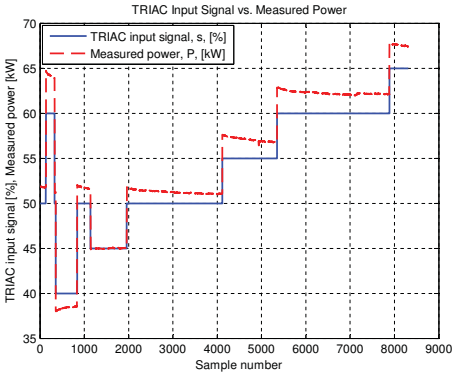


Fig. 2. The dataset to be used for system identification.

4. INTRODUCTION TO ADAPTIVE SYSTEM IDENTIFICATION

Adaptive system identification continuously tunes model parameters in order to adapt the model to slowly varying process parameters. This is particularly useful when models are used for online-applications, such as model-based control.

The basic principle of adaptive system identification is to adjust the parameter vector, θ, at each new sample, k, based on the one-step-ahead prediction error $e_k = y_k - \hat{y}_k$. The prediction $\hat{y}_k$ is computed using the model parameters estimated at timestep $k-1$. A possible reason for e_k is model imperfection, therefore θ is adjusted whenever $e_k \neq 0$. However, other likely reasons for e_k are process disturbances and measurement noise, which do not justify any changes to θ. Hence, how much θ is to be changed based on a given e_k should balance the covariance of the change of θ (encourage change of θ) and the covariances of process disturbances and measurement noise (discourage change of θ).

A commonly used approach for adaptive system identification is to apply a forgetting factor $\lambda < 1$. At timestep t, ${e_t}^2$ carries a weight of 1 in the optimization criterion W, while ${e_{t-1}}^2$ carries a weight of λ, ${e_{t-2}}^2$ carries a weight of λ^2, and so on. Hence, older samples will be "forgotten" in the sense that they have less impact on W.

A comprehensive introduction to adaptive system identification is given in Ljung (1999, Chapter 11). The results of the present paper are developed using the tools for adaptive system identification in the MATLAB System Identification Toolbox. An in-depth documentation of this toolbox is given in Ljung (2009).

5. IDENTIFYING A NON-ADAPTIVE ARMAX MODEL

In the present paper, adaptive, linear system identification is used to handle the nonlinearities of the dynamics from Δs to ΔP. For the purpose of comparing adaptive and non-adaptive models, a non-adaptive ARMAX model is

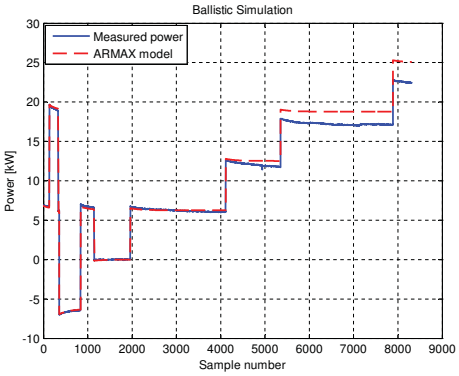


Fig. 3. Ballistic simulation of the non-adaptive ARMAX model.

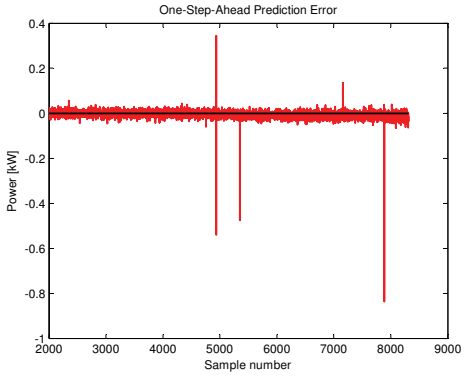


Fig. 4. One-step-ahead prediction error, e, of the non-adaptive ARMAX model over the validation set. Zero is marked by a black line.

identified based on the initialization set. As shown in Komperød and Lie (2010), $n_a = 1$ and $n_b = 2$ are suitable polynomial orders to model the dynamics from Δs to ΔP. Therefore, the ARMAX model will be on the form

$$\Delta P_k + a_1 \Delta P_{k-1} = b_1 \Delta s_{k-1} + b_2 \Delta s_{k-2} + e_k + c_1 e_{k-1}, \tag{6}$$

where n_c has been chosen equal to n_a, i.e. $n_a = n_c = 1$, and $e_k \stackrel{\text{def}}{=} \Delta P_k - \Delta \hat{P}_k$. Hence, the parameter vector to be identified is $\theta = [a_1\ b_1\ b_2\ c_1]^T$. The parameters were identified using the MATLAB command `armax`. This command identifies the parameter vector, θ, that gives the smallest W over the dataset used for identification, in this case the initialization set.

Figure 3 shows ballistic simulation of the ARMAX model over the entire dataset. The model has good fit to measured data over the initialization set, i.e. over the section of the dataset that was used to identify the model. After the step taking place at sample number 4117, the model fit becomes rather poor.

Figure 4 shows the one-step-ahead prediction error, e, over the validation set. The figure reveals five samples which have very large prediction errors. The prediction errors at samples 4936, 4937, and 7158 are caused by outliers in the dataset. These outliers are discussed in Komperød and Lie (2010). The prediction errors at samples 5350 and 7887 coincide with steps in Δs. Hence, these prediction errors are probably caused by model imperfection, rather than outliers. Visual inspection of Figure 4 indicates that the prediction errors are close to zero-mean in the beginning of the validation set, but tend to be negative in the last part of the validation set. The mean prediction error of the first 300 samples of the validation set is 5.0×10^{-4}kW, while the corresponding value for the last 300 samples is -1.49×10^{-2}kW, i.e. almost 30 times larger (absolute value) for the last samples. Hence, the ARMAX model systematically gives too large predictions, $\Delta \hat{P}_k$, at the last part of the validation set.

For the entire validation set the prediction error criterion for the non-adaptive ARMAX model is $W = 4.16 \times 10^{-4}\text{kW}^2$. Please note that W is a criterion that quantifies the magnitude of the prediction errors, while the mean-values discussed in the previous paragraph indicate whether the prediction errors are zero-mean or not. Hence, there is no close relationship between W and the mean-values.

6. IDENTIFYING AN ADAPTIVE ARMAX MODEL

In Section 5 it was concluded that the non-adaptive ARMAX model does not perform very well on the validation set. The model seems to give systematically too high one-step-ahead predictions in the last part of the validation set, which results in non-zero-mean prediction errors. This section considers how that problem can be handled by using adaptive system identification. The results of the present section are developed using the MATLAB command `rarmax` (Recursive ARMAX).

To compare the adaptive ARMAX model with the non-adaptive ARMAX model, the criterion W of (5) is computed over the validation set for both models. In other words: What is to be examined is whether adaptive system identification gives better one-step-ahead predictions, $\Delta \hat{P}$, over the validation set.

The performance of adaptive models depends on whether the adaptive algorithm is provided good initial values for θ, as well as some other variables that are to be initialized (for details about these variables, please refer to the documentation of the MATLAB command `rarmax`). The command `rarmax` was run twice: It was first run using the initialization set, and then run using the validation set. The parameter vector θ and other relevant variables computed during the first run were provided as initial values for the second run. Only the second run was considered when computing W.

The choice of the forgetting factor, λ, heavily influences the performance of adaptive models. Figure 5 shows the

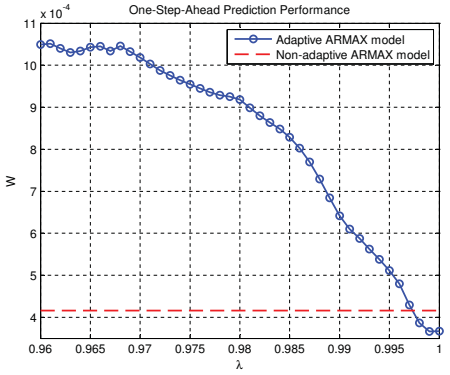


Fig. 5. The criterion W calculated for the validation set, plotted as function of the forgetting factor λ. The performance of the non-adaptive ARMAX model is also indicated.

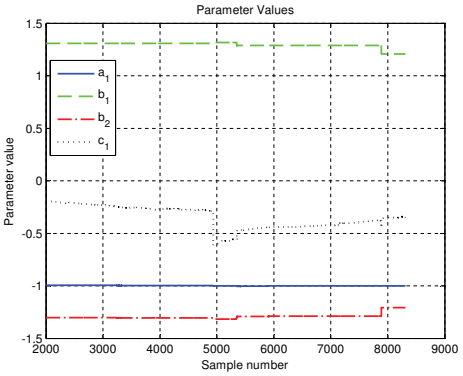


Fig. 6. The parameters of the adaptive ARMAX model as they change over the validation set. The forgetting factor is $\lambda = 0.9994$.

optimization criterion W computed over the validation set, plotted as function of λ. The resolution of λ is 10^{-3}. The performance of the non-adaptive ARMAX model identified in Section 5 is also indicated in the figure. The figure shows that the adaptive model gives better performance than the non-adaptive model for the highest values of λ. However, the adaptive model gives poor performance if λ is chosen too low. In other words: For the purpose of one-step-ahead prediction, it is desirable to tune the parameter vector θ online, but it should be updated with very small steps (having high λ). A reasonable explanation of this conclusion is that if λ is too small, the previous input and output data will be forgotten so soon that there are not enough data available to compute a reliable model.

The best performance in Figure 5 is obtained at $\lambda = 0.999$, although the difference between $\lambda = 0.999$ and $\lambda = 1.000$ is negligible. The optimization criterion W was plotted again for the range $\lambda \in [0.9980, 1.0000]$ with a resolution of 10^{-4}. This plot is not shown. The best performance is at $\lambda = 0.9994$. W is strictly increasing in both directions from this point.

For $\lambda = 0.9994$, the optimization criterion is $W = 3.57 \times 10^{-4}\text{kW}^2$, which is 14% lower than the non-adaptive ARMAX model. Hence, the adaptive model gives significantly better predictions. However, it should be emphasized that the λ value is chosen based on the model's performance at the validation set, i.e. it is not decided based on an independent dataset. For an online application, λ must be decided in advance, which may give somewhat poorer performance.

Figure 4 showed that the non-adaptive ARMAX model gives a systematic error at the last part of the validation set. A similar plot for the adaptive ARMAX model shows no systematic error. This plot is not shown. The mean values of the prediction errors, e, over the first 300 samples and over the last 300 samples of the validation set are 4.1×10^{-4}kW and 3.7×10^{-4}kW, respectively.

Figure 6 shows how the parameters of the adaptive ARMAX model change throughout the validation set. There are most variations in c_1, which indicates that the adaptive algorithm mainly changes the noise model, rather than the input-output model, to improve the prediction performance. There are only small variations in b_1 and b_2, except for those changes that coincide with the last step of Δs at sample 7886. The changes of a_1 are so small that they may seem negligible. Unfortunately, they are not. The parameter crosses the line -1. Hence, the pole of the model moves across the unit circle. This means that the model changes between being stable and being unstable. Hence, even though the model may be good for one-step-ahead prediction, it is useless for ballistic simulation and multi-step-ahead prediction.

7. IDENTIFICATION WITH ADAPTIVE GAIN

In Section 6, an adaptive algorithm was used to adjust all parameters of the ARMAX model (6). Although the adaptive model does improve the one-step-ahead prediction, it is not advisable to use this approach because the model changes between being stable and unstable. This section presents an approach for adaptive identification which eliminates that problem. The approach also gives a much better prediction performance than the approach of Section 6.

Komperød and Lie (2010) considers nonlinear, non-adaptive system identification of the dynamics from Δs to ΔP. The resulting model of that paper is a Hammerstein model. The Hammerstein model gives significant improvement of both ballistic simulation and one-step-ahead prediction compared to linear models. A Hammerstein model is a linear transfer function of which input is processed through a nonlinear, static function. The static function used in Komperød and Lie (2010) is a second order polynomial. In the range of interest, the first derivate of this polynomial does not change sign (plus/minus). Hence, processing the model input through this polynomial is equivalent to let the model's gain be a function of the input. Inspired by this result, the present section considers

how to make the gain of the model (6) adaptive, while keeping the pole, the zero, and the noise model fixed.

Equation (6) can be written on polynomial form as

$$A(q)\Delta P_k = B(q)\Delta s_k + C(q)e_k. \tag{7}$$

Changing the gain of the input-output model, without changing the pole, the zero, nor the noise model, can be achieved by multiplying $B(q)$ by a gain g

$$A(q)\Delta P_k = gB(q)\Delta s_k + C(q)e_k. \tag{8}$$

Equation (8) will now be rewritten to a form that allows g to be identified using adaptive system identification. For numerical reasons it is desirable to use the recursive least squares method for tuning g. Equation (8) can be written as

$$\frac{A(q)}{C(q)}\Delta P_k = g\frac{B(q)}{C(q)}\Delta s_k + e_k. \tag{9}$$

Using $e_k = \Delta P_k - \Delta\hat{P}_k$ gives

$$\frac{A(q)}{C(q)}\Delta P_k = g\frac{B(q)}{C(q)}\Delta s_k + \Delta P_k - \Delta\hat{P}_k. \tag{10}$$

Solving for $\Delta\hat{P}_k$ gives

$$\begin{aligned}\Delta\hat{P}_k &= \Delta P_k - \frac{A(q)}{C(q)}\Delta P_k + g\frac{B(q)}{C(q)}\Delta s_k \\ &= \frac{C(q)-A(q)}{C(q)}\Delta P_k + g\frac{B(q)}{C(q)}\Delta s_k.\end{aligned} \tag{11}$$

Please note that $C(q) - A(q)$ contains no constant term, because both $A(q)$ and $C(q)$ have the constant term 1 which cancels in the subtraction. The right side of (11) therefore depends on ΔP only up to $k-1$. Hence, the form (11) is suitable as a predictor.

The gain g is to be tuned so that W of (5) is minimized for the predictor (11). Adding e_k to both sides of (11) gives

$$\Delta P_k = \frac{C(q)-A(q)}{C(q)}\Delta P_k + g\frac{B(q)}{C(q)}\Delta s_k + e_k, \tag{12}$$

where the prediction error, e_k, is to be used for computation of W.

In order to fit the form of the ordinary least squares method, equation (12) is rewritten to

$$\underbrace{\Delta P_k - \underbrace{\frac{C(q)-A(q)}{C(q)}\Delta P_k}_{\Delta P_k^G}}_{\Delta P_k^F} = \underbrace{g\frac{B(q)}{C(q)}\Delta s_k}_{\Delta s_k^F} + e_k. \tag{13}$$

The notations indicated by braces in (13) will be used in the succeeding text. In (13), g is the parameter to be identified, Δs_k^F is the regressor, ΔP_k^F is the output to be estimated (predicted) from the regressor, and e_k is the error to be minimized. Hence, the adaptive algorithm will tune g so that $g\Delta s_k^F$ gives the best possible prediction $\Delta\hat{P}_k^F$. When $\Delta\hat{P}_k^F$ is computed at timestep $k-1$, the desired prediction $\Delta\hat{P}_k$ can easily be calculated as ΔP_k^G is known at timestep $k-1$.

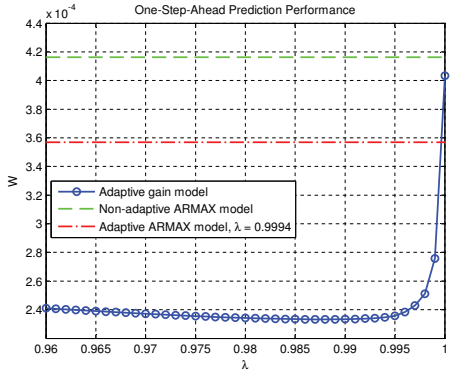


Fig. 7. The optimization criterion W of the adaptive gain model calculated over the validation set. The performance of the non-adaptive ARMAX model and the adaptive ARMAX model with $\lambda = 0.9994$ are also shown.

