Reduction and control for PDE models of manufacturing systems

D. Onck

SE 420 478

Master's Thesis

Supervisor: Prof.dr.ir. J.E. Rooda
Coach: Dr.ir. A.A.J. Lefeber

Eindhoven University of Technology
Department of Mechanical Engineering
Systems Engineering Group

Eindhoven, June 2006

ASSIGNMENT

EINDHOVEN UNIVERSITY OF TECHNOLOGY June 2005
Department of Mechanical Engineering
Systems Engineering Group

Student	D. Onck
Supervisor	Prof.dr.ir. J.E. Rooda
Advisor	Dr.ir. A.A.J. Lefeber
Start	April 2005
Finish	April 2006
Title	Control of manufacturing systems using data based identification of PDE models

Subject

Recent research on control of manufacturing systems focusses on the use of partial differential equations (PDE's) to describe the behavior of manufacturing systems.
Numerical methods for solving PDE's involve discretization with respect to the spatial domain into grid cells. For large scale problems or fine discretized PDE's this results in large sets of equations as the governing equations need to be solved for every single grid cell.
A method used to reduce the complexity of numerically solving PDE based models is the method of Proper Orthogonal Decomposition (POD). This reduction technique allows the most prominent dynamics of a system to be described by a smaller order model than the original. The POD method relies on characteristic spatial dynamics to be described by a relatively small set of basis vectors, reducing the number of equations that needs to be solved. This basis is derived from measured or simulated data, thereby capturing the spatial dynamics specifically for the observed system.
The PDE models are developed using relations from queuing theory. Therefore, specific parameters of these PDE models are only known for manufacturing systems that meet the assumptions imposed by these relations. When these models are used to describe the behavior of real-life manufacturing systems it is therefore not clear how these parameters should be determined. For these systems, system identification (SID) technique's can be used to estimate the unknown model parameters.

Assignment

Investigate the use of the POD reduction method to find reduced order models for manufacturing systems. Use the reduction technique in combination with SID technique's to find estimates for unknown parameters in the PDE models. Of particular interest is the accuracy of the obtained models with respect to the transient behavior of the manufacturing system. When a system is properly identified a control strategy for the observed system can be designed. Present the results of the performed study in a report and provide recommendations for future research.

Prof.dr.ir. J.E. Rooda

Dr.ir. A.A.J. Lefeber

Systems Engineering

Department of Mechanical Engineering

Summary

As a result of the increasing complexity of products and more strict requirements for production, manufacturing systems are becoming increasingly difficult to control. Therefore, more advanced control strategies are needed to control these manufacturing systems. To use well developed continuous control theory for the control of manufacturing systems, continuous approximation models are needed as manufacturing systems are typically of discrete nature. Within the Systems Engineering group at the Department of Mechanical Engineering of the Technische Universiteit Eindhoven, research is done on the use of partial differential equations (PDEs) to describe the dynamical behavior of manufacturing systems.

Solving PDEs typically involves discretization of the spatial domain and solving the governing equations for every single grid cell. In this thesis a reduction method is investigated to reduce the number of function evaluations needed to solve these discretized PDE models. The reduction method is based on the method of Proper Orthogonal Decomposition (POD), which allows for an optimal reduction of the number of modes needed to describe the spatial dynamics of a system. This reduction method has already been successfully applied to computational fluid dynamics (CFD) models for a glass melt oven by the Control Systems group at the department of Electrical Engineering of the Technische Universiteit Eindhoven [Hui05].

Here, the reduction technique is applied to a PDE model that can be used to describe the dynamical behavior of a manufacturing system. Because of some issues regarding the implementation of the derived reduced order model for the manufacturing system, and the complexity of this reduced order model, a different model is derived to describe the behavior of a manufacturing system. This model is of a simple form where the behavior of every single workstation within the manufacturing line is captured in a separate differential equation. The result is a set of ordinary differential equations (ODEs) that make up the model for the complete manufacturing line. When the POD reduction method is applied to the model of the manufacturing system that consists of this set of ODEs, it is shown that the computational effort needed to perform simulations for the model can indeed be reduced, while still obtaining sufficiently accurate results. However, the reduction in computational effort seems to be less efficient for systems with a relative small set of grid cells that makes up the spatial domain of the system.

The model where every single ODE describes the behavior of a single workstation, allows for a control strategy to be derived using a backstepping approach. The resulting control strategy stabilizes the throughput of the continuous approximation model for the manufacturing system onto a predefined trajectory for the demand, compensating for permanent backlog. However, to obtain feasible control actions, the original controller actions need to be saturated. When the throughput of the system at a particular moment in time does not meet the requested throughput at that moment, the performance of the control strategy seemed to vary strongly with the applied saturation level. Another observation for the developed control strategy is that the number of terms that make up the strategy grows explosively when the system consists of more workstations. Therefore it is recommended that further research is done to find better control strategies for the systems observed in this research. Moreover it is interesting to investigate the use of the POD reduction technique in combination with the development of control strategies based on the backstepping approach, to reduce the number of terms that make up the control strategy.

Samenvatting

Doordat producten complexer worden en er aan steeds strengere eisen moet worden voldaan met betrekking tot het produceren, wordt het lastiger om fabricage systemen te regelen. Het is daarom noodzakelijk om geavanceerdere regelstrategieën te zoeken om in de toekomst dergelijke systemen te kunnen blijven regelen. Voor het gebruik van bewezen regeltechniek voor continue systemen, is een continu benaderingsmodel nodig aangezien fabricage systemen typisch discreet van aard zijn. Binnen de sectie Systems Engineering van de opleiding Werktuigbouwkunde aan de Technische Universiteit Eindhoven wordt daarom onderzoek gedaan naar het gebruik van partiële differentiaal vergelijkingen (PDV's) om het gedrag van fabricage systemen te beschrijven.