When implementing this adaptive system identification approach, the values of the parameters a_1, b_1, b_2, and c_1 are based on the non-adaptive ARMAX model which was identified from the initialization set in Section 5.

For the purpose of MATLAB implementation, the command `rarx` (Recursive ARX) may be used. Although `rarx` simplifies the implementation in MATLAB, it is not difficult to implement the recursive least squares algorithm from scratch, see for example Ljung (1999, Section 11.2).

To avoid confusion between the adaptive ARMAX model developed in Section 6 and the adaptive model developed in this section, the latter model will be referred to as the adaptive gain model.

Figure 7 shows the performance criterion W for the adaptive gain model as function of the forgetting factor λ. The adaptive gain model outperforms the adaptive and non-adaptive ARMAX models, except for at the highest λ value. The figure also shows that the adaptive gain model is very robust to the choice of λ: W is in the range 2.33×10^{-4} to 2.43×10^{-4}kW^2 for all λ values between 0.960 and 0.997. The best performance is at $\lambda = 0.988$, which gives $W = 2.33 \times 10^{-4}$kW^2.

For $\lambda = 0.988$ the mean prediction error over the first 300 samples of the validation set is -1.9×10^{-4}kW. For the last 300 samples the corresponding value is 1.0×10^{-3}kW. The first of these values are better than for both the adaptive and non-adaptive ARMAX models. The last value is somewhat poorer than the adaptive ARMAX model, but much better than the non-adaptive

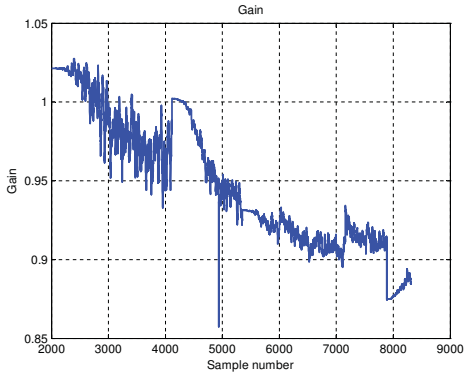


Fig. 8. The gain, g, as it is tuned by the adaptive gain algorithm over the validation set. The forgetting factor is $\lambda = 0.988$.

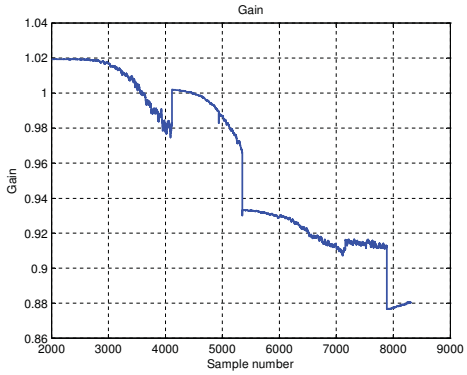


Fig. 9. The gain, g, as it is tuned by the adaptive gain algorithm over the validation set. The forgetting factor is $\lambda = 0.997$.

ARMAX model. Visual inspection of the prediction error plot reveals no systematic error.

Figure 8 shows the gain, g, over the validation set for $\lambda = 0.988$. The gain is subject to high-frequent variations. This is to be expected when the forgetting factor is low, because a lower forgetting factor gives larger parameter updates for a given prediction error, e. Even though the high-frequent variations of g do not influence the one-step-ahead prediction performance, these variations may be unfortunate for other applications, such as model-based control. As shown in Figure 7, the adaptive gain algorithm is very robust to the choice of λ. Hence, for some applications it may be desirable to choose a higher λ in order to reduce high-frequent variations in g. Figure 9 shows g for $\lambda = 0.997$. This λ value gives $W = 2.43 \times 10^{-4}\text{kW}^2$, which is only 4.3% higher than for $\lambda = 0.988$. The high-frequent variations of g are significantly attenuated when increasing λ from 0.988 to 0.997.

8. MODEL WEAKNESSES

The parameters a_1, b_1, b_2, and c_1, which were used as non-adaptive parameters in the adaptive gain model, were identified from a short section of the dataset. More accurate parameters can be found by identifying a non-adaptive ARMAX model over one or several longer datasets.

If the adaptive gain model is to be used for online applications, it may be necessary to implement an algorithm that increases λ during long periods of small excitations in s. This is to prevent information that was recorded during large excitations from being lost. If information from only small excitations is available, it may be insufficient for maintaining a reliable model.

9. FURTHER WORK

Adaptive system identification run over several datasets, and over several input/output pairs of the CZ process, may be used to examine whether the adaptive parameters change in certain patterns during the batch progress. If so, this may give information of how the process dynamics change throughout the batch. This information may be used for further modeling and analysis, as well as monitoring and control.

Komperød et al. (2010) considers how the crucible temperature control can be improved using cascade control. The paper covers model-based PID tuning of a power controller. The power controller has P as process input and uses s as controlling element. The adaptive gain method suggested in Section 7 of the present paper may be used to make the power controller adaptive by adjusting the controller gain, K_p, inverse proportional to the adaptive process gain, g. However, Komperød et al. (2010) shows that the power controller will have good reference (set-point) tracking and good disturbance rejection, even with conservative gain margin and phase margin. Hence, making the controller adaptive is most likely not necessary for achieving good control performance.

The modeling work presented in this paper is part of a larger project which aims to model and analyze the most important input/output pairs of the CZ process. These results will then be used to improve modeling and control of the CZ process.

10. CONCLUSIONS

This paper presents two approaches for adaptive system identification of the dynamics from the TRIAC input signal, s, to the actual (measured) heating element power, P. The dataset used for identification was logged from a real-life CZ process.

The first approach, referred to as the adaptive ARMAX model, uses the MATLAB command `rarmax` to tune all parameters of the ARMAX model. This approach improves the one-step-ahead prediction performance compared to the non-adaptive ARMAX model. Unfortunately, the adaptive ARMAX model changes between being stable and unstable, which makes it useless for ballistic simulation and multi-step-ahead prediction. The one-step-ahead prediction performance of the adaptive ARMAX model is very sensitive to the choice of the forgetting factor, λ.

The second approach, referred to as the adaptive gain model, is successful. This model is an ARMAX model, where only the gain of the input-output model is adaptive, while the pole, the zero, and the noise model are fixed. The adaptive gain can be computed using the recursive least squares method. The adaptive gain model has much better one-step-ahead prediction performance than the adaptive ARMAX model, and is very robust to the choice of λ. As the pole of the adaptive gain model is fixed, it is not an issue whether the model can change between being stable and unstable.

ACKNOWLEDGEMENTS

The authors are most grateful to SINTEF Materials and Chemistry in Trondheim, Norway, for giving access to data from the company's CZ process, and for allowing these data to be used in this paper and other publications. Eivind Johannes Øvrelid at SINTEF is acknowledged for sharing his knowledge and experience of the CZ process.

The first author is most grateful for the financial support of his PhD study from NorSun AS, Østfold Energi AS, and the Research Council of Norway. The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is also very much appreciated.

REFERENCES

Komperød, M., Bones, J.A., and Lie, B. (2010). Rejection of power disturbances in the Czochralski crystallization process using cascade control. In *51st International Conference of Scandinavian Simulation Society (SIMS 2010)*. Oulu, Finland.

Komperød, M. and Lie, B. (2010). Empirical modeling of heating element power for the Czochralski crystallization process. *Modeling, Identification and Control*, 31(1), 19–34. doi:10.4173/mic.2010.1.2.

Lan, C.W. (2004). Recent progress of crystal growth modeling and growth control. *Chemical Engineering Science*, 59, 1437–1457.

Lee, K., Lee, D., Park, J., and Lee, M. (2005). MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 3, 252–257.

Ljung, L. (1999). *System Identification - Theory for the User, 2nd Edition*. Prentice-Hall, Upper Saddle River, New Jersey.

Ljung, L. (2009). *System Identification Toolbox 7 - User's Guide*. The MathWorks, Inc.

Paper F

Rejection of Power Disturbances in the Czochralski Crystallization Process Using Cascade Control

The candidate presented this paper at the 51st Conference on Simulation and Modelling (SIMS 2010). The paper is also included in the conference's proceedings. The conference was held October 14th-15th 2010 in Oulu, Finland. The research presented in this paper is based on logged process data from experiments at the Czochralski (CZ) crystallization process at SINTEF Materials and Chemistry in Trondheim, Norway. The candidate participated in these experiments, except for the experiment used to identify the model given by Equation (5). The candidate has done all the research presented in this paper and written the paper during his PhD study.

SIMS 2010 Proceedings
The 51st Conference on Simulation and Modelling
14-15 October 2010 Oulu, Finland

Editor: Esko Juuso, University of Oulu

ISBN 978-952-5183-42-9

http://www.automaatioseura.fi

Rejection of Power Disturbances in the Czochralski Crystallization Process Using Cascade Control ⋆

Magnus Komperød * **John Atle Bones** ** **Bernt Lie** ***

* *Østfold University College, Faculty of Engineering, 1757 Halden, Norway (e-mail: magnus.komperod@hiof.no).*
** *SINTEF Materials and Chemistry, Department of Metallurgy, 7465 Trondheim, Norway (e-mail: johnatle.bones@sintef.no).*
*** *Telemark University College, Faculty of Technology, 3901 Porsgrunn, Norway (e-mail: bernt.lie@hit.no).*

Abstract: The Czochralski (CZ) crystallization process is used to produce monocrystalline silicon, which is used in solar cells and electronics. The temperature of the crucible containing the molten silicon must be tightly controlled to achieve high silicon quality. SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a CZ process. At this plant the crucible is heated by a heating element encircling the crucible. The heating element power is manipulated using a triode for alternating current (TRIAC). Presently the crucible temperature is controlled using a single-loop PID controller, which output manipulates the TRIAC.
This paper suggests using cascade control for controlling the crucible temperature. The cascade inner loop is a power control loop, and the cascade outer loop is the temperature control loop. Hence, the output of the temperature controller will be the reference (setpoint) to the power controller.
The main motivations for using cascade control are (i) fast and effective rejection of power disturbances, and (ii) robustness to parameter variations in the inner loop.
This paper also presents model-based PID tuning of the power controller. Different PID tunings are compared by the bandwidth of the control loop and Bode diagrams of the closed loop transfer function and the sensitivity function. An integral controller (I controller) gives best control performance.
Robustness to parameter variations in the inner process is simulated using Bode diagrams of the closed inner loop for increased and decreased process gain.

Keywords: Cascade control, Czochralski crystallization process, Disturbance rejection, Model-based PID tuning, Temperature control.

1. INTRODUCTION

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials, i.e. materials of homogeneous crystal structure. Among the most important applications is production of monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. The CZ process is a batch process. Initially multicrystalline silicon is melted in a crucible. Tight crucible temperature control is most important for producing silicon of high quality.

SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a CZ process. At this plant the crucible temperature is controlled using a single-loop PID controller, which output is connected to a triode for alternating current (TRIAC). The TRIAC manipulates the power to a cylinder-shaped heating element, which encircles the crucible. The actual power to the heating element is measured, but presently not included in the control strategy.

Experiments done at the SINTEF CZ plant reveal that there are significant process disturbances to the heating element power. In other words: There are responses in the measured power that can not be explained by the TRIAC. These power disturbances are known to be process disturbances, not measurement noise, because they also influence the crucible temperature. Due to their influence on the crucible temperature, and hence the silicon quality, these disturbances are most unfortunate.

The contributions of the present paper are: (i) A solution to effectively reject the power disturbances. The solution is based on cascade control. (ii) A real-life example of how the cascade control rejects a real power disturbance is presented. (iii) The inner process, i.e. the process to be controlled by the cascade slave controller, has an unusual frequency response. A frequency-based PID controller tun-

⋆ This paper is based on data that SINTEF Materials and Chemistry in Trondheim, Norway, most kindly has provided the authors. Some of the data were logged during experiments financially supported by NorSun AS. The first author's PhD study receives financial support from NorSun AS, Østfold Energi AS, and the Research Council of Norway.

ing is presented. (iv) It is shown how the suggested cascade control improves robustness to parameter variations in the inner process.

An introduction to crystal growth, including the CZ process, is given in Lan (2004). The publication also provides an extensive number of references for further reading. The principle of cascade control is presented in many process control books, for example Seborg et al. (2004). This book also presents frequency-based controller tuning. Lee et al. (2005) presents a feasible control strategy for the CZ process, which has proven to be successful on a real-life process.

2. NOTATION AND DEFINITIONS

Table 1 presents the notation used in this paper.

Table 1. Notation used in this paper.

EC	Power controller (the letter E refers to an electric variable).
ET	Power transmitter (power sensor).
GM	Gain margin [dB].
$h_0(z)$	Open loop transfer function.
$h_c(z)$	Time-discrete transfer function of the power controller, EC.
$h_p(z)$	Time-discrete transfer function of the dynamics from Δs to ΔP.
k	Index referring to sample number in the dataset.
K_i	Gain of integral controller (I controller) [s^{-1}].
K_p	Gain of PID controller.
$M(z)$	Closed loop transfer function.
$N(z)$	Sensitivity function.
P	Actual (measured) power to the heating element [kW].
P_{nom}	Nominal power to the heating element [kW].
P_{ref}	Reference (setpoint) to the power controller, EC, [kW].
ΔP	Actual (measured) power to the heating element [kW] as deviation from the nominal power, i.e. $\Delta P = P - P_{\text{nom}}$.
PM	Phase margin [°].
s	TRIAC input signal [%] (output from control system).
s_{nom}	Nominal TRIAC input signal [%].
Δs	TRIAC input signal [%] as deviation from the nominal input signal, i.e. $\Delta s = s - s_{\text{nom}}$.
T	Crucible temperature [°C].
T_{ref}	Reference (setpoint) to temperature controller [°C].
Δt	Sampling time [s].
t_i	Integral time of PID controller [s].
TC	Temperature controller.
TT	Temperature transmitter (temperature sensor).
U	Voltage at power grid [V].
z	The time-shift operator defined by $x_{k+1} = zx_k$ and $x_{k-1} = z^{-1}x_k$.
ω	Frequency [rad/s].
ω_c	Bandwidth of control loop [rad/s].

A variable with subscript k refers to the variable's value at timestep k. For example P_k refers to the heating element power at timestep k. Subscript "nom" refers to the variable's nominal value. The operating point $(s_{\text{nom}}, P_{\text{nom}})$ is defined to be a steady state operating point. Even though the CZ process is a batch process, it seems reasonable to approximate the dynamics of the heating element as a time-invariant sub-process. Also, as shown in Sections 6 and 7, the cascade control suggested in this paper makes the sub-process robust to low-frequent disturbances and parameter variations.

For a process transfer function, $h_p(z)$, and a controller transfer function, $h_c(z)$, the open loop transfer function, $h_0(z)$, is defined as

$$h_0(z) \stackrel{\text{def}}{=} h_p(z)h_c(z), \tag{1}$$

the closed loop transfer function, $M(z)$, is defined as

$$M(z) \stackrel{\text{def}}{=} \frac{h_0(z)}{1 + h_0(z)}, \tag{2}$$

and the sensitivity function, $N(z)$, is defined as

$$N(z) \stackrel{\text{def}}{=} \frac{1}{1 + h_0(z)}. \tag{3}$$

The bandwidth of a control loop, ω_c, is defined implicitly as

$$|h_0(e^{j\omega_c \Delta t})| \stackrel{\text{def}}{=} 0\text{dB} = 1. \tag{4}$$

It is understood that $|h_0(e^{j\omega\Delta t})| \geq 1 \,\forall\, \omega \in [0, \omega_c\rangle$.

3. THE CZOCHRALSKI CRYSTALLIZATION PROCESS

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials. The plant considered in this paper converts multicrystalline silicon into monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Monocrystalline silicon wafers give solar cells of higher efficiency than multicrystalline silicon wafers.

The CZ process is a batch process of which the main steps are illustrated in Figure 1. (i) Initially multicrystalline silicon is melted in a crucible. (ii) When the silicon is molten, the tip of a seed crystal is dipped into the melt. The seed crystal is monocrystalline, having the crystal structure that is to be produced. (iii) When the tip of the seed crystal begins to melt, the crystal is slowly elevated. As the crystal is lifted, the molten silicon solidifies on the crystal. (iv) The crystal grows radially and axially. The produced crystal is referred to as an ingot. The crucible temperature and the vertical pulling speed are used to control the ingot diameter. Stable growing conditions, including the crucible temperature, are essential for producing high crystal quality. (v) As the final ingot length is reached, the crystal growth is terminated by slowly decreasing the crystal diameter until zero. During the entire batch process, the crucible is

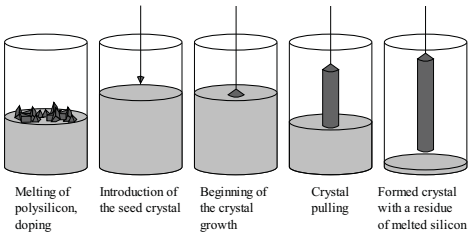


Fig. 1. The main batch steps of the CZ process. Illustration from Wikipedia (the illustration is released to public domain by the copyright holder).

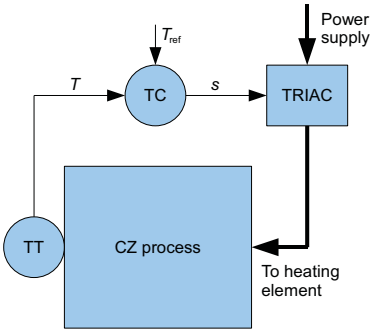


Fig. 2. The present temperature control strategy.

rotated in one direction, and the seed crystal is rotated in the opposite direction.

Lan (2004) gives an introduction to crystal growth, including the CZ process, and provides an extensive number of references for further reading. Many papers have been written about control of the CZ process. A feasible control strategy which has proven to be successful on a real-life CZ process is presented in Lee et al. (2005).

4. THE PRESENT TEMPERATURE CONTROL

SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a CZ process. At this plant the crucible is heated by a cylinder-shaped heating element, which encircles the crucible. The heating element power, P, is manipulated using a TRIAC. The present temperature control is a single-loop PID controller. This control strategy is illustrated in Figure 2. If the temperature, T, becomes too low, i.e. below the temperature reference, T_{ref}, the temperature controller, TC, increases its output signal which is connected to the TRIAC input signal, s. Then the TRIAC will let a larger part of the power grid voltage through, which increases P and finally T. Presently P is measured, but not included in the control strategy.

The temperature control of Figure 2 is a decent control strategy. For many applications with small process disturbances or more relaxed demands for temperature control performance, this simple control strategy is sufficient.

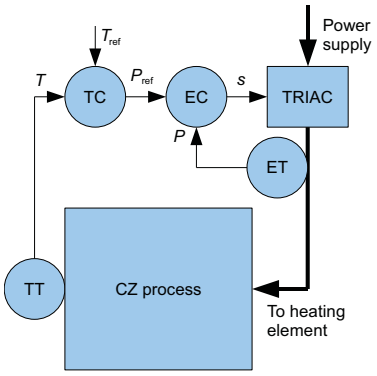


Fig. 3. The cascade control strategy suggested by the authors.

Experiments at the SINTEF CZ plant have revealed that there are significant process disturbances influencing P. In other words: There are responses in P that can not be explained by excitations of s. These responses are known to be process disturbances, not measurement noise, because there are coinciding responses in T. What causes these disturbances is not known for sure. A reasonable explanation is voltage variations at the power grid.

Using the control strategy of Figure 2, a power disturbance can not be rejected until it has caused some change to T. First then the control strategy can observe a deviation from the nominal process values. Also, due to measurement noise at TT, TC should not be too aggressively tuned. Although the power disturbance will be rejected, a most unfortunate temperature change must occur before the disturbance is detected at all.

5. PROPOSED SOLUTION: CASCADE CONTROL

The authors suggest to expand the single-loop control strategy of Figure 2 to a cascade control strategy as illustrated in Figure 3. Using this cascade control, the output of TC is used as reference (setpoint), P_{ref}, to a power controller, EC. EC will ensure that P, which is measured by the power transmitter, ET, follows the reference, P_{ref}. In other words: EC ensures that the power requested by TC is actually applied to the heating element. Both TC and EC are regular PID controllers.

The main motivation for using the suggested cascade control strategy is fast and effective rejection of power disturbances. EC ensures that such disturbances will be compensated in a very few seconds, before they have any significant impact to T. Also, this control strategy makes the temperature control loop more robust to nonlinearities and possible parameter variations in the dynamics between s and P. Robustness to parameter variations is considered in Section 7.

Figure 4 shows a real-life example of how EC rejects a power disturbance. This is a real disturbance, not artificial nor simulated. The disturbance occurs at time 2358s. Immediately, EC detects the disturbance and rejects it by

Fig. 4. A real-life power disturbance is effectively rejected by the power controller, EC.

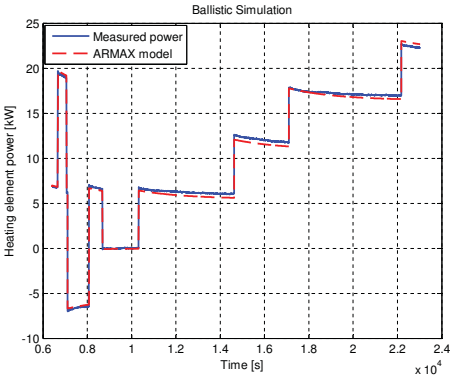


Fig. 5. Ballistic simulation of the model used for tuning the power controller, EC.

increasing s until P is restored to its reference value, P_{ref}. The data shown in Figure 4 is logged during melting of the silicon, i.e. before the actual crystal pulling begins. During the melting, TC is run in manual mode (open loop). Therefore P_{ref} is constant in Figure 4.

The power rejection shown in Figure 4 was logged during initial testing of the cascade control strategy. At this time EC had not been properly tuned. Section 6 considers model-based PID tuning of this controller. It is believed that using the computed PID parameters will further improve rejection of power disturbances.

6. MODEL-BASED PID TUNING OF POWER CONTROLLER

The disturbance rejection shown in Figure 4 is based on a non-scientific, trial-and-error PID tuning of the power controller, EC. Although this tuning provides a decent disturbance rejection, it is believed that proper tuning will improve the performance. This section considers model-based PID tuning of EC.

The output of the temperature controller, TC, is the reference to EC, P_{ref}. Hence, the tuning of EC should ensure that P effectively tracks P_{ref} over a wide range of frequencies. In other words: The control loop should have large bandwidth, ω_c.

The PID tuning to be presented in this section uses a frequency domain approach. This subject is covered in many control engineering books, for example Seborg et al. (2004, Chapter 14). The tuning is based on the linear ARMAX model developed in Komperød and Lie (2010). Ignoring the noise model, this ARMAX model is on the form

$$\Delta P_k + a_1 \Delta P_{k-1} = b_1 \Delta s_{k-1} + b_2 \Delta s_{k-2}. \tag{5}$$

The terms Δs and ΔP refer to deviation from the steady state operating point ($s_{\text{nom}} = 45\%, P_{\text{nom}} = 45\text{kW}$), i.e. $\Delta s = s - 45\%$ and $\Delta P = P - 45\text{kW}$. The parameter values are $a_1 = -0.9991$, $b_1 = 1.292$, and $b_2 = -1.291$.

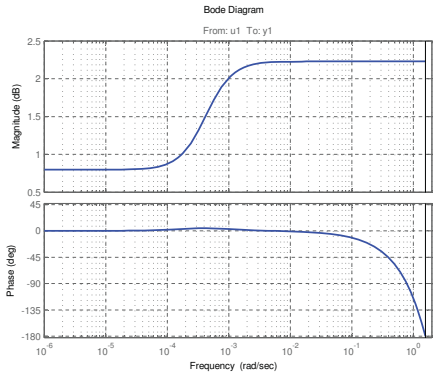


Fig. 6. Bode diagram of $h_p(z)$.

The sample time is $\Delta t = 2$s. Figure 5 shows ballistic simulation of the model (5). The model was run against its own calibration dataset, as no independent dataset was available. The reason for *not* splitting the dataset in a calibration section and a validation section is explained in Komperød and Lie (2010). The model (5) can be written as a time-discrete transfer function

$$h_p(z) \stackrel{\text{def}}{=} \frac{\Delta P(z)}{\Delta s(z)} = \frac{b_1 z + b_2}{z^2 + a_1 z}. \tag{6}$$

Figure 6 shows the Bode diagram of the transfer function (6). The frequency response is somewhat unusual as the magnitude, $|h_p(e^{j\omega\Delta t})|$, increases by the frequency, ω. Also the phase, $\angle h_p(e^{j\omega\Delta t})$, is increasing somewhat in a part of the frequency range, having its maximum of $+5°$ at approximately $\omega = 4 \times 10^{-4}$rad/s.

According to SINTEF Materials and Chemistry, the voltage at the power grid, U, may vary as much as $\pm 10\%$ of the nominal voltage. It is reasonable to assume that

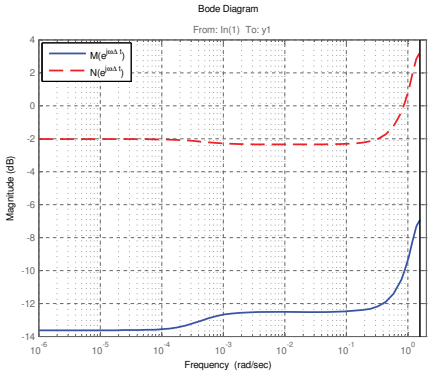


Fig. 7. Bode magnitude diagram of $M(z)$ and $N(z)$ using P controller with $K_p = 0.24$.

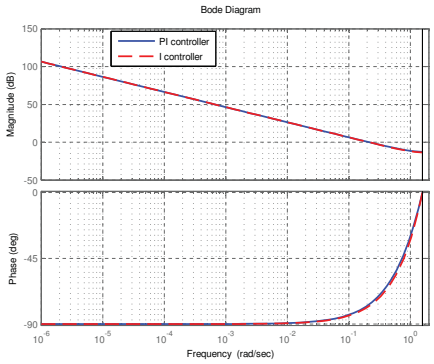


Fig. 8. Bode diagram for a PI controller and an I controller, both with $K_p = 0.021$ and $t_i = 0.1$s.

P is proportional to U^2. Hence, $|h_p(z)|$ is assumed to be proportional to U^2.

The model (5) is identified from a dataset logged at the SINTEF CZ process. The actual U during logging is unknown. It may have been 10% lower than the nominal value. If so, the maximum U, i.e. the nominal voltage plus 10%, may be approximately 20% higher than during logging. Hence, the factor U^2 may increase by 44%. Therefore, the gain margin, GM, required by voltage variation is a factor of 1.5. There may also be other causes to parameter variations, and the process is somewhat nonlinear. It seems reasonable to require additional GM of a factor of 2. Hence, the total GM should be a factor of $1.5 \times 2 = 3$. This is equal to 10dB. Due to the process' nonlinear character, the phase margin, PM, is conservatively chosen to 75°.

As the process transfer function, $h_p(z)$, is somewhat unusual, it is intuitive to try a proportional controller (P controller) first, because this is the simplest controller configuration. From Figure 6 it can easily be concluded that the controller gain, K_p, must be $-12.3\text{dB} = 0.24$ for GM to be 10dB. PM will then be infinite, because $|h_0(e^{j\omega\Delta t})| < 0\text{dB}\ \forall\ \omega$ using this K_p value.

Figure 7 shows $|M(e^{j\omega\Delta t})|$ and $|N(e^{j\omega\Delta t})|$ for the closed loop from P_{ref} to P using a P controller with $K_p = 0.24$. The P controller turns out to be a very poor choice: (i) $|M(e^{j\omega\Delta t})|$ is far below 0dB for all frequencies, which means that any reference signal to EC will be very poorly tracked. According to definition (4) this control loop does not have a bandwidth at all. (ii) $|M(e^{j\omega\Delta t})|$ increases for higher frequencies, which means that high-frequent measurement noise will be better tracked than a low-frequent reference signal. (iii) $|N(e^{j\omega\Delta t})|$ is relatively high for all frequencies. This means that power disturbances will be poorly rejected. Actually, in all frequency ranges this closed loop will track disturbances much better than the reference.

Proper investigation of Figure 6 would have revealed that a P controller is a poor choice. The reason is that $|h_p(e^{j\omega\Delta t})|$, and hence $|h_0(e^{j\omega\Delta t})|$, are small for low frequencies, which explains the issues (i)-(iii) of the previous paragraph. To achieve good reference tracking and good disturbance rejection, $|h_0(e^{j\omega\Delta t})|$ should be as large as possible within the constraints applied by the chosen stability margins. However, good reference tracking for high frequencies may cause high-frequent measurement noise to be tracked as well.

The magnitude $|h_0(e^{j\omega\Delta t})|$ can be increased for lower frequencies by including integral action, i.e. to use a PI controller instead of a P controller. The control loop bandwidth, ω_c, was compared for PI controllers with various integral time, t_i. For each t_i value, the gain, K_p, was set to the highest possible value that obeys both GM and PM. The result is shown in Table 2. The table shows that ω_c increases as t_i decreases, even though K_p is reduced to meet the stability demands.

Table 2. The controller gain, K_p, and the bandwidth, ω_c, for different integral time, t_i. K_p is given as ratio (not in dB).

t_i [s]	K_p	ω_c [rad/s]
60.0	0.240	5×10^{-3}
15.0	0.225	2×10^{-2}
3.0	0.180	8×10^{-2}
0.5	0.080	2×10^{-1}
0.1	0.021	3×10^{-1}

As K_p and t_i decrease, the proportional action will have less contribution to the controller dynamics. Hence, as $K_p \to 0$, while the ratio K_p/t_i is constant, the dynamics of the PI controller will converge to the dynamics of an integral controller (I controller). Figure 8 shows the Bode diagram for a PI controller and an I controller, both with $K_p = 0.021$ and $t_i = 0.1$s. The PI controller is implemented as the transfer function

$$h_c(z) = \frac{(K_p + \frac{K_p \Delta t}{t_i})\, z - K_p}{z - 1}, \tag{7}$$

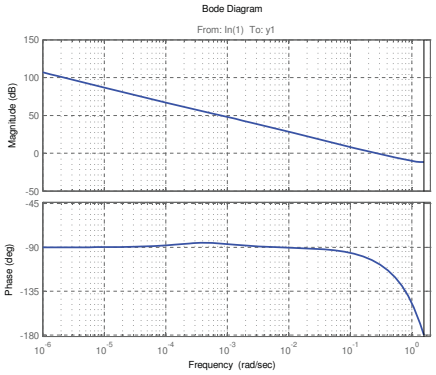


Fig. 9. Bode diagram of $h_0(z)$ for an I controller with $K_i = 0.2\text{s}^{-1}$.