Het oplossen van PDV's gebeurt typisch door het discretiseren van het ruimtelijke domein waarbij vervolgens de geldende behoud wetten voor elk element dienen te worden opgelost. In dit verslag wordt onderzoek beschreven naar een reductie methode die de hoeveelheid berekeningen kan reduceren die nodig zijn om een discrete oplossing te vinden voor het PDV model. De reductie methode is gebaseerd op de methode van Proper Orthogonal Decomposition (POD), die het mogelijk maakt om de hoeveelheid modes die nodig zijn om de ruimtelijke dynamica van het model te beschrijven optimaal te reduceren. Deze reductie methode is al succesvol toegepast op numerieke stroming dynamica (CFD) modellen voor een glassmeltoven door de sectie Control Systems binnen de opleiding Elektrotechniek aan de Technische Universiteit Eindhoven [Hui05].

In het hier beschreven onderzoek wordt de reductie methode toegepast op een PDV welke gebruikt kan worden om het dynamische gedrag van een fabricage systeem te beschrijven. Vanwege enkele problemen met betrekking tot de implementatie van het afgeleide gereduceerde model en de complexiteit van dit model, is een simpeler model afgeleid om het gedrag van een fabricage systeem te beschrijven. Dit simpelere model bestaat uit een aantal differentiaal vergelijkingen (DV's) welke ieder afzonderlijk het gedrag van één enkel werkstation beschrijft. Het resultaat is een stelsel van differentiaal vergelijkingen dat het gedrag beschrijft van een compleet fabricage systeem. Het blijkt dat wanneer de POD reductie methode wordt toegepast op dit model, de hoeveelheid rekenwerk die nodig is om een simulatie uit te voeren voor het model inderdaad kan worden gereduceerd, terwijl nog steeds voldoende nauwkeurige resultaten worden behaald. De reductie lijkt echter minder sterk te zijn voor systemen waar het ruimtelijke domein uit een relatief kleine set van elementen bestaat.

Voor het op DV's gebaseerde model is een regelstrategie afgeleid waarbij gebruik is gemaakt van een back-stepping methode. De hiermee afgeleide regelstrategie zorgt er voor dat de doorzet van het benaderingsmodel een vooraf gedefinieerde trajectorie voor de vraag volgt, waarbij wordt gecompenseerd voor permanente backlog. Voor fysiek relevante regelsignalen echter, dient het bereik van de regelacties te worden beperkt. Het blijkt dat de prestatie van de regelstrategie afhangt van het gekozen bereik voor de regelsignalen wanneer de doorzet op een bepaald moment niet overeenkomt met de vraag op dat moment. Daarnaast blijkt dat het aantal termen waaruit de regelstrategie bestaat explosief toeneemt voor systemen die uit meerdere werkstations zijn opgebouwd. Het wordt daarom aanbevolen om verder onderzoek te verrichten naar andere regelmethoden voor het regelen van de in dit onderzoek beschreven systemen. Daarnaast is het interessant om verder uit te zoeken hoe de POD reductie methode kan worden gebruikt om de hoeveelheid termen waaruit een regelstrategie bestaat die is afgeleid met een back-stepping methode kan worden teruggedrongen.

Contents

Chapter 1

Introduction

The control of manufacturing systems is becoming increasingly complex as the complexity of products and the requirements for production increase. In the semiconductor industry for example, integrated circuits require more and more complex processing steps as the complexity in chip design increases. In the past, many scheduling algorithms and heuristics have been developed to assist in the control of manufacturing systems. However, as the complexity of manufacturing systems and requirements for production keep increasing, there is a need for more advanced control strategies.

Recent research on the control of manufacturing systems focusses on the use of well developed control theory as available for continuous systems. However, as manufacturing systems are typically of discrete nature, continuous approximation models are needed to develop control strategies based on continuous control theory. When such a strategy is derived it has to be coupled to the actual manufacturing system, where some conversions may have to take place to translate between continuous signals and signals of discrete nature.

To find suitable continuous approximation models, research is done on the use of partial differential equations (PDEs) to describe the behavior of manufacturing systems. These PDEs are of continuous nature and therefore allow the design of control strategies for the control of manufacturing systems.

The dynamical behavior of PDEs used to describe the behavior of manufacturing systems depend, like many other PDE models, on both time and place. Solving PDE's numerically, typically involves the discretization of the spatial domain into grid cells. The discretized governing equations then need to be solved for every single grid cell. This can result in large sets of equations for large scaled, or fine discretized models. Solving PDE based models can therefore be time-consuming. For complex manufacturing systems, and control strategies that require many model evaluations, computation time might therefore become an issue.

Objective

In this research a method is investigated to reduce the computational effort needed to evaluate PDE-based models. The technique used to reduce the number of equations that have to be solved is called the method of Proper Orthogonal Decomposition (POD). This reduction technique allows the most prominent dynamics of the system to be captured in a reduced order model. The method relies on the characteristic spatial dynamics of the model to be described by a small set of basis functions, or modes. The dynamics of the original model are then projected on this reduced set of basis functions using a Galerkin projection, which results in a reduced order model.

A benefit of this type of model reduction, other than the reduction in computational effort needed to solve the PDE-models, is that the spatial and temporal dynamics of the observed system are separated. This allows the reduced order models to be written as a set of ordinary differential equations (ODEs), for which well developed control theory is available. Therefore, in the second part of this research a control strategy is derived based on a model consisting of a set of ODEs, that allows a simple manufacturing line to follow a predefined trajectory for the demand. As manufacturing systems are typically of nonlinear nature, the control strategy is derived using a backstepping approach, a well known method for the control of sets of nonlinear ODEs.