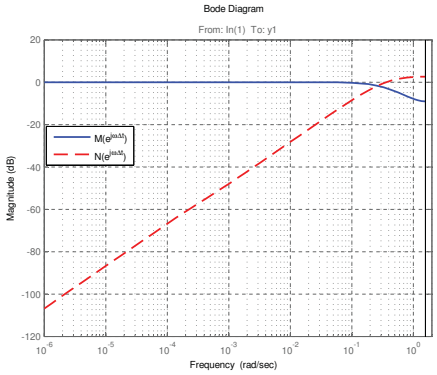


Fig. 10. Bode magnitude diagram of $M(z)$ and $N(z)$ for an I controller with $K_i = 0.2\text{s}^{-1}$.

and the I controller is implemented as the transfer function

$$h_c(z) = \frac{\frac{K_p \Delta t}{t_i}\, z}{z - 1}, \tag{8}$$

where Δt is the sampling time. Figure 8 shows that there is very little difference between the dynamics of the two controllers. Hence it is concluded that an I controller will be as good as an PI controller. For an I controller it is not necessary to tune both K_p and t_i. Instead, the tuning parameter K_i is defined as

$$K_i \stackrel{\text{def}}{=} \frac{K_p}{t_i}. \tag{9}$$

For an I controller, $K_i = 0.2\text{s}^{-1}$ is the highest value that obeys both GM and PM. For this K_i, GM is 12dB and PM is 75°. The bandwidth is $\omega_c = 0.3$rad/s.

Figure 9 shows the Bode diagram of $h_0(z)$ for an I controller with $K_i = 0.2\text{s}^{-1}$. Figure 10 shows the magnitudes of $M(z)$ and $N(z)$ for the same controller. The magnitude plot shows that the I controller has very good tracking performance up to ω_c. Also the disturbance rejection is very good almost up to ω_c. Considering the time domain, Figure 11 shows the response of the reference unit step. The response is very quick and has no overshoot. After 6s the response has reached almost 90% of its steady state value.

As an I controller gives high ω_c, the authors did not consider whether a PID controller with derivative action would further increase ω_c, because too high ω_c may be unfortunate due to measurement noise.

The authors intend to implement the tuning presented in this section at SINTEF Materials and Chemistry. It will be validated that the tuning gives satisfactory stability properties. It will also be considered whether the controller output, s, is subject to high-frequent variations ("noise"). If so, it may be desirable to reduce K_i and/or apply a lowpass measurement filter to the controller input, P.

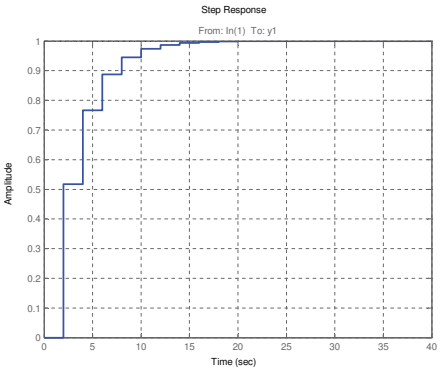


Fig. 11. Reference unit step for an I controller with $K_i = 0.2\text{s}^{-1}$.

7. ROBUSTNESS TO PARAMETER VARIATIONS

One of the most important advantages of cascade control, compared to single-loop control, is increased robustness to parameter variations in the inner process. In this case the inner process is the dynamics from s to P.

This section considers how the transfer function $M(z)$, i.e. the closed loop from P_{ref} to P, is influenced by voltage variations at the power grid. As discussed in Section 6, it is reasonable to assume that voltage variations affect $h_p(z)$ as changes of the transfer function's gain. A gain increment of a factor 1.5, i.e. 3.5dB, was considered to be the worst case scenario at the upside. Using the same argumentation for the downside gives a factor 0.67, i.e. -3.5dB, as the worst case.

Figure 12 shows Bode diagram of $M(z)$ for the nominal gain of $h_p(z)$, i.e. the gain of the transfer function (6), and ±3.5dB relative to the nominal gain. An I controller with $K_i = 0.2\text{s}^{-1}$ is used. Figure 12 shows that the differences in magnitude between the nominal gain and

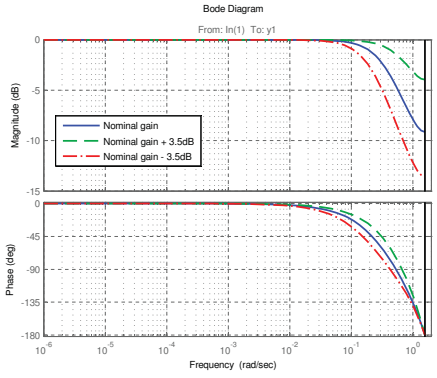


Fig. 12. Bode diagram of $M(z)$ for the nominal gain ± 3.5dB.

the ± 3.5dB gains are below 1dB up to $\omega = 0.1$rad/s. For higher frequencies the differences increase significantly. However, only for the highest frequencies the differences are above 3.5dB. Visual inspection of the phase diagram shows that the maximum phase differences between the nominal gain and the ± 3.5dB gains are approximately $15°$. For frequencies below $\omega = 0.01$rad/s there are very small phase differences.

8. WEAKNESSES OF THE SUGGESTED CONTROL STRATEGY

A general disadvantage of cascade control is that the control strategy depends on one additional measurement. That is the measurement of the inner process. For the process considered in this paper the additional measurement is the power transmitter, ET. A faulty or very noisy signal from the additional measurement may reduce the control performance compared to single-loop control. In other words: One more measurement is one more issue that may go wrong.

Except for this general concern of cascade control, there are to the authors' knowledge no significant disadvantages of the suggested control strategy.

9. FURTHER WORK

The suggested cascade control will be implemented on the CZ process at SINTEF Materials and Chemistry. The power controller tuning will be tested. It may be necessary to reduce K_i and/or apply a lowpass filter due to measurement noise. When the inner process is operating properly in closed loop, the focus will be shifted toward modeling and control of the outer process, i.e. the dynamics from the power reference, P_{ref}, to the crucible temperature, T.

Komperød and Lie (2010) and Komperød et al. (2010) consider nonlinear system identification and adaptive system identification, respectively, of the dynamics from s to P. The results of these papers can be used to make the power controller, EC, adaptive. However, as shown in the present paper, the inner loop will have high bandwidth even with conservative stability margins. Hence, it is likely that adaptive control is not necessary for the temperature cascade control to be successful.

The work presented in this paper is part of a larger project, which focus towards modeling, monitoring, and control of the CZ process.

10. CONCLUSIONS

Presently the crucible temperature at the SINTEF CZ process is controlled using a single-loop PID controller. This paper suggests using cascade control instead of single-loop control. The cascade inner loop will control the heating element power, and the cascade outer loop will control the crucible temperature. Hence, the temperature controller output will be the reference (setpoint) to the power controller.

The main motivations for using cascade control are (i) fast and effective rejection of power disturbances and (ii) increased robustness to parameter variations in the inner process.

This paper also considers model-based PID tuning of the power controller. A proportional controller (P controller) turns out to be a very poor choice, while an integral controller (I controller) works very well. The performance of various controller tunings were compared by considering the bandwidth of the control loop, as well as Bode diagrams of the closed loop transfer function and the sensitivity function.

Robustness to parameter variations was examined by considering Bode diagrams of the closed inner loop for various process gains. At lower frequencies the loop is very robust to changes of the process gain.

To the authors' knowledge the suggested cascade control has no significant disadvantages, except that it depends on one additional measurement.

ACKNOWLEDGEMENTS

The authors are most grateful to SINTEF Materials and Chemistry in Trondheim, Norway, for giving access to data from the company's CZ process, and for allowing these data to be used in this paper and other publications. The authors are also most grateful to NorSun AS for providing financial support to some of the experiments from which the data are logged. Eivind Johannes Øvrelid at SINTEF is acknowledged for sharing his knowledge and experience of the CZ process.

The first author is most grateful for the financial support of his PhD study from NorSun AS, Østfold Energi AS, and the Research Council of Norway. The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is also very much appreciated.

REFERENCES

Komperød, M., Bones, J.A., and Lie, B. (2010). Adaptive system identification of heating element power for the Czochralski crystallization process. In *51st International Conference of Scandinavian Simulation Society (SIMS 2010)*. Oulu, Finland.

Komperød, M. and Lie, B. (2010). Empirical modeling of heating element power for the Czochralski crystallization process. *Modeling, Identification and Control*, 31(1), 19–34. doi:10.4173/mic.2010.1.2.

Lan, C.W. (2004). Recent progress of crystal growth modeling and growth control. *Chemical Engineering Science*, 59, 1437–1457.

Lee, K., Lee, D., Park, J., and Lee, M. (2005). MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 3, 252–257.

Seborg, D.E., Edgar, T.F., and Mellichamp, D.A. (2004). *Process Dynamics and Control, 2nd Edition*. John Wiley and Sons, Inc., New Jersey, USA.

Paper G

A Sensor Fusion Algorithm for Filtering Pyrometer Measurement Noise in the Czochralski Crystallization Process

This article was published in the open-access journal Modeling, Identification and Control (MIC) in 2011. The research presented in this article is based on logged process data from an experiment at the Czochralski (CZ) crystallization process at SINTEF Materials and Chemistry in Trondheim, Norway. The candidate participated in this experiment. The candidate has done all the research presented in the article and written the article during his PhD study.

PAPER G 129
Modeling, Identification and Control, Vol. 32, No. 1, 2011, pp. 17–32, ISSN 1890–1328

A Sensor Fusion Algorithm for Filtering Pyrometer Measurement Noise in the Czochralski Crystallization Process

M. Komperød [1] J. A. Bones [2] B. Lie [3]

[1] *Faculty of Engineering, Østfold University College, N-1757 Halden, Norway. E-mail:* magnus.komperod@hiof.no

[2] *SINTEF Materials and Chemistry, Department of Metallurgy, N-7465 Trondheim, Norway. E-mail:* johnatle.bones@sintef.no

[3] *Faculty of Technology, Telemark University College, N-3901 Porsgrunn, Norway. E-mail:* bernt.lie@hit.no

Abstract

The Czochralski (CZ) crystallization process is used to produce monocrystalline silicon for solar cell wafers and electronics. Tight temperature control of the molten silicon is most important for achieving high crystal quality. SINTEF Materials and Chemistry operates a CZ process. During one CZ batch, two pyrometers were used for temperature measurement. The silicon pyrometer measures the temperature of the molten silicon. This pyrometer is assumed to be accurate, but has much high-frequency measurement noise. The graphite pyrometer measures the temperature of a graphite material. This pyrometer has little measurement noise. There is quite a good correlation between the two pyrometer measurements. This paper presents a sensor fusion algorithm that merges the two pyrometer signals for producing a temperature estimate with little measurement noise, while having significantly less phase lag than traditional lowpass-filtering of the silicon pyrometer. The algorithm consists of two sub-algorithms: (i) A dynamic model is used to estimate the silicon temperature based on the graphite pyrometer, and (ii) a lowpass filter and a highpass filter designed as complementary filters. The complementary filters are used to lowpass-filter the silicon pyrometer, highpass-filter the dynamic model output, and merge these filtered signals. Hence, the lowpass filter attenuates noise from the silicon pyrometer, while the graphite pyrometer and the dynamic model estimate those frequency components of the silicon temperature that are lost when lowpass-filtering the silicon pyrometer. The algorithm works well within a limited temperature range. To handle a larger temperature range, more research must be done to understand the process' nonlinear dynamics, and build this into the dynamic model.

Keywords: Complementary filters; Czochralski crystallization process; Measurement noise filtering; Pyrometer temperature measurement; Sensor fusion algorithm.

1 Introduction

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials, i.e. materials that have homogeneous crystal structures. Among the most important applications is production of monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics.

The CZ process is a batch process. During the process, multicrystalline silicon is melted in a crucible. The crucible is heated using a heating element, which

doi:10.4173/mic.2011.1.2

power is manipulated using a triode for alternating current (TRIAC). Tight control of the silicon temperature is most important for achieving high crystal quality. A reliable temperature measurement without too much measurement noise is decisive for achieving good temperature control.

This paper considers the temperature measurement of a real-life CZ process owned and operated by SINTEF Materials and Chemistry in Trondheim, Norway (hereafter referred to as SINTEF). During an experiment two pyrometers were used for temperature measurement. One pyrometer, referred to as the silicon pyrometer, measures the temperature in the molten silicon. The molten silicon temperature is the desired temperature signal for monitoring and control. Unfortunately, the signal from this pyrometer has considerable high-frequency measurement noise. Attenuating the noise using a traditional lowpass filter is an intuitive and feasible solution. However, this solution will cause significant phase lag over the filter, which is unfortunate for the temperature control.

The other pyrometer, referred to as the graphite pyrometer, has little noise. However, this pyrometer has the disadvantage of measuring the temperature of a graphite material encircling the silicon crucible. Hence, the signal of the pyrometer does not represent the actual silicon temperature. Fortunately, there is quite a high correlation between the silicon temperature and the graphite pyrometer signal.

The contribution of this paper is a sensor fusion algorithm that fuses the two pyrometer measurements. The purpose of the algorithm is to estimate the temperature of the molten silicon. For a given cut-off frequency, the algorithm estimate gives the same amount of measurement noise as a traditional lowpass filter, but has significantly less phase lag. The lower the cut-off frequency is chosen, the larger improvement of using the sensor fusion algorithm. On the other hand, if the cut-off frequency is chosen high, there is little phase lag over a traditional lowpass filter, and the algorithm does not give any significant improvement.

The sensor fusion algorithm presented in this paper is implemented as complementary filters. The filters are chosen as a lowpass Butterworth filter and a highpass Butterworth filter. A statistically optimal temperature estimate can in theory be computed using a Kalman filter or a Wiener filter. However, using Kalman filter or Wiener filter depends on noise descriptions that are not known. Using complementary Butterworth filters, the sensor fusion algorithm has one tuning parameter; the Butterworth filters' cut-off frequency.

The authors have searched for scientific papers issuing pyrometer measurement noise in the CZ processes. Unfortunately, no relevant results were found. There are several possible reasons for this negative search result: (i) Even though the silicon pyrometer at SINTEF has much measurement noise, this may not be an issue of other pyrometers at other CZ plants. (ii) The noise is attenuated using traditional lowpass filters despite the unfortunate phase lag. (iii) The noise problem is reduced by using controller tunings that attenuate high-frequency noise, for example avoiding derivative action and high gain in PID controllers. (iv) The noise problem is handled in the commercial CZ industry and is not published in scientific papers.

An introduction to complementary filters is given in Brown and Hwang (1997). Lyons (2011) gives a general introduction to digital signal processing. The sensor fusion algorithm also includes an empirical model developed using system identification. A comprehensive introduction to system identification is given in Ljung (1999).

Lan (2004) gives an introduction to crystal growth, including the CZ process, and provides an extensive number of references for further reading. There are many scientific papers covering modeling and control of the CZ process, for example Irizarry-Rivera and Seider (1997a,b) and Lee et al. (2005). However, the authors' literature search indicates that the important topic of sensor technology in the CZ process has received very limited attention.

2 Notation and Definitions

Table 1 presents the notation used in this paper. A variable with subscript k refers to the variable's value at timestep k. For example T_k refers to the silicon temperature at timestep k.

Please note the difference between $\hat{T}^g$ and u: u is the raw signal from the graphite pyrometer in Volt, while $\hat{T}^g$ is an estimate of the *silicon* temperature based on the signal u. These variables are related through the equation $\hat{T}^g = G(z)\, u$.

To simplify notation, the arguments s and z will be used to specify whether a transfer function H is time continuous or time discrete, respectively. That is, $H(s)$ and $H(z)$ describe the same model or filter, where $H(s)$ is the continuous version and $H(z)$ is the discrete version. This notation is erroneous in a strict mathematical sense, as $H(s)$ and $H(z)$ are different mathematical functions. However, it simplifies the notation without any risk of confusion.

3 The Czochralski Crystallization Process

The Czochralski (CZ) crystallization process is used to convert multicrystalline materials into monocrystalline materials. The plant considered in this paper converts multicrystalline silicon into monocrystalline silicon. Monocrystalline silicon is used in solar cell wafers and in computers and electronics. Solar cells based on monocrystalline silicon have higher efficiency than solar cells based on multicrystalline silicon.

The CZ process is a batch process, which main steps are illustrated in Figure 1. (i) Initially multicrystalline silicon is melted in a crucible. (ii) When the silicon is molten, the tip of a seed crystal is dipped into the melt. The seed crystal is monocrystalline and has the crystal structure that is to be produced. (iii) When the tip of the seed crystal begins to melt, the crystal is slowly elevated. As the crystal is lifted, the molten silicon solidifies on the crystal. During solidification, the crystal structure of the seed crystal is extended onto the solidifying silicon. (iv) The crystal grows radially and axially. The produced crystal is referred to as an ingot. The temperature of the molten silicon and the vertical pulling speed are used to control the ingot diameter. Stable growing conditions are essential to produce high crystal quality. (v) As the final ingot length is reached, the crystal growth is terminated by slowly decreasing the crystal diameter to zero. During the entire batch process, the crucible is rotated in one direction, and the seed crystal is rotated in the opposite direction.

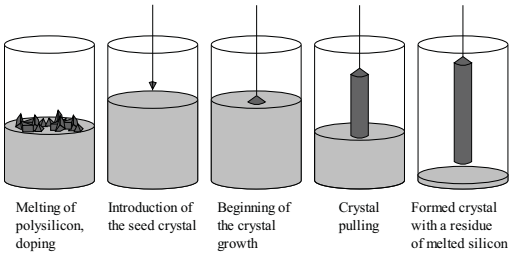


Figure 1: The main batch steps of the CZ process. Illustration from Wikipedia (the illustration is released to public domain by the copyright holder).

Table 1: Notation used in this paper.

$G(z)$	The dynamic model from the graphite pyrometer to the silicon temperature.
$H(z)$	A lowpass Butterworth filter.
k	An index referring to sample number in the dataset.
T	The true, but unknown, temperature of the molten silicon [°C].
$\hat{T}$	The temperature of the molten silicon [°C] estimated by the sensor fusion algorithm.
$\hat{T}^g$	The temperature of the molten silicon [°C] estimated based on the signal from the graphite pyrometer, u.
$\hat{T}^s$	The temperature of the molten silicon [°C] measured by the silicon pyrometer.
t_s	The sample time [s].
u	The raw signal from the graphite pyrometer [V].
$\bar{u}$	$\bar{u}$ is u rescaled by a gain and a bias. $\bar{u}$ is only used for comparing u and $\hat{T}^s$ in plots. $\bar{u}$ is not used in the sensor fusion algorithm.
z	The time-shift operator defined by $x_{k+1} = zx_k$ and $x_{k-1} = z^{-1}x_k$.
ω	Frequency [rad/s].
ω_c	The cut-off frequency of Butterworth filters [rad/s].

SINTEF Materials and Chemistry in Trondheim, Norway, owns and operates a CZ process. Initially there was one pyrometer at this plant. This pyrometer measures the temperature at a graphite material in the CZ process. This pyrometer will be referred to as the graphite pyrometer. The pyrometer has the advantage of little measurement noise, but it does not measure the temperature directly in the molten silicon. Hence, using this pyrometer for temperature control, the temperature of the graphite material is actually controlled, not the temperature of the silicon. This choice of sensor location is then based on the assumption that stable graphite temperature implies stable silicon temperature.

A second pyrometer was installed at the plant. This pyrometer is able to measure the temperature directly in the molten silicon, which is the desired temperature to control. This pyrometer will be referred to as the silicon pyrometer. The authors have logged process data from only one experiment where this pyrometer was tested. The temperature measured by this pyrometer seems reasonable based on the melt temperature of silicon. Also the measured temperature response seems reasonable based on step changes in the heating element power. Unfortunately, the temperature signal of this pyrometer is very noisy. An intuitive and feasible solution is to use a traditional lowpass filter to attenu-

ate the noise. However, lowpass-filtering will give phase lag over the filter which is unfortunate for temperature control. The more the signal is smoothed in the filter, the more phase lag.

4 Sensor Fusion and Complementary Filters

Online measurements of process variables are decisive for most control systems to operate properly. There are often several sensor technologies available for measuring a specific variable. If no conventional sensor is available, it may be possible to develop a soft sensor. A soft sensor does not measure the desired process variable directly, but relies on other measurements and an algorithm to estimate the desired process variable. In the following text, the terms "sensor" and "sensor technology" will be used for both conventional sensors and soft sensors.

For measuring a specific process variable, different sensor technologies may have different qualities, such as accuracy and amount of measurement noise. A sensor fusion algorithm is an algorithm that combines several sensors for estimating a process variable. The algorithm's purpose is to achieve an estimate with better qualities than any of the individual sensors provides. The term "sensor fusion" is a general term for using multiple sensors to estimate a variable. A number of algorithms can be used to fuse the sensor signals, including Kalman filter and Bayesian networks.

A special case of the sensor fusion approach is when different sensors have desirable qualities in different frequency ranges. The typical case is when some sensors are accurate at low frequencies, while other sensors are accurate at high frequencies. A commonly used example is estimation of position based on a position sensor and a velocity sensor. The position sensor may not be sufficiently accurate to keep up with smaller position variations. On the contrary, time-integrating the velocity sensor may keep up with smaller position changes, but this estimate is likely to drift over time due to accumulation of small measurement errors. The intuitive solution is to consider the velocity sensor over short time spans, and consider the position sensor over longer time spans. In the frequency domain this translates into lowpass-filtering the position sensor and highpass-filtering the time-integrated velocity (Brown and Hwang, 1997).

Complementary filters is a simple and intuitive approach for fusing sensors which qualities can be discriminated by frequency. Assume that a process variable y is measured by two sensors. The sensor outputs are $\hat{y}_1 = y + v_1$ and $\hat{y}_2 = y + v_2$, respectively, where v_1 and v_2 are measurement noise / measurement errors. Assume that v_1 is low-frequency noise and v_2 is high-frequency noise. Hence, highpass-filtering $\hat{y}_1$ removes v_1, but also removes the low-frequency components of the signal y. Similarly, lowpass-filtering $\hat{y}_2$ removes v_2, but also removes the high-frequency components of y. Summing the highpass-filtered $\hat{y}_1$ and the lowpass-filtered $\hat{y}_2$ will include both the high-frequency and the low-frequency components of y. Assuming the applied lowpass filter is $H(z)$, the highpass filter is chosen as the complementary filter $1 - H(z)$ (Brown and Hwang, 1997). Complementary filters are illustrated in Figure 2.

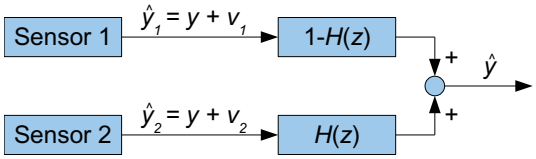


Figure 2: Complementary filters with two inputs. The figure is inspired by Brown and Hwang (1997, Figure 4.9).

The filter output $\hat{y}$ can be written as

$$\begin{aligned} \hat{y} &= [1 - H(z)]\hat{y}_1 + H(z)\hat{y}_2 \\ &= [1 - H(z)](y + v_1) + H(z)(y + v_2) \\ &= y + [1 - H(z)]v_1 + H(z)v_2. \end{aligned} \tag{1}$$

Ideally, the lowpass filter $H(z)$ removes v_2 and the highpass filter $1 - H(z)$ removes v_1. Then the estimated output will be identical to the actual process variable, i.e. $\hat{y} = y$ (Brown and Hwang, 1997).

The choice of the lowpass filter $H(z)$ is important to achieve a good estimate $\hat{y}$. If the noise characteristics of v_1 and v_2 are known, the statistically optimal $H(z)$ can be computed using a Wiener filter or a Kalman filter (Brown and Hwang, 1997). However, in many real-life applications the noise characteristics are not known.

Although the term complementary filters is not used, Ljung (1999, Section 3.3) uses complementary filters when discussing observers and predictors. Actually, this reference was the most inspiring for the authors when developing the sensor fusion algorithm.

5 Basic Principle

This paper presents a sensor fusion algorithm that aims at attenuating measurement noise of the silicon pyrometer, while giving less phase lag than a traditional lowpass filter. Hence, the output of the algorithm, $\hat{T}$, is

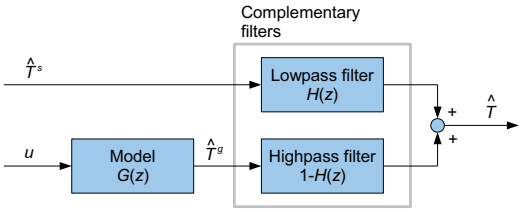


Figure 3: Basic principle of the sensor fusion algorithm.

an estimate of the true, but unknown, silicon temperature, T. This section gives a brief presentation of the algorithm, while the following sections provide a thorough explanation.

The silicon pyrometer measures the temperature of the molten silicon in the CZ process. The pyrometer output signal, $\hat{T}^s$, seems reasonable based on process knowledge and step-response tests. Unfortunately, the signal is very noisy. The graphite pyrometer measures the temperature of a graphite material. The signal from this pyrometer, u, has little noise. There is quite a good correlation between the signals of the two pyrometers.

The sensor fusion algorithm consists of two sub-algorithms: (i) A dynamic model, $G(z)$, and (ii) complementary filters, $H(z)$ and $1 - H(z)$.

The dynamic model, $G(z)$, estimates the silicon temperature, T, as a function of the graphite pyrometer, u. This estimate is noted $\hat{T}^g$. Please note that the estimate does not depend on the silicon pyrometer.

The model estimate, $\hat{T}^g$, is less accurate than the silicon pyrometer, but has the significant advantage of little noise. Due to the measurement noise of the silicon pyrometer, the model estimate is more reliable than the silicon pyrometer, $\hat{T}^s$, at high frequencies. However, the silicon pyrometer is assumed to be more reliable at low frequencies than the model estimate. The complementary filters take advantage of these qualities by lowpass-filtering the silicon pyrometer through $H(z)$, highpass-filtering the model output through $1 - H(z)$, and sum these two filtered signals. The sensor fusion algorithm is illustrated in Figure 3.

6 Presentation of the Raw Data

The authors have access to data from only one CZ batch where both the silicon pyrometer and the graphite pyrometer were used. Analyses of the data and initial tests of the sensor fusion algorithm conclude that the dynamics from the graphite pyrometer, u, to the silicon temperature, T, (measured by the silicon pyrometer, $\hat{T}^s$) is nonlinear. The nonlinearity will be demonstrated in Section 10. The present section covers the data used for developing the sensor fusion algorithm.

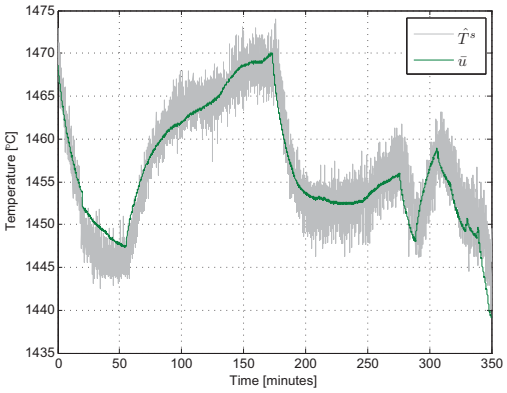


Figure 4: The silicon pyrometer, $\hat{T}^s$, and the rescaled raw signal of the graphite pyrometer, $\bar{u}$, plotted over the data section used to develop the sensor fusion algorithm.

When deciding which part of the dataset to use for developing the sensor fusion algorithm, the authors were looking for a continuous section where there are significant excitations in the temperature, while the temperature range is not wide enough for nonlinearities to be significant. There is only one section in the dataset that meets these demands. This section covers 5 hours and 50 minutes of the CZ batch, having sampling interval of 2 seconds. That is 10501 samples. The grey curve of Figure 4 shows the temperature measured by the silicon pyrometer, $\hat{T}^s$, over the chosen data section. The temperature is in the range of approximately 1445°C to 1470°C. For comparison, during the entire CZ batch (melting of the silicon not included) the temperature varies in the range of approximately 1425°C to 1480°C. Hence, the range used to develop the sensor fusion algorithm is approximately 45% of the total temperature range during the CZ batch. Figure 4 shows that the silicon pyrometer gives a very noisy measurement signal. The noise peak-to-peak amplitude is 5 to 10°C.

The green curve of Figure 4 illustrates the raw signal, u, of the graphite pyrometer over the chosen data section. For the purpose of comparing this signal with the silicon pyrometer, $\hat{T}^s$, the signal has been rescaled to the same range as the silicon pyrometer. The rescaled signal, $\bar{u}$, shown in the figure is on the form $\bar{u} = p_1 u + p_0$, where p_1 and p_0 are polynomial coefficients. The coefficients are computed as the least squares fit of $\bar{u}$

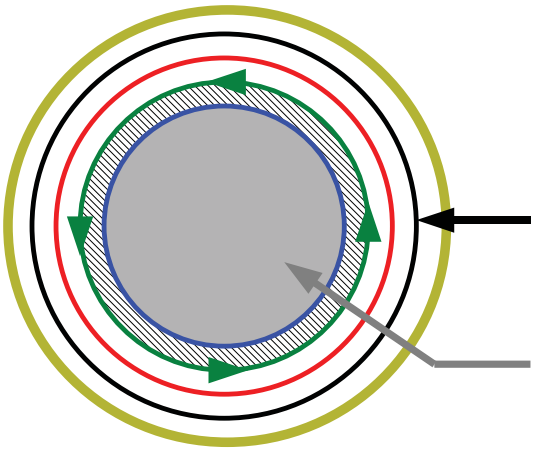


Figure 5: Location of the silicon pyrometer and the graphite pyrometer in the SINTEF CZ process (seen from above). Please refer to the main text for explanation.

to $\hat{T}^s$. Please note that $\bar{u}$ is the *static* best fit of u to $\hat{T}^s$. The *dynamic* relationship between u and $\hat{T}^s$ will be developed in Section 8 using system identification. Figure 4 clearly shows that the graphite pyrometer has very little measurement noise compared to the silicon pyrometer. The figure also shows that there is a good correlation between the silicon pyrometer and the graphite pyrometer.

Figure 5 illustrates the location of the silicon pyrometer and the graphite pyrometer in the CZ process (seen from above). The black arrow indicates the measurement location of the graphite pyrometer, and the grey arrow indicates the measurement location of the silicon pyrometer. The grey area in center represents the molten silicon, and the blue circle represents the crucible. The crucible is located in a rotating device (green color) shaped as a cylinder with bottom, and without top. The red color represents the heating element, which power is controlled by a triode for alternating current (TRIAC). The black color represents the graphite ring at which the graphite pyrometer measures temperature. The yellow color represents insulation and the outer wall. The colors in Figure 5 are chosen to simplify the explanation of the figure. The colors do not represent the colors in the process.