Approach

The theory on model reduction using the method of proper orthogonal decomposition (POD) is discussed in Chapter 2. This theory is illustrated using an example for a basic PDE-model which can be used to describe the behavior of a manufacturing system. The third chapter deals with some issues regarding the implementation of a reduced order model for the manufacturing system. This chapter also discusses an alternative method for numerically solving a PDE model, a first order Godunov method, which results in a set od ODEs that make up a full order model for the manufacturing system. This full order model can be reduced using a method very similar to the POD reduction method, which is discussed in section 3.2.3. In Chapter 4 the performance of the different implementations for the PDE model is discussed. Here the effect of an actual reduction of the number of POD basis functions is shown for the reduced order model based on the Godunov method. Chapter 5 discusses a backstepping approach which is used to derive control strategies for models based on the Godunov method. The performance of these strategies is discussed both in the continuous setting as in a discrete setting where the controlled system is subject to variability. Conclusions and recommendations can be found in Chapters 6 and 7 respectively

Chapter 2

Model reduction

Models based on Partial Differential Equations (PDE), are typically solved numerically where the spatial domain is discretized into grid cells. For large scale PDE models, or fine discretized models, this results in large amounts of grid cells. Numerically solving a PDE can therefore be a computationally intensive task as the governing equations have to be solved for every individual grid cell at every single step in time. Therefore, for an overall reduction of the computational effort, the reduction of the number of grid cells has the greatest impact. A reduction method that uses this principle is the method of Proper Orthogonal Decomposition (POD), which is discussed here. In the first section of this chapter the spectral decomposition that forms the base for this POD reduction technique is discussed. The second section discusses the theory on the POD technique, that allows for the optimal reduction of the number of grid cells. In the third section a method used to derive a model for the reduced set of "grid cells", a Galerkin projection, is discussed. The theory is illustrated using an example of a basic PDE model for a manufacturing system.

2.1 Spectral decomposition

The POD reduction technique is based on spectral decomposition. This expansion allows the overall dynamics of the observed system to be described by a set of time-*independent* basis functions accompanied by a set of corresponding time-*dependent* coefficient functions.

Consider a time set $\mathbb{T} \subseteq \mathbb{R}$ and a spatial domain $\mathbb{X}$. The generalized Fourier series is an expansion that maps $x \in \mathbb{X}$ to $f(x,t) \in \mathbb{R}$ for any $t \in \mathbb{T}$ [Ast04] and is given by:

$$f(x,t) = \sum_{i=1}^{\infty} a_i(t)\varphi_i(x). \tag{2.1}$$

Here $\varphi_i(x)$ denotes the i^{th} basis function and $a_i(t)$ denotes the corresponding coefficient function, also known as a Fourier coefficient. The basis functions are assumed to be orthonormal such that $(\varphi_i(x), \varphi_i(x)) = 1$ and $(\varphi_i(x), \varphi_j(x)) = 0$ if $i \neq j$, where the mapping (a, b) denotes the inner product of a with b. The basis functions describe the spatial dynamics of the observed system.

The Fourier coefficients are defined by:

$$a_i(t) := (f(x,t), \varphi_i(x)). \tag{2.2}$$

It should be noted that though $a_i(t)$ is derived from functions dependent on x, the resulting function is independent from x. The coefficient functions, or Fourier coefficients, describe the temporal dynamics of the system. That is, the Fourier coefficients as defined in (2.2) determine the effect that corresponding basis functions have at a particular moment in time, such that $f(x,t)$ is reconstructed.

The spectral decomposition (2.1) can be reduced by truncating the expansion to the first n modes, which results in the truncated expansion:

$$f_n(x,t) = \sum_{i=1}^{n} a_i(t)\varphi_i(x). \tag{2.3}$$

For the accuracy of the truncated expansion (2.3) it is important that the n most prominent modes are taken into account. This ensures that the time-averaged truncation error between the original expansion and the truncated expansion is minimal.

Let the time-averaged truncation error be defined by:

$$\langle |f(x,t)|^2 \rangle - \langle |f_n(x,t)|^2 \rangle = \left\langle \Big| \sum_{i=n+1}^{\infty} a_i(t)\varphi_i(x) \Big|^2 \right\rangle = \sum_{i=n+1}^{\infty} \langle a_i^2(t) \rangle, \tag{2.4}$$

where $|.|$ denotes the l_2-norm and $\langle . \rangle$ denotes a time averaged variable. In the last step of (2.4) the orthonormality of the basis functions is used to simplify the truncation error.

From (2.4) it can be seen that the time averaged error is minimal when the time averaged coefficient functions of the modes ignored by the truncated expansion are minimal. This optimality can be obtained by making sure that the time averaged Fourier coefficients are ordered such that:

$$\langle a_1^2(t) \rangle \geq \langle a_2^2(t) \rangle \geq \cdots \geq \langle a_n^2(t) \rangle \geq \cdots \geq \langle a_\infty^2(t) \rangle. \tag{2.5}$$

This means that, when (2.5) is met, for *any* truncation level n the "energy" in the first n coefficients is maximal. Therefore the correlation of the first n basis functions with

the original function $f(x,t)$ is maximal. A POD basis is, by definition, a basis that meets this ordering criterium (2.5) and the criterium for the orthonormality of the basis functions.

2.2 POD basis

Suppose a discrete set of measured or simulated data is given, which consists of k time samples for l locations:

$$\boldsymbol{M} = \begin{bmatrix} f(x_1,t_1) & \cdots & f(x_1,t_k) \\ \vdots & \ddots & \vdots \\ f(x_l,t_1) & \cdots & f(x_l,t_k) \end{bmatrix}. \tag{2.6}$$

It should be noted that the maximal number of POD basis functions that can be determined from this discrete set of data, is limited to the amount of locations that are observed. That is, in the truncated expansion (2.3) $n \leq l$, as using more than l basis functions gives no additional information on the spatial dynamics captured in $\boldsymbol{M}$. Introducing more basis functions only results in dependence on other basis functions, which results in a basis that is not orthonormal. The number of spatial coordinates observed in $\boldsymbol{M}$ limit the maximal number of orthogonal basis functions that can be determined for this discrete set of data.

From (2.5) it can be seen that the truncation error can be minimized by ordering the basis functions and corresponding Fourier coefficients such that the magnitude of the Fourier coefficients decreases for every following set of basis functions and coefficients. A POD basis is then found by solving the following optimization problem:

maximize

$$F(\varphi_1,\ldots,\varphi_n) = \sum_{i=1}^{n} \langle (\boldsymbol{M}(k),\varphi_i)^2 \rangle, \tag{2.7a}$$

subject to:

$$|\varphi_i|^2 = 1; \qquad i = 1,2,\ldots,n \tag{2.7b}$$

and

$$(\varphi_i,\varphi_j) = 0; \qquad \forall\ i \neq j, \tag{2.7c}$$

where (2.7a) defines the energy in the first n coefficients, (2.7b) demands the normalization of the basis vectors and (2.7c) demands the basis vectors to be orthogonal. Note that in (2.7a), (2.2) is used to determine the Fourier coefficients.