7 Deciding the Complementary Filters $H(z)$ and $1 - H(z)$

Figure 3 illustrates the complementary filters to be used in the sensor fusion algorithm. For two complementary filters, there is only one filter characteristic to choose. For the sensor fusion algorithm, the transfer function of the lowpass filter $H(z)$ is to be chosen. The highpass filter is then given by $1 - H(z)$.

The sensor fusion algorithm provides an estimate, $\hat{T}$, of the silicon temperature. The accuracy of this estimate depends on $H(z)$. $H(z)$ can be computed using a Wiener filter or a Kalman filter (Brown and Hwang, 1997). These approaches give a temperature estimate, $\hat{T}$, that is optimal in the sense of minimizing the estimation error variance, i.e. $E\,(T - \hat{T})^2$. However, the Kalman filter depends on covariance matrices that represent the measurement noise of the silicon pyrometer, i.e. $T - \hat{T}^s$, and the estimation error of the model output, i.e. $T - \hat{T}^g$. The Wiener filter depends on the same information presented in other terms. Unfortunately, this information is not known.

A simpler approach is used in this paper. The filter $H(z)$ is chosen as a Butterworth filter. There are now two parameters to be specified: (i) The cut-off frequency, ω_c, and (ii) the filter order. The cut-off frequency will depend on the tolerance for measurement noise. This tolerance depends on the usage of the temperature estimate, $\hat{T}$. For example, if the estimate is to be used for temperature control, the tolerance for high-frequency noise will depend on the controller tuning. A controller with significant derivative action will have less tolerance than a controller using mainly integral action.

As this paper considers only the sensor fusion algorithm, not its usage, the cut-off frequency, ω_c, will be chosen based on the frequency content of the silicon pyrometer, $\hat{T}^s$. The cut-off frequency is chosen as the lowest frequency component that in the time domain can not be distinguished from high-frequency measurement noise. That is, the logged data of the silicon pyrometer, $\hat{T}^s$, are transformed to the frequency domain using the discrete Fourier transform. Then the lowest frequency component is removed, and the remaining components are transformed back to the time domain. If there is any visible signal pattern beyond measurement noise in the time domain, the second lowest frequency component is removed. Then the remaining frequency components are transformed to the time domain. This is repeated until there is no visual pattern in the time domain beyond measurement noise. For the data sequence $\hat{T}^s$, approximately the 15 lowest frequency components can be distinguished from the high-frequency noise in the time domain.

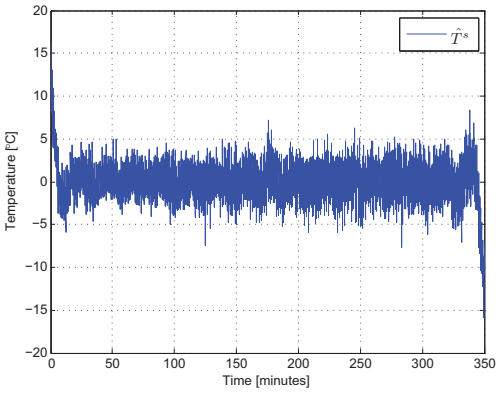


Figure 6: The silicon pyrometer, $\hat{T}^s$, shown in the time domain, after removing the 15 lowest frequency components.

Figure 6 shows the silicon pyrometer, $\hat{T}^s$, in the time domain, after removing the 15 lowest frequency components. Hence, the cut-off frequency, ω_c, will be chosen equal to the lowest frequency component presented in Figure 6. In other words: The cut-off frequency will be chosen to attenuate the data shown in Figure 6. The lowest frequency component that is represented in the figure is the 16th component, which corresponds to 4.8×10^{-3} rad/s. Hence, the cut-off frequency is chosen as $\omega_c = 4.8 \times 10^{-3}$ rad/s.

It is emphasized that this approach is only a suggestion for how to choose the cut-off frequency. When using the sensor fusion algorithm in a real-life application, such as temperature control, the application's tolerance to measurement noise will decide the cut-off frequency. Also, the number of frequency components, in this case 15, has been chosen based on human inspection of data plots, and is therefore not precise.

The next issue is to choose the Butterworth filter order. The higher filter order, the sharper cut-off. Considering the sensor fusion algorithm of this paper, the silicon pyrometer, $\hat{T}^s$, has desirable qualities at low frequencies, and the model estimate, $\hat{T}^g$, has desirable qualities at high frequencies. However, it seems unlikely that there are sharp frequency limits between the desirable and the undesirable qualities. Gradual changes between the qualities seem more likely. Hence, it seems reasonable to choose a low filter order to get a gradual transition from $\hat{T}^s$ to $\hat{T}^g$ for increasing frequencies. The filter order is therefore chosen to be one.

A continuous time lowpass Butterworth filter with $\omega_c = 4.8 \times 10^{-3}$ rad/s has the transfer function (Haugen, 2004)

$$H(s) = \frac{1}{1 + \frac{s}{\omega_c}}. \tag{2}$$

The dynamic model, $G(z)$, to be developed in Section 8 will be a discrete time model. It is therefore desirable to also have the lowpass filter in discrete time. There are several ways to convert a continuous time transfer function to discrete time. A method referred to as the bilinear transform method will be used here. This method is described in Lyons (2011). The method is to replace s in a continuous transfer function, $H(s)$, with

$$s = \frac{2}{t_s}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right) \tag{3}$$

to obtain the time discrete transfer function $H(z)$. Here, t_s is the sampling time, which is 2 seconds. Using this method, the discrete time lowpass filter becomes

$$H(z) = \frac{\omega_c t_s + \omega_c t_s z^{-1}}{\omega_c t_s + 2 + (\omega_c t_s - 2)z^{-1}}. \tag{4}$$

The complementary continuous highpass filter is

$$1 - H(s) = \frac{\frac{s}{\omega_c}}{1 + \frac{s}{\omega_c}}. \tag{5}$$

This is a highpass Butterworth filter (Haugen, 2004). In discrete time this becomes

$$1 - H(z) = \frac{2 - 2z^{-1}}{\omega_c t_s + 2 + (\omega_c t_s - 2)z^{-1}}. \tag{6}$$

Figure 7 shows the Bode diagram of the complementary filters $H(z)$ and $1 - H(z)$.

8 Identifying the Dynamic Model $G(z)$

The sensor fusion algorithm presented in this paper aims at attenuating the measurement noise of the silicon pyrometer, while giving significant less phase lag over the filter than a traditional lowpass filter. Figure 3 illustrates the sensor fusion algorithm. The upper input of the summation point represents a traditional lowpass-filtering of the silicon pyrometer, $\hat{T}^s$. This signal is believed to give an accurate representation of the silicon temperature, T, but the high-frequency components are removed and the signal is phase lagged in the filter. Hence, the purpose of the lower input of the summation point in Figure 3 is to estimate the high-frequency components of the silicon temperature, T, and give this estimate a positive phase by highpass-filtering it. Therefore, the success of the sensor fusion

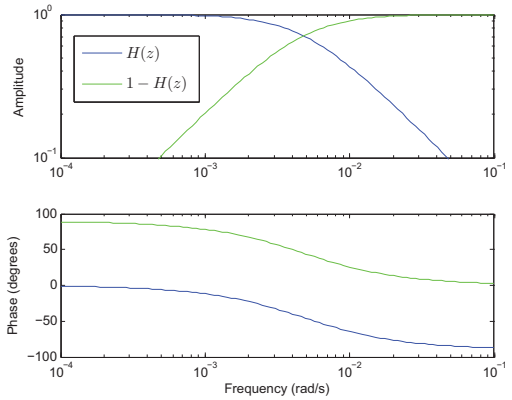


Figure 7: Bode diagram of the complementary filters; the lowpass filter, $H(z)$, and the highpass filter, $1-H(z)$.

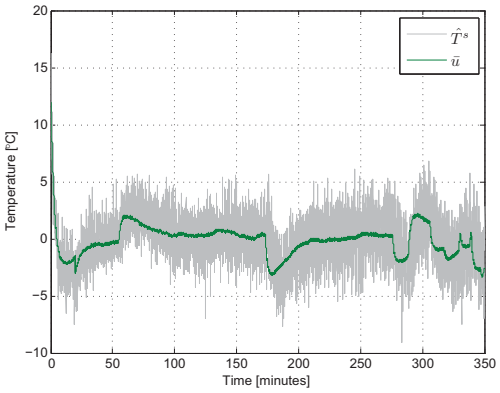


Figure 8: The silicon pyrometer, $\hat{T}^s$, and the rescaled graphite pyrometer, $\bar{u}$, after being detrended and highpass-filtered through $1-H(z)$.

algorithm depends on how accurate the model output, $\hat{T}^g$, describes the silicon temperature, with emphasis to the higher frequency components. This section discusses how the model, $G(z)$, is developed to meet this demand.

The model, $G(z)$, will be developed using system identification. The model input is the output signal of the graphite pyrometer, u. The desired model output is the silicon temperature, T. As the exact temperature is not known, the best estimate available is the temperature measured by the silicon pyrometer, $\hat{T}^s$. Unfortunately, the noise at $\hat{T}^s$ is likely to reduce the quality of the model. However, no better options are available.

In the sensor fusion algorithm, the model's output, $\hat{T}^g$, will be highpass-filtered through the filter $1-H(z)$ as shown in Figure 3. Hence, when developing the model, $G(z)$, the low-frequency components should be deemphasized in favor of the high-frequency ones. This is achieved by detrending (subtracting sample mean) and highpass-filtering the raw data using the same highpass filter that is used in the complementary filters, i.e. $1-H(z)$. Figure 8 shows the silicon pyrometer, $\hat{T}^s$, and the rescaled version of the graphite pyrometer signal, $\bar{u}$, after being detrended and filtered through the highpass filter $1-H(z)$. In other words: The data shown in Figure 8 are the data shown in Figure 4 after being detrended and highpass-filtered. When identifying the model, $G(z)$, the input data used is u and the output data used is $\hat{T}^s$, both after being detrended and highpass-filtered. That is, the data used for identifying the model are the data shown in Figure 8, except that u is used instead of its rescaled version $\bar{u}$.

The next issue is to decide a model structure for the model $G(z)$. The dynamics from the graphite pyrometer, u, to the silicon temperature, T, (measured by the silicon pyrometer, $\hat{T}^s$) seems to be nonlinear. This nonlinearity is demonstrated in Section 10. Flexible, nonlinear black-box model structures usually contain significantly more parameters than linear model structures. As the authors do not have independent data for validating the model, the number of parameters should be limited to avoid overfitting of the model. Therefore, a simple, linear model structure was chosen for the sensor fusion algorithm presented in this paper. Section 11 discusses further work, including this nonlinearity issue.

As pointed out in Ljung (1999), including a noise model in the model structure is equivalent to prefilter the measured data. Hence, a noise model may counteract the data prefiltering done above. Therefore, an output error (OE) model structure will be used. OE models have no noise model, i.e. the noise model is simply 1. The MATLAB System Identification Toolbox includes two linear OE model structures: (i) The process model structure and (ii) the polynomial OE model structure.

The process model is a time continuous transfer function. The human model builder can specify the number of poles (one to three), whether the transfer function should have a zero, and whether the transfer function should have a time delay. Using the maximum number of poles, and a zero and a time delay, the model structure is on the form

$$G(s) = K\frac{1+t_z s}{(1+t_1 s)(1+t_2 s)(1+t_3 s)}e^{-t_d s}, \quad (7)$$

where the parameters to be identified are K, t_1, t_2, t_3, t_z, and t_d. As pointed out in Ljung (2009), the main advantage of this model structure is that the time delay is estimated. For most other model structures the time delay must be specified by the human model builder. On the other hand, a disadvantage of the process model structure is that the human model builder must specify whether or not the system is underdamped (has complex conjugate poles). If the system is underdamped, the model structure is on the form

$$G(s) = K\frac{1+t_z s}{(1+2\zeta t_\omega s+(t_\omega s)^2)(1+t_3 s)}e^{-t_d s}. \quad (8)$$

Please refer to Ljung (2009) for further explanation of the process model structure.

The polynomial OE model structure is a model structure in the same family as the more well-known ARX and ARMAX model structures. For single input, single output (SISO) systems, the only difference between these three model structures is the noise models. The OE model has no noise model, i.e. the noise model is simply 1. The SISO polynomial OE model structure is on the form

$$G(z) = \frac{B(z)}{F(z)}z^{-n_d}, \quad (9)$$

where

$$B(z) = \sum_{i=0}^{n_b} b_i z^{-i}, \quad (10)$$

$$F(z) = 1+\sum_{i=1}^{n_f} f_i z^{-i}. \quad (11)$$

Here, b_i and f_i are polynomial coefficients, and n_d is the time delay in number of samples.

The parameters of the model structures (7) (or (8)) and (9) are identified using the prediction error method (PEM). For an introduction to PEM, please refer to Ljung (1999). Using PEM identification, the model parameters are identified as the solution of a multivariable optimization problem. In most cases the optimization problem is nonlinear, and the optimization algorithm is in danger of being trapped in a local minimum. Quoting Ljung (1999): "For output error structures, on the other hand, convergence to false local minima is not uncommon."

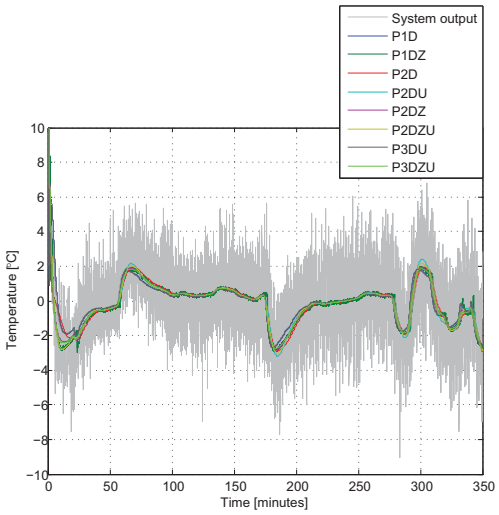


Figure 9: The eight successful process models. 'P1', 'P2', and 'P3' refer to the number of poles (one, two, or three). 'D' means that time delay is estimated. 'Z' means that a zero is estimated. 'U' means that two poles are underdamped (complex conjugate).

The identification work was started by identifying process models ((7) and (8)) using the MATLAB System Identification Toolbox. Ten models with different number of poles, real or complex poles, and with or without zero were identified. The models were then plotted against the system output, i.e. the highpass-filtered $\hat{T}^s$. Eight of the ten models give good fit to the system output. It is assumed that the failure of the other two models is because the PEM algorithm was trapped in a local minimum. The argument for this conclusion is that simpler model structures, which are subsets of the faulty model structures, did succeed. The eight successful models are shown in Figure 9. There are some differences in the initial values, i.e. the first 20-30 minutes. There are also some smaller differences in the range 325 to 345 minutes (this may be difficult to see in the figure). However, Figure 9 shows that the models are very similar in explaining the system output, and it is very difficult to conclude which model is the better one.

The main reason for beginning the identification work by identifying process models is the process model structure's ability to estimate time delay, i.e. t_d in (7) and (8). Among the eight successful models, t_d ranges from

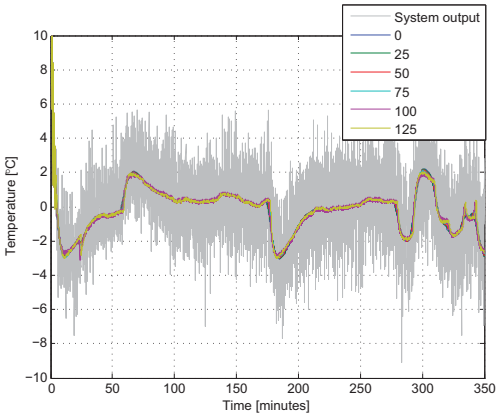


Figure 10: Six polynomial OE models with $n_b = 1$ and $n_f = 2$. The time delay, n_d, ranges from 0 to 125 samples in increments of 25 samples.