It can be shown [Ast04, Hui05], that this optimization problem relates to the following eigenvalue problem:

$$C(\varphi_i(x)) = \left\langle \big(\boldsymbol{M}(x,t), \varphi_i\big), \boldsymbol{M}(x,t) \right\rangle = \lambda_i \varphi_i, \tag{2.8}$$

where $C(\varphi_i(x))$ defines the correlation of the basis function $\varphi_i(x)$ with the measurements (2.6). Using (2.8), the POD basis functions can be determined together with the corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_l$. Note that the eigenvalues are ordered and that they are a measure for the "energy" captured by the corresponding basis vectors, similar to (2.5). Furthermore, the eigenvalues need not be distinct. The eigenvectors are independent even for eigenvalues that are the same.

A correlation measure between a reconstructed set of measurements based on the first n basis vectors and the original set of measurements can be defined by:

$$P_n = \frac{\sum_{i=1}^{n} \lambda_i}{\sum_{i=1}^{l} \lambda_i}; \qquad n = 1, \ldots, l. \tag{2.9}$$

This correlation level indicates how good a reconstructed data set f_n (2.3) approximates the original set f_l.

The method used to determine a set of POD basis functions and reconstructing the measured data using these basis functions is now illustrated using a simple example for a manufacturing system:

Example 2.1: ▷ Consider a manufacturing system that consists of a production line with 100 identical workstations. Each workstation consists of a infinite capacity buffer and a machine with exponentially distributed process times with mean μ. Suppose that the density distribution within the manufacturing system can be described by the following PDE model [Ber04]:

$$\rho_t(x,t) + \frac{\mu m}{(m + \rho(x,t))^2} \rho_x(x,t) = 0, \tag{2.10a}$$

with initial condition:

$$\rho(x,0) = f(x), \tag{2.10b}$$

and boundary condition:

$$\rho(0,t) = \frac{m\lambda(t)}{\mu - \lambda(t)}. \tag{2.10c}$$

In this model ρ denotes the density [lots/unit of place], μ denotes the mean processing rate of a machine [lots/hour], m denotes the number of machines and $\lambda(t)$ denotes the mean rate [lots/hour] at which lots are released into the system at time t. The variables ρ_t and ρ_x denote the derivative of ρ with respect to time and place respectively. The lots enter the model at $x = 0$, and leave the system at $x = 1$. The release rate of lots into the manufacturing line is also exponentially distributed.

A case is considered where the manufacturing system is initially in steady state, and is subsequently ramped up to a new state that corresponds to a higher utilization level. This ramp up is started by applying a new influx that corresponds to the newly desired state. The initial conditions and boundary conditions can then be written as:

$$\rho(x,0) = \frac{m\lambda_1}{\mu - \lambda_1} \qquad x > 0, \tag{2.11}$$

and:

$$\rho(0,t) = \frac{m\lambda_2}{\mu - \lambda_2} \qquad t \geq 0, \tag{2.12}$$

respectively, where λ_1 is the release rate that corresponds to the initial steady state, and λ_2 corresponds to the target state for the manufacturing line.

Using these conditions, the PDE model (2.10a) can be solved analytically:

$$\rho(x,t) = \begin{cases} \frac{m\lambda_1}{\mu-\lambda_1} & \text{for} \quad 0 < t \leq \frac{m\mu x}{(\mu-\lambda_1)^2} \\ \sqrt{\frac{\mu m t}{x}} - m & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_1)^2} \leq t \leq \frac{m\mu x}{(\mu-\lambda_2)^2} \\ \frac{m\lambda_1}{\mu-\lambda_2} & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_2)^2} \leq t, \end{cases} \tag{2.13a}$$

with the following derivatives with respect to place:

$$\rho_x(x,t) = \begin{cases} 0 & \text{for} \quad 0 < t \leq \frac{m\mu x}{(\mu-\lambda_1)^2} \\ -\frac{1}{2x}\sqrt{\frac{\mu m t}{x}} & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_1)^2} \leq t \leq \frac{m\mu x}{(\mu-\lambda_2)^2} \\ 0 & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_2)^2} \leq t, \end{cases} \tag{2.13b}$$

and time:

$$\rho_t(x,t) = \begin{cases} 0 & \text{for} \quad 0 < t \leq \frac{m\mu x}{(\mu-\lambda_1)^2} \\ \frac{1}{2t}\sqrt{\frac{\mu m t}{x}} & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_1)^2} \leq t \leq \frac{m\mu x}{(\mu-\lambda_2)^2} \\ 0 & \text{for} \quad \frac{m\mu x}{(\mu-\lambda_2)^2} \leq t. \end{cases} \tag{2.13c}$$

It can be seen that (2.13a) is indeed a solution to (2.10a) by substituting (2.13b) and (2.13c) into (2.10a).

A discretization of the solution is shown in Figure 2.1, where $\lambda_1 = 0.5$, $\lambda_2 = 1.0$ and the mean processing rate of the machines $\mu = 2.0$ [lots/hour]. The spatial domain $\mathbb{X}$ is uniformly discretized into 100 discrete points, so every discrete point in $\mathbb{X}$ corresponds to a single workstation. The temporal domain $\mathbb{T}$ is also discretized into 100 points. Since the system is originally in steady state, the initial flowrate λ_1 is achieved throughout the complete manufacturing line, which corresponds to an utilization level of 25%. The overall density that corresponds to this flux can be calculated using (2.11) and results in a density of $33\frac{1}{3}$ [lots/place]. At time $t = 0$ the release rate of lots into the manufacturing system is raised to 1.0 [lots/hour], the system is ramped up to a 50% utilization level.

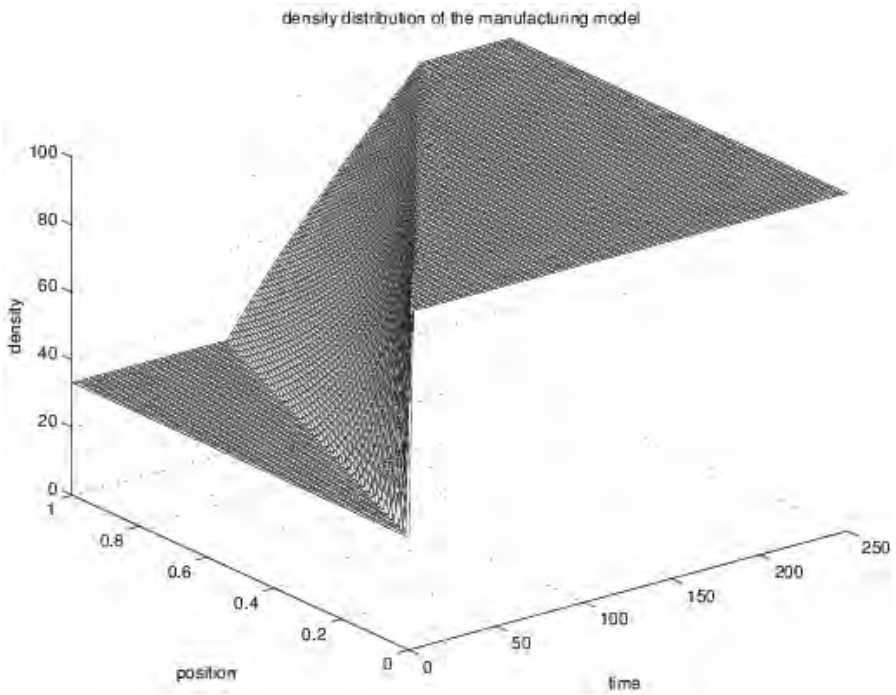


Figure 2.1: Discretized solution for the manufacturing model case, used to calculate the POD basis functions

Figure 2.1 clearly shows that the density at the workstations further down the production line slowly increases as lots move further down the production line. Eventually the

system reaches the steady state were the overall density is 100, which corresponds to the target utilization of the system.

To obtain the POD basis functions and the corresponding eigenvalues, the eigenvalue decomposition is performed for $C(\varphi_i(x))$ (2.8), where $\boldsymbol{M}(x,t)$ contains the discretized solution to the observed case as shown in Figure 2.1, and the inner product is calculated using a matrix multiplication. The resulting eigenvalues are shown in Figure 2.2. It should be noted that the maximum number of eigenvalues and corresponding eigenvectors in this case is 100 as only data is available for 100 locations in $\mathbb{X}$.

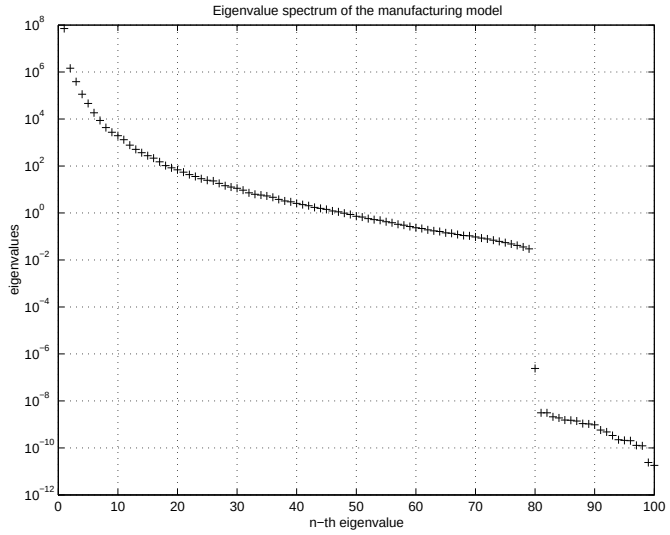


Figure 2.2: Eigenvalue spectrum of the manufacturing system data

The figure shows a sharp descent for the 80-th eigenvalue, indicating that the following eigenvectors do not substantially contribute to the reconstruction of the original data. However, it is possible to perform a stronger reduction while still obtaining good results.

The error of the reconstructed data with the original data is calculated for different orders of reduction and shown in Figure 2.3. It can be seen in this figure that for more than 12 POD basis functions, the maximal absolute error does not improve much. Therefore the original data is approximated using only the first twelve basis functions.

These first twelve POD basis functions are shown in Figure 2.4. The figure shows that the first POD basis function describes the general spatial characteristics of the data shown in Figure 2.1. The following basis functions subsequently add more detail, especially to the region near the entrance of the manufacturing line ($x = 0$). This can be explained by the fact that the changes in density in the region close to $x = 0$ are strongest.

The result of a reconstruction based on these first twelve POD basis functions is shown in Figure 2.5. The figure shows that the reduced set of basis functions captures the