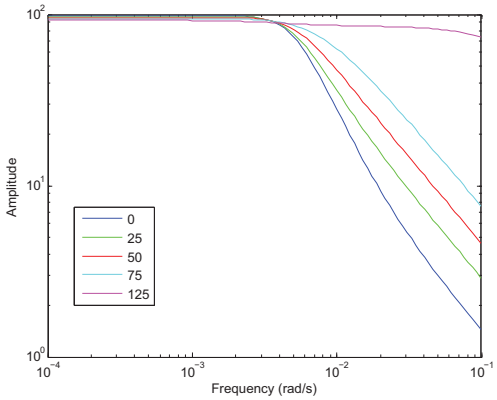


Figure 11: Bode magnitude diagram of the five successful polynomial OE models. The models' time delay, n_d, range from 0 to 125 samples in increments of 25 samples. The model having $n_d = 100$ is excluded.

0 to 210 seconds. Hence, it must be concluded that using process models for estimating time delay was not very successful for this dataset. However, this is not an unfortunate conclusion: Models with a wide range of time delays give good fit to the system output. Hence, the models seem robust to the choice of time delay. In other words: The poles and the zero seem to be able to compensate a somewhat erroneous time delay. The robustness to the choice of time delay was confirmed by polynomial OE models: Six models with $n_b = 1$ and $n_f = 2$ were identified. The models have different time delays, n_d, ranging from 0 to 125 samples in increments of 25 samples (i.e. from 0 to 250 seconds in increments of 50 seconds). Otherwise these six models were identified under the same conditions. Figure 10 shows the model fit of these six models. Figure 10 confirms the observation of Figure 9: A wide range of time delays give good model fit. Hence, it is concluded that the models are robust to the choice of time delay.

The polynomial OE model with $n_d = 100$ gives an unusual step-response and frequency response (not shown). The model will therefore be ignored in the following discussion.

Zooming in on Figure 10 shows that some models give smooth outputs, while other models give outputs with some high-frequency noise. It turns out that longer time delay gives more noisy output. The reason is to be found in the Bode magnitude diagram of the models. This diagram is shown in Figure 11. The longer time delay, n_d, the higher amplitude at high frequencies. Hence, longer time delay means less lowpass-filtering of the model input signal, u. This explains why longer time delay gives more noisy output.

It remains to be explained *why* longer time delays give less attenuation for increasing frequencies. Visual inspection of Figure 8 strongly indicates that there is some phase lag from u to $\hat{T}^s$, because the peaks in $\bar{u}$ seem to occur before the corresponding peaks in $\hat{T}^s$. Long time *delay* gives large phase lag in the frequency domain. Long time *constants* also give large phase lag. Hence, it seems reasonable to conclude that models with long time delay use the time delay to give phase lag, while models with short time delay use long time constants to give phase lag. Long time delays do not change the magnitude of the Bode diagram, while long time constants attenuate high frequencies.

Quoting Ljung (1999): "Our acceptance of models should be guided by 'usefulness' rather than 'truth'." For the purpose of the sensor fusion algorithm, it is not of main interest to know the exact time delay. The implications of short or long time delay are more important. Hence, it seems reasonable to decide whether or not a lowpass-filtering effect in the model $G(z)$ is desirable, and choose the time delay based on this decision. There is some noise present in the signal from the graphite pyrometer, u. Hence, it is reasonable to require at least some lowpass-filtering. From a physical consideration of the process, it is reasonable to assume that the melted silicon in the crucible heats and cools off slower than the graphite ring. This further favors lowpass-filtering. The time delay is therefore chosen as $n_d = 0$. It is emphasized that this is not an exact scientific conclusion, but a choice that is reasonable based on the need for lowpass-filtering due to some measure-

ment noise in the graphite pyrometer output, u, as well as physical consideration of the process.

The next issue is to choose the polynomial orders n_b and n_f. The models shown in Figure 10 have $n_b = 1$ and $n_f = 2$, i.e. there are two parameters to be identified in each of the nominator and denominator. As the models give good fit, it is reasonable to assume that these polynomial orders are quite good. However, various polynomial orders will be tested to find the optimal ones. Polynomial OE models were identified with all combinations of $n_b \in \{0, 1, 2\}$ and $n_f \in \{1, 2, 3\}$, i.e. in total nine models. For all models the time delay is $n_d = 0$. The fit of these nine models are shown in Figure 12. It may be difficult to separate the nine models in the figure. However, this is not essential. The main point of the figure is that the models form three groups. The first group is all models with $n_f = 1$ (regardless of n_b). The second group is $n_b = 0$ and $n_f \in \{2, 3\}$. The third group is the remaining four models, i.e. $n_b \in \{1, 2\}$ and $n_f \in \{2, 3\}$. Hence, it seems reasonable to conclude that for models with $n_b = 0$ and/or $n_f = 1$, there is some dynamics that is not properly modeled due to too few polynomial parameters. However, the modeled dynamics does not change significantly as n_b increases from 1 to 2 (provided $n_f \geq 2$). Similarly, the dynamics does not change significantly as n_f increases from 2 to 3. It is therefore concluded that $n_b = 1$ and $n_f = 2$ are sufficiently high polynomial orders.

Summing up the identification work: The final model $G(z)$ is chosen as a polynomial OE model, (9), with $n_b = 1$, $n_f = 2$, and $n_d = 0$. Hence, the model is on the form

$$G(z) = \frac{b_0 + b_1 z^{-1}}{1 + f_1 z^{-1} + f_2 z^{-2}}. \tag{12}$$

The model has four parameters to be identified. The final model is shown in Figure 13. The final model turns out to be identical to the model labeled "0" in Figure 11. This figure shows the model's Bode magnitude diagram.

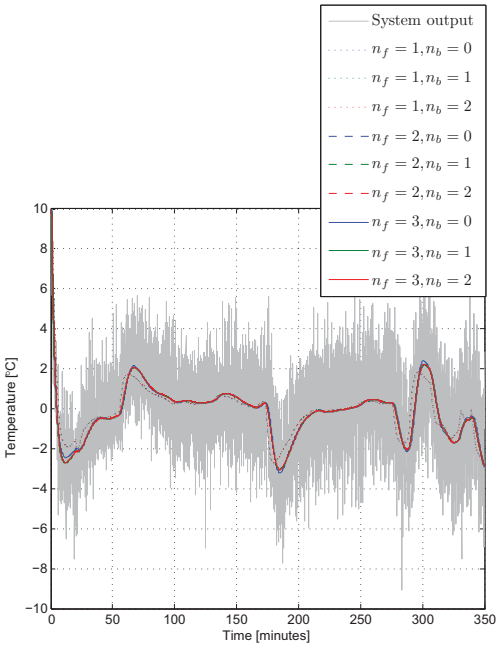


Figure 12: Polynomial OE models with all combinations of $n_b \in \{0, 1, 2\}$ and $n_f \in \{1, 2, 3\}$. The time delay is $n_d = 0$ for all models.

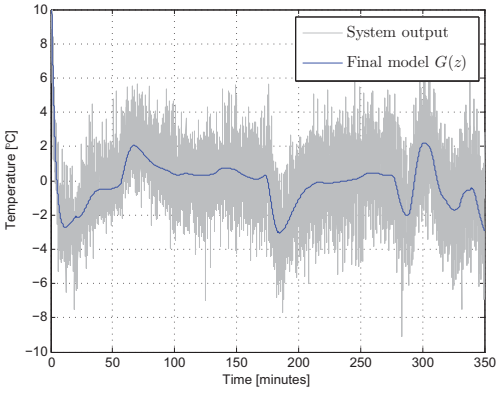


Figure 13: The final model $G(z)$.

9 Merging the Sub-Algorithms

As shown in Figure 3, the sensor fusion algorithm consists of two sub-algorithms: (i) The complementary filters, $H(z)$ and $1 - H(z)$, which filter characteristics were developed in Section 7, and (ii) the dynamic model, $G(z)$, which was developed in Section 8. These sub-algorithms are now to be merged into the final sensor fusion algorithm. The output of the sensor fusion algorithm is the silicon temperature estimate, $\hat{T}$, which is a function of the silicon pyrometer, $\hat{T}^s$, and the graphite pyrometer, u. The estimate is given by

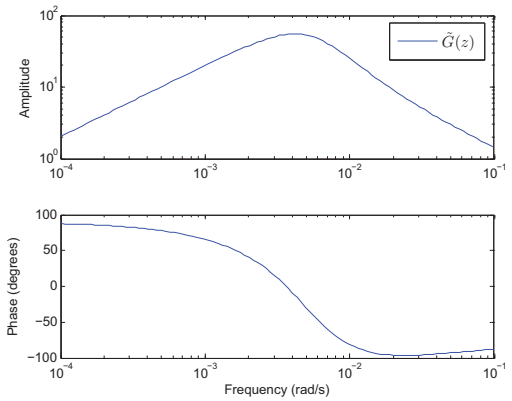


Figure 14: Bode diagram of $\tilde{G}(z)$.

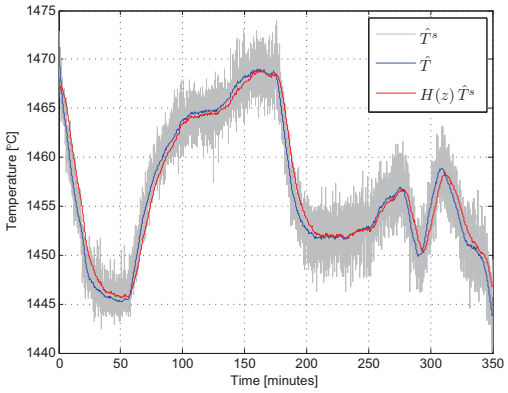


Figure 15: The output of the sensor fusion algorithm, $\hat{T}$, and the lowpass-filtered silicon pyrometer, $H(z)\,\hat{T}^s$, plotted against the unfiltered silicon pyrometer, $\hat{T}^s$.

$$\begin{aligned}\hat{T} &= H(z)\,\hat{T}^s + (1 - H(z))\,\hat{T}^g \\ &= H(z)\,\hat{T}^s + (1 - H(z))\,G(z)\,u.\end{aligned} \tag{13}$$

It is convenient to define

$$\tilde{G}(z) \stackrel{\text{def}}{=} (1 - H(z))\,G(z). \tag{14}$$

The estimated silicon temperature can then be written

$$\hat{T} = H(z)\,\hat{T}^s + \tilde{G}(z)\,u. \tag{15}$$

The transfer function $H(z)$ is a lowpass filter. The Bode diagram of $H(z)$ is shown in Figure 7. The transfer function $\tilde{G}(z)$ is a series connection of the model $G(z)$, which has lowpass characteristics, and the highpass filter $1 - H(z)$. Hence, $\tilde{G}(z)$ has bandpass characteristics. The Bode diagram of $\tilde{G}(z)$ is shown in Figure 14.

10 Validating the Sensor Fusion Algorithm

The purpose of the sensor fusion algorithm is to filter the measurement noise of the silicon pyrometer, $\hat{T}^s$, with significant less phase lag than a traditional lowpass filter. It is then natural to compare the output of the sensor fusion algorithm, $\hat{T}$, with the output of the lowpass filter $H(z)$. That is, the sensor fusion algorithm, $\hat{T} = H(z)\,\hat{T}^s + \tilde{G}(z)\,u$, is to be compared with $H(z)\,\hat{T}^s$. Figure 15 shows $\hat{T}$ and $H(z)\,\hat{T}^s$ plotted against the unfiltered silicon pyrometer signal, $\hat{T}^s$. Figure 16 shows the same data as Figure 15, zoomed in at three different areas. The figures show that the sensor fusion algorithm, $\hat{T}$, follows the unfiltered signal, $\hat{T}^s$, much closer than the lowpass-filtered value $H(z)\hat{T}^s$ does.

The output of the sensor fusion algorithm, $\hat{T}$, consists of two contributions: $H(z)\,\hat{T}^s$ and $\tilde{G}(z)\,u$. Visual comparison of the two contributions shows that the high-frequency noise at $\tilde{G}(z)\,u$ is neglectable compared to $H(z)\,\hat{T}^s$ (this plot is not shown). Hence, $\hat{T}$ has the same amount of high-frequency noise as $H(z)\,\hat{T}^s$.

Figures 15 and 16 compare $\hat{T}$ and $H(z)\,\hat{T}^s$ in the time domain. It is also of interest to compare these two signals in the frequency domain. However, it is not straight forward how to do this. The following approach has been chosen here: Two models, $R(z)$ and $S(z)$, are identified. $R(z)$ models the dynamics from $\hat{T}^s$ to $\hat{T}$, and $S(z)$ models the dynamics from $\hat{T}^s$ to $H(z)\,\hat{T}^s$. Both models are on the form (9) with $n_b = 1$, $n_f = 2$, and $n_d = 0$. The intention of the models is to get an estimate of which frequency components of the silicon temperature, T, (measured by the silicon pyrometer, $\hat{T}^s$) that are preserved in $\hat{T}$ and $H(z)\,\hat{T}^s$, respectively. Figure 17 shows the Bode diagram of the models $R(z)$ and $S(z)$. As expected, the model $S(z)$ is identical to the lowpass filter $H(z)$. The Bode diagram shows that $S(z)$ attenuates at much lower frequencies than $R(z)$. This means that the sensor fusion algorithm preserves higher frequencies of the silicon temperature, T, than the lowpass filter $H(z)$ does, without letting through more noise. The phase diagram shows that the sensor fusion algorithm can handle much higher frequencies before it gives significant phase lag. The magnitude diagram indicates that

the sensor fusion algorithm slightly amplifies some frequency components around $\omega = 10^{-2}$ rad/s. This is an undesirable behavior.

Figures 15 through 17 show that the sensor fusion algorithm is successful when validated on the same measurement data that were used to identify the model $G(z)$. However, the dynamics from the graphite pyrometer output, u, to the silicon temperature, T, (measured by the silicon pyrometer, $\hat{T}^s$) is nonlinear. The authors still chose to use a simple, linear model with few parameters to avoid overfitting the model. It is now of interest to validate how well the sensor fusion algorithm performs when tested on measurement data in a different temperature range. First the raw data will be presented. Figure 4 shows the 350 minutes of raw data used to identify the model $G(z)$. In this figure, the graphite pyrometer output, u, is replaced by a rescaled version $\bar{u}$. The relationship between u and $\bar{u}$ is a first order polynomial, i.e. $\bar{u} = p_1 u + p_0$. The polynomial coefficients p_0 and p_1 were identified as the best (in a least squares sense) static fit between u and the silicon pyrometer, $\hat{T}^s$, over the data shown in Figure 4. Figure 18 shows the 350 minutes of logged data shown in Figure 4 and the following 150 minutes. While $\bar{u}$ and $\hat{T}^s$ follow closely over the first 350 minutes, i.e. the data range used to identify the polynomial coefficients p_0 and p_1, $\bar{u}$ and $\hat{T}^s$ deviate significantly over the last 150 minutes. The authors can not see any other explanation of this deviation than that the temperature is significantly lower over the last 150 minutes. If this assumption is correct, the relationship between $\bar{u}$ and $\hat{T}^s$ must be significantly nonlinear.