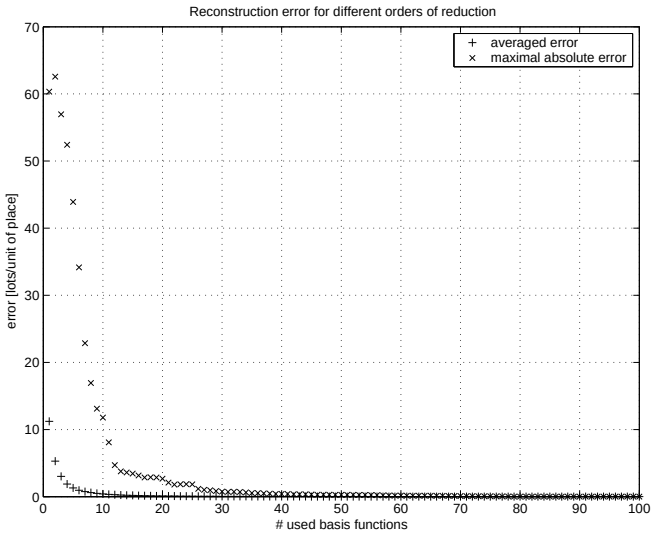


Figure 2.3: Errors for different reduction orders

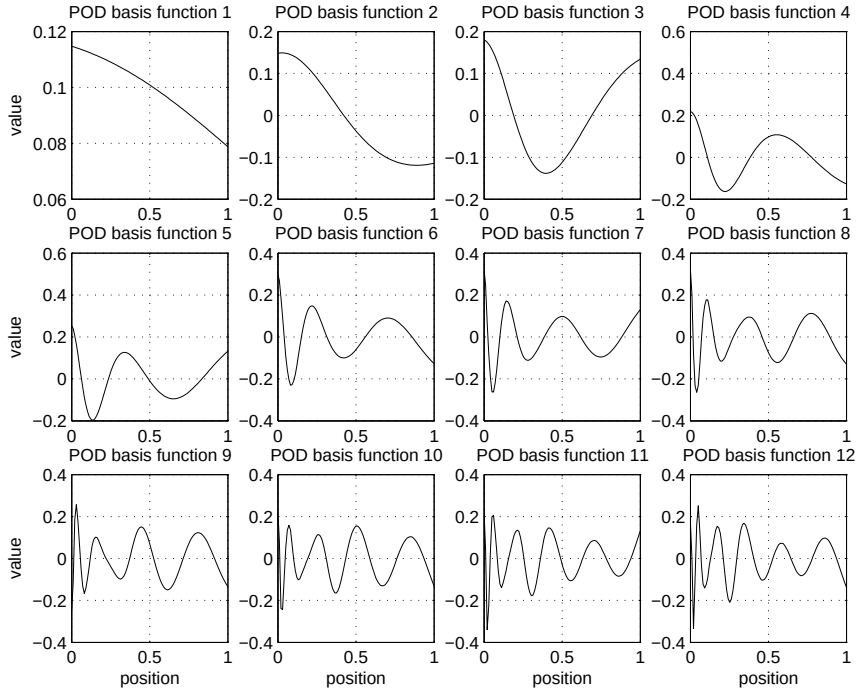


Figure 2.4: The first 12 POD basis functions used to approximate the data of the manufacturing model.

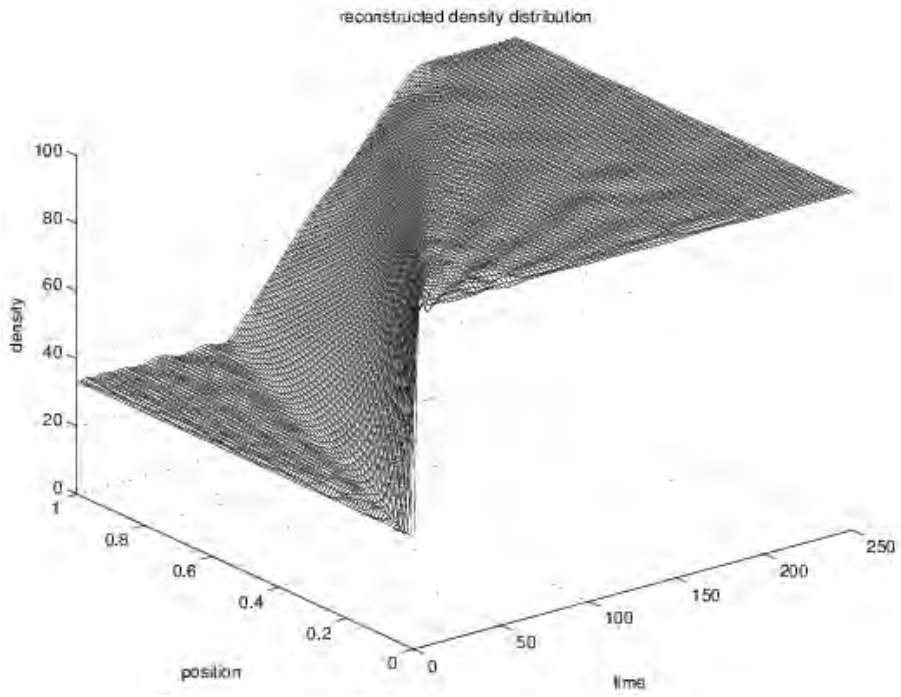


Figure 2.5: Reconstructed data of the manufacturing model, based on the first 12 POD basis functions

main characteristics of the original data (Figure 2.1) quite well.

The difference between the reconstructed and original data is shown in Figure 2.6, it can be seen here that the largest error occurs at the discontinuity in the initial condition. This can be explained by the fact that the POD basis functions describe the time averaged spatial characteristics of the original data, while the discontinuity only occurs for time $t = 0$. The figure further shows some errors along the edges $x = \frac{(\mu-\lambda_1)^2}{m\mu}t$ and $x = \frac{(\mu-\lambda_2)^2}{m\mu}t$ since these sharp edges are not exclusive for a specific location and thus difficult to capture in a reduced set of basis functions.

$\triangleleft$

This section showed that the POD method can be used to approximate a data set by using a reduced set of basis vectors and corresponding Fourier coefficients. When the Fourier coefficients are determined from the original data set using (2.2) however, only the original data set can be reconstructed. In order to obtain a reduced order model, that allows other behavior than the behavior captured in the data set to be simulated, a PDE model can be projected on the reduced set of basis functions. This projection, a Galerkin projection, is discussed in the next section.

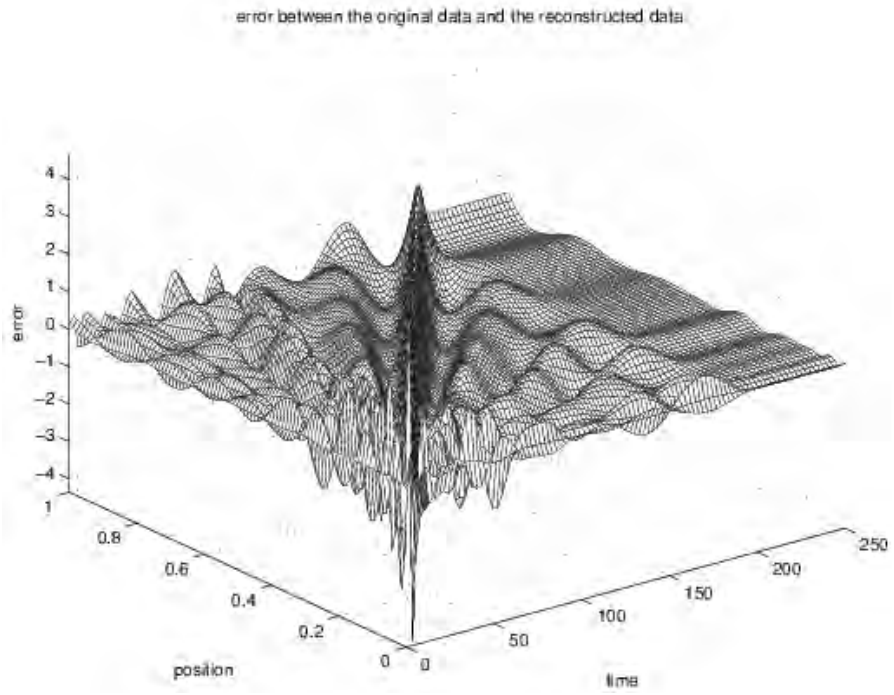


Figure 2.6: Error of the reconstructed data based on the first 12 basis functions compared to the original data used to determine the POD basis functions

2.3 Galerkin projection

In the previous section the procedure for obtaining a reduced order POD basis from measured data was discussed. With the obtained POD basis functions and the corresponding Fourier coefficients (2.2), the original data could be reconstructed. In this section a Galerkin projection method is discussed. This projection method allows a reduced order model to be derived for the original PDE model, which allows dynamical behavior to be simulated different from the behavior captured in the data set used to determine the POD basis functions.

Consider a PDE of the form:

$$M(f(x,t)) = D(f(x,t)), \tag{2.14}$$

where $M(f(x,t))$ contains only temporal derivatives in a polynomial form and $D(f(x,t))$ involves only spatial derivatives. Now a residual function is defined as:

$$R(f(x,t)) = M(f(x,t)) - D(f(x,t)). \tag{2.15}$$

Example 2.2: ▷ For the PDE model (2.10) used in the previous example this results in:

$$M(\rho(x,t)) = \rho_t(x,t), \tag{2.16}$$

and

$$D(\rho(x,t)) = -\frac{\mu m}{(m+\rho(x,t))^2}\rho_x(x,t). \tag{2.17}$$

The residual for the model is then given by:

$$R(\rho(x,t)) = \rho_t(x,t) + \frac{\mu m}{(m+\rho(x,t))^2}\rho_x(x,t). \tag{2.18}$$

◁

Using the truncated spectral expansion (2.3), the residual function (2.15) can be formulated as:

$$R(f_n(x,t)) = M(f_n(x,t)) - D(f_n(x,t)). \tag{2.19}$$

Since in general $f(x,t) \neq f_n(x,t)$, it could be that $R(f_n(x,t)) \neq 0$. This would mean that the dynamics of the reduced order model do not match the dynamics of the original model, the reduced order model behaves differently. It is however preferred that the reduced order model follows the dynamical behavior of the original model as close as possible. A constraint is needed on the projection method, to allow the reduced order model to match the dynamical behavior of the original model as close as possible. Therefore, the Galerkin projection is defined as [Ast04]:

$$(\varphi_j(x), R(f_n(x,t))) = 0; \qquad j = 1,\ldots,n. \tag{2.20}$$

This definition states that the residual of the reduced order model dynamics is not correlated to the first n basis functions at all. That is, the relation between the temporal dynamics and the spatial dynamics for the original model still hold in the space spanned by the first n basis functions. There is no "leakage" of dynamics and no extra dynamical behavior is added to the reduced order model. It can be seen that when the residual of the reduced order model dynamics is reconstructed, the approximation of this residual in the space spanned by the first n basis functions is still zero:

$$\tilde{R}(f_n(x,t)) = \sum_{j=1}^{n}(\varphi_j(x), R(f_n(x,t)))\varphi_j = 0 \qquad j = 1,\ldots,n. \tag{2.21}$$

Example 2.3: ▷ When the residual error defined in the previous example (2.18) is projected onto the first n basis functions, the residual error can be written as:

$$\begin{aligned}(\varphi_j(x), R(f_n(x,t))) &= \left(\varphi_j(x), \rho(x,t)_t + \frac{\mu m}{(m+\rho(x,t))^2}\rho_x(x,t)\right) \qquad j = 1,\ldots,n \\ &= \left(\varphi_j(x), \sum_{i=1}^{n} \dot{a}_i(t)\varphi_i(x) + \frac{\mu m}{(m+\sum_{p=1}^{n} a_p(t)\varphi_p(x))^2}\sum_{i=1}^{n} a_i(t)\varphi_i'(x)\right) \\ &= \dot{a}_j(t) + \left(\varphi_j(x), \frac{\mu m}{(m+\sum_{p=1}^{n} a_p(t)\varphi_p(x))^2}\sum_{i=1}^{n} a_i(t)\varphi_i'(x)\right),\end{aligned} \tag{2.22}$$

where $\varphi_i'(x)$ denotes the derivative of $\varphi_i(x)$ with respect to place (x) and the orthonormality of the basis functions reduce the first term on the right hand side to $\dot{a}_j(t)$. Using the definition for the Galerkin projection (2.20), the reduced order model can be written as:

$$\dot{a}_j(t) = -\left(\varphi_j(x), \frac{\mu m}{(m+\sum_{p=1}^{n} a_p(t)\varphi_p(x))^2}\sum_{i=1}^{n} a_i(t)\varphi_i'(x)\right) \qquad j = 1,\ldots,n. \tag{2.23}$$

◁

The previous example showed that using a POD basis and Galerkin projection as defined in (2.7) and (2.20) respectively, it is possible to determine a reduced order model for a PDE of the form (2.14). Since the reduced order model consists of a set of ODEs it can be solved using well developed techniques for solving ODEs. This set of ODEs however, does not yet incorporate the boundary conditions of the original PDE model.