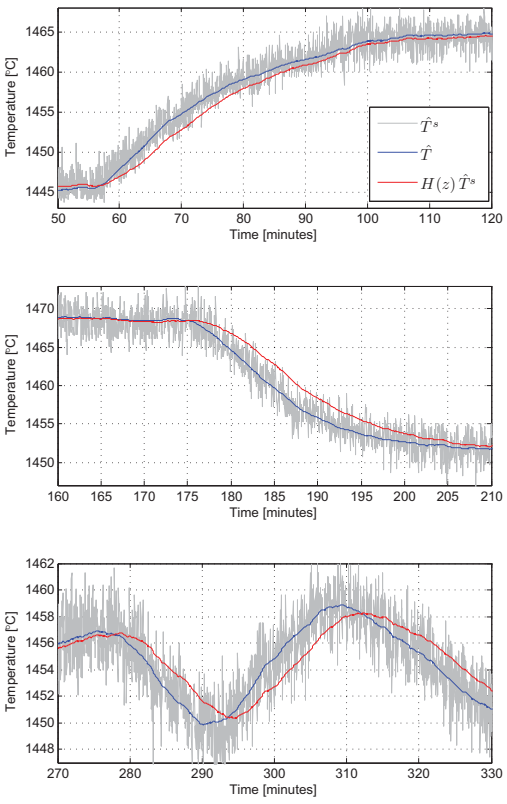


Figure 16: The same data shown in Figure 15, zoomed in at three different areas.

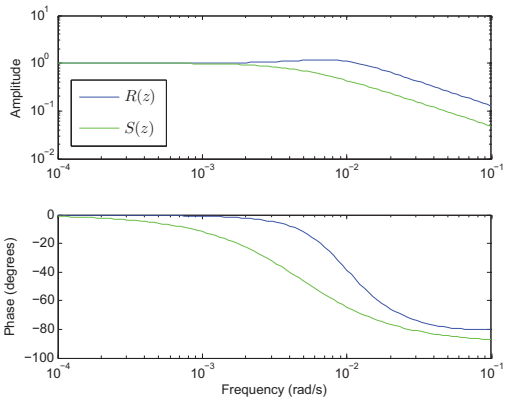


Figure 17: Bode diagram of the models $R(z)$ and $S(z)$.

Based on Figure 18, one can not expect the linear model, $G(z)$, which is developed based on the first 350 minutes, to perform well over the last 150 minutes. Figure 19 compares the output of the sensor fusion algorithm, $\hat{T}$, and the lowpass-filtered silicon pyrometer, $H(z)\,\hat{T}^s$, over the last 150 minutes of Figure 18. The sensor fusion algorithm performs poorly until approximately 390 minutes. This is because there is a large temperature drop from 310 minutes to 380 minutes that the linear model, $G(z)$, is not able to handle properly. However, it seems that the sensor fusion algorithm performs quite well after approximately 390 minutes, when the temperature settles at a new, lower level. This is because the highpass filter $1 - H(z)$ of $\tilde{G}(z)$ removes any static or low-frequency error of the temperature estimate $\hat{T}^g$. As the model, $G(z)$, is not identified for the low temperature range between 390 and 500 minutes, the performance of the sensor fusion algorithm may be poorer than in the temperature range used for identification.

Figures 15 and 16 give a visual impression of how

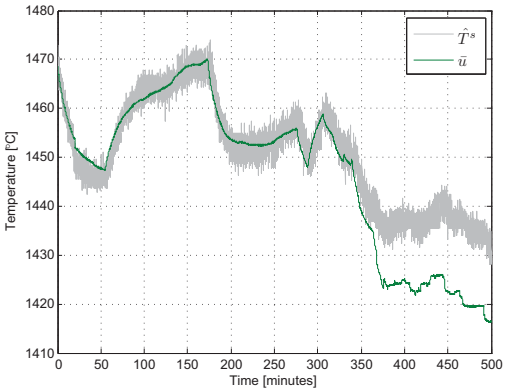


Figure 18: The rescaled graphite pyrometer, $\bar{u}$, and the silicon pyrometer, $\hat{T}^s$, over an extended time period.

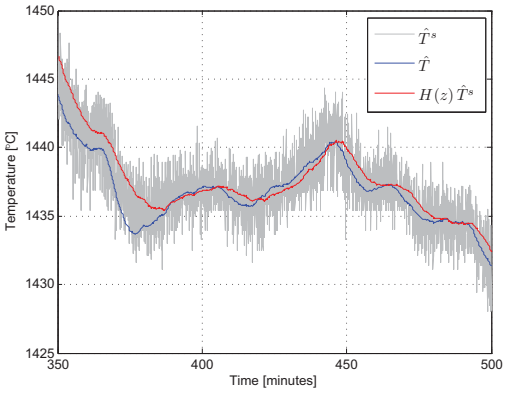


Figure 19: The output of the sensor fusion algorithm, $\hat{T}$, the lowpass-filtered silicon pyrometer, $H(z)\,\hat{T}^s$, and the unfiltered silicon pyrometer, $\hat{T}^s$, over an extended time period.

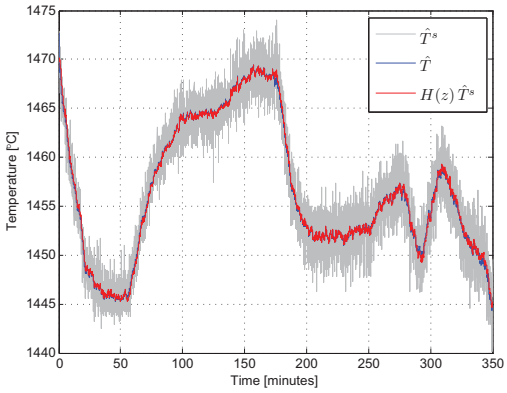


Figure 20: The output of the sensor fusion algorithm, $\hat{T}$, and the lowpass-filtered silicon pyrometer, $H(z)\,\hat{T}^s$, for cut-off frequency $\omega_c = 2.4 \times 10^{-2}$ rad/s.

much the sensor fusion algorithm improves the temperature estimate compared to a traditional lowpass filter. However, how much the sensor fusion algorithm improves the temperature estimate highly depends on the cut-off frequency of the complementary filters $H(z)$ and $1 - H(z)$. Figure 20 compares the output of the algorithm, $\hat{T}$, and the output of the lowpass filter, $H(z)\,\hat{T}^s$, when the cut-off frequency is increased by a factor of five, i.e. from $\omega_c = 4.8 \times 10^{-3}$ rad/s to $\omega_c = 2.4 \times 10^{-2}$ rad/s. Increasing the cut-off frequency is equivalent to increasing the tolerance for high-frequency noise. Figure 20 shows that the curves for $\hat{T}$ and $H(z)\,\hat{T}^s$ are almost identical. Hence, there is no significant improvement of using the sensor fusion algorithm over the traditional lowpass filter for this choice of ω_c. Comparing Figures 15 and 20 shows that there is much more high-frequency noise at $\hat{T}$ and $H(z)\,\hat{T}^s$ in the latter figure.

On the other hand, decreasing the cut-off frequency, i.e. having less tolerance for measurement noise, the improvement of using the sensor fusion algorithm, compared to a traditional lowpass filter, is much larger. Figure 21 compares $\hat{T}$ and $H(z)\,\hat{T}^s$ when the cut-off frequency is decreased by a factor of five, i.e. from $\omega_c = 4.8 \times 10^{-3}$ rad/s to $\omega_c = 9.6 \times 10^{-4}$ rad/s. For this cut-off frequency the improvement of using the sensor fusion algorithm is very large. There is hardly any visible high-frequency noise at neither $\hat{T}$ nor $H(z)\,\hat{T}^s$. For the simulations shown in Figures 20 and 21, the raw data were prefiltered with the chosen cut-off frequencies, and the model $G(z)$ was re-identified.

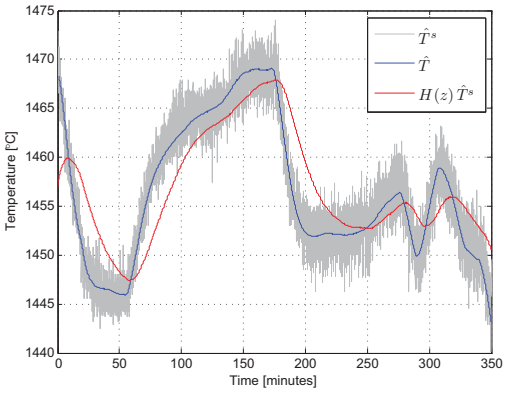


Figure 21: The output of the sensor fusion algorithm, $\hat{T}$, and the lowpass-filtered silicon pyrometer, $H(z)\,\hat{T}^s$, for cut-off frequency $\omega_c = 9.6 \times 10^{-4}$ rad/s.

11 Algorithm Weaknesses and Further Work

The sensor fusion algorithm presented in this paper is developed based on logged measurement data from a single CZ batch. Therefore, the algorithm has been validated based on the same data used for developing the algorithm. A simple, linear model with few parameters was chosen to model the dynamics from the graphite pyrometer to the silicon temperature. The model structure was chosen to avoid overfitting, because flexible, nonlinear black-box model structures typically have significantly more parameters. However, the dynamics is known to be nonlinear (see Figure 18). An important part of further work will therefore be to understand this nonlinear dynamics, either physically or empirically, and build this into the model. As the CZ process is a batch process, the dynamics may also be time-varying. In particular, it seems reasonable to assume that the dynamics may vary with the level of molten silicon in the crucible. Preferably, the model should be developed based on data from several CZ batches to make sure the data are representative and sufficiently informative. Also, the algorithm should be validated based on data from several *independent* CZ batches.

According to Brown and Hwang (1997), the complementary filters $H(z)$ and $1 - H(z)$ can be computed to give a statistically optimal temperature estimate, i.e. to minimize the variance of the estimation error. A Kalman filter or a Wiener filter can be used for this computation. Even though the complementary filters can be chosen optimally in theory, modeling errors and estimation errors in the noise / disturbance covariance matrices are likely to give a sub-optimal temperature estimate. Also, even though the temperature estimate is in fact optimal, it may be too noisy for its desired application. However, even if there are practical issues related to using Kalman filter or Wiener filter, this is an interesting issue to consider for further work.

The lowpass filter removes high-frequency noise from the silicon pyrometer. The graphite pyrometer, in combination with the dynamic model, is used to estimate those frequency components of the silicon temperature that are removed by the lowpass filter. A disadvantage of this approach is that it requires two pyrometers. The heating element heats the entire CZ process, including the crucible and the silicon. It may be possible to replace the graphite pyrometer with the measured heating element power. That is, to develop a model from the heating element power to the silicon temperature, and use this model to estimate the high-frequency components of the silicon temperature that are removed when lowpass-filtering the silicon pyrometer.

12 Conclusions

SINTEF Material and Chemistry operates a Czochralski (CZ) crystallization process. During one CZ batch, two pyrometers were used: The silicon pyrometer measures the temperature of the molten silicon. This pyrometer is assumed to be accurate, but its output signal has much high-frequency noise. The noise can be attenuated using a traditional lowpass filter. However, this approach will give a phase lag that is unfortunate for the temperature control. The graphite pyrometer measures the temperature of a graphite material. Hence, the graphite pyrometer does not give an exact representation of the silicon temperature. However, the graphite pyrometer has little measurement noise. There is quite a good correlation between the silicon pyrometer and the graphite pyrometer.

This paper presents a sensor fusion algorithm that attenuates the measurement noise of the silicon pyrometer, while giving significant less phase lag than a traditional lowpass filter. The algorithm consists of two sub-algorithms: (i) A dynamic model and (ii) complementary filters.

The dynamic model estimates the silicon temperature as a function of the graphite pyrometer. The model is a linear output error (OE) model with four parameters. The parameters were identified using the prediction error method (PEM), where the graphite pyrometer is the system input and the silicon temperature (measured by the silicon pyrometer) is the

system output. A linear model structure was chosen despite the fact that the dynamics is known to be nonlinear. Flexible, nonlinear black-box model structures typically have significantly more parameters than linear model structures. As no independent data were available for model validation, it was desirable to use a model structure with few parameters to avoid model overfitting.

A lowpass filter and a highpass filter are designed as complementary filters. The silicon pyrometer is lowpass-filtered, and the output of the OE model is highpass-filtered. These two filtered signals are then summed. This sum is the output of the sensor fusion algorithm, i.e. the estimated silicon temperature. In other words: The lowpass filter attenuates noise from the silicon pyrometer, while the OE model estimates the high-frequency components of the silicon temperature that are lost in the lowpass-filtering.

Validation of the sensor fusion algorithm shows that it works well on the data that were used to identify the OE model. The algorithm gives significantly less phase lag than traditional lowpass-filtering of the silicon pyrometer. The algorithm performs poorly when there are large, quick temperature changes outside the temperature range used for model identification. This is to be expected, because the linear model can not handle the nonlinear dynamics between the graphite pyrometer and the silicon temperature. The algorithm seems to perform quite well when the temperature is within a limited temperature range, even if this range is outside the temperature range used for model identification. The usefulness of the model depends on the choice of cut-off frequency of the filters. For high cut-off frequencies, the algorithm gives little or no improvement. For low cut-off frequencies, the algorithm gives a large improvement.

Further work on the algorithm should include analysis of the nonlinear dynamics from the graphite pyrometer to the silicon temperature, and build this into the dynamic model. This is likely to improve the algorithm's ability to handle large, quick temperature variations. The suggested analysis and modeling work, as well as more thorough validation of the algorithm, require logged data from more CZ batches.

Acknowledgements

The authors are most grateful to SINTEF Materials and Chemistry in Trondheim, Norway, for giving access to data from the company's CZ process, and for allowing these data to be used in this paper and other publications. The authors are also very grateful to NorSun AS for providing financial support to the CZ experiment which this paper is based on.

The first author is most grateful for the financial support of his PhD study from NorSun AS, Østfold Energi AS, and the Norwegian Research Council. The cooperation with NorSun AS, Prediktor AS, and SINTEF Materials and Chemistry is also very much appreciated.

References

Brown, R. G. and Hwang, P. Y. C. *Introduction to random signals and applied Kalman filtering - 3rd edition*. John Wiley & Sons Inc., 1997.

Haugen, F. *Dynamic Systems - Modeling, Analysis and Simulation*. Tapir Academic Press, Trondheim, Norway, 2004.

Irizarry-Rivera, R. and Seider, W. D. Model-predictive control of the Czochralski crystallization process Part I. Conduction-dominated melt. *Journal of Crystal Growth*, 1997a. 178:593–611. doi:10.1016/S0022-0248(97)00085-7.

Irizarry-Rivera, R. and Seider, W. D. Model-predictive control of the Czochralski crystallization process Part II. Reduced-order convection model. *Journal of Crystal Growth*, 1997b. 178:612–633. doi:10.1016/S0022-0248(97)00086-9.

Lan, C. W. Recent progress of crystal growth modeling and growth control. *Chemical Engineering Science*, 2004. 59:1437–1457. doi:10.1016/j.ces.2004.01.010.

Lee, K., Lee, D., Park, J., and Lee, M. MPC based feedforward trajectory for pulling speed tracking control in the commercial Czochralski crystallization process. *International Journal of Control, Automation, and Systems*, 2005. 3:252–257.

Ljung, L. *System Identification - Theory for the User, 2nd Edition*. Prentice-Hall, Upper Saddle River, New Jersey, 1999.

Ljung, L. *System Identification Toolbox 7 - User's Guide*. The MathWorks, Inc., 2009.

Lyons, R. G. *Understanding digital signal processing - 3rd edition*. Prentice Hall, 2011.

ISBN 978-82-7206-336-7 (printed version)
ISBN 978-82-7206-337-4 (electronic version)
ISSN 1893-3068

www.hit.no
2012