When it is assumed that the function $f(x,t)$ is a square integrable function, the inner product can be defined as [Ast04]:

$$(a,b) := \int_0^1 a(x,t)b(x,t)dx. \tag{2.24}$$

This definition allows the boundary conditions to be included in the reduced order model by using integration by parts, which is illustrated in the following example.

Example 2.4: ▷ The boundary conditions for the reduced order model for the manufacturing system, discussed in the previous examples, can be included by using integration by parts. These boundary conditions are especially important for a manufacturing model as the boundary condition at $x = 0$ allows the flux of lots into the system to be

specified explicitly. This is a prerequisite for the control of the manufacturing system as it allows the influx to be altered in time. Using definition (2.24), (2.23) can be written as:

$$\dot{a}_j(t) = -\sum_{i=1}^{n} \int_0^1 \varphi_j(x) \frac{\mu m}{(m + \sum_{p=1}^{n} a_p(t)\varphi_p(x))^2} a_i(t)\varphi_i'(x)dx \qquad j = 1, \ldots, n. \quad (2.25)$$

The integral in this equation can be integrated by parts, allowing the boundary conditions to be explicitly defined:

$$\int_a^b f dg = fg|_a^b - \int_a^b g df. \quad (2.26)$$

For simplicity (2.25) is first rewritten into:

$$\dot{a}_j(t) = -\int_0^1 \varphi_j(x)\big(\hat{v}_\rho(\hat{\rho})\hat{\rho} + \hat{v}(\hat{\rho})\big)\hat{\rho}_x dx \qquad j = 1, \ldots, n, \quad (2.27)$$

with the reconstructed variables:

$$\begin{aligned} \hat{\rho}(x,t) &= \sum_{i=1}^{n} a_i(t)\varphi_i(x) & v(\hat{\rho}) &= \frac{\mu}{m + \hat{\rho}(x,t)} \\ \hat{\rho}_x(x,t) &= \sum_{i=1}^{n} a_i(t)\varphi_i'(x) & v_\rho(\hat{\rho}) &= \frac{-\mu}{\big(m + \hat{\rho}(x,t)\big)^2} \\ \hat{\rho}_t(x,t) &= \sum_{i=1}^{n} \dot{a}_i(t)\varphi_i(x) & v_{\rho\rho}(\hat{\rho}) &= \frac{2\mu}{\big(m + \hat{\rho}(x,t)\big)^3}. \end{aligned} \quad (2.28)$$

Using the following choices for f and dg in (2.26):

$$\begin{aligned} f &= \varphi_j(x)\big(\hat{v}_\rho(\hat{\rho})\hat{\rho} + \hat{v}(\hat{\rho})\big) & dg &= -\hat{\rho}_x dx \\ df &= \big(\hat{v}_\rho(\hat{\rho})\hat{\rho} + \hat{v}(\hat{\rho})\big)\varphi_j'(x)dx + \big(\hat{v}_{\rho\rho}(\hat{\rho})\hat{\rho}\hat{\rho}_x + 2\hat{v}_\rho(\hat{\rho})\hat{\rho}_x\big)\varphi_j(x)dx & g &= -\hat{\rho}, \end{aligned}$$

the reduced order model becomes:

$$\begin{aligned} \dot{a}_j(t) = &-\big(\hat{v}_\rho(\hat{\rho})\hat{\rho} + \hat{v}(\hat{\rho})\big)\varphi_j(x)\hat{\rho}\Big|_{x=0}^{x=1} \\ &+ \int_0^1 \Big(\big(\hat{v}_\rho(\hat{\rho})\hat{\rho} + \hat{v}(\hat{\rho})\big)\varphi_j'(x) + \big(\hat{v}_{\rho\rho}(\hat{\rho})\hat{\rho}\hat{\rho}_x + 2\hat{v}_\rho(\hat{\rho})\hat{\rho}_x\big)\varphi_j(x)\Big)\,\hat{\rho}dx. \end{aligned} \quad (2.29)$$

The first part on the right hand side of (2.29) describes the flux of lots into the manufacturing system (release rate) and the flux of lots out of the system (throughput) at respectively $x = 0$ and $x = 1$. This means that the release rate of lots into the manufacturing system can now be explicitly defined. The second part on the right hand side of (2.29) describes the internal dynamics of the system. Solution of the set of differential equations (2.29) yields the time dependent functions $a_j(t)$. Together with the basis functions $\varphi_j(x)$, the density distribution can be reconstructed in $\mathbb{X}$, using (2.3). $\lhd$

In this chapter the theory on model reduction using the method of Proper Orthogonal Decomposition and a Galerkin projection was discussed. The theory was illustrated using examples for a simple PDE model of a manufacturing line. This resulted in a reduced order model for the manufacturing line, where the influx of lots into the system can be specified using the boundary condition at $x = 0$.

To discuss the resulting reduced order model of the manufacturing system, the model should first be implemented in a technical computing tool like, for example, Matlab. This implementation however, proved to be problematic. The issues related to this implementation are discussed in the following chapter.

Chapter 3

Implementation issues

In the previous chapter a reduced order model was derived for a manufacturing line consisting of identical workstations. Implementation of this model however, revealed some problems. Therefore, in the first section of this chapter some issues related to the implementation of the reduced order model (2.29) are discussed. The second section of this chapter discusses the derivation of a different reduced order model for the PDE model of the manufacturing system, which is of a simpler form than the reduced order model derived in the previous chapter.

The implementations for the reduced order model in this chapter are simulated for a test case where the manufacturing line is initially in a steady state that corresponds to an utilization level of 25%. The system is subsequently ramped up to a state corresponding to a 75% utilization level. It should be noted that for the simulation results shown in this chapter no actual reduction is performed. That is, for these simulation results all the 100 basis functions are taken into account.

Figure 3.1 shows the results for the test case based on the analytical solution of the PDE model for the manufacturing line (2.13a). This result is used to verify the different implementations for the reduced order model, discussed in the following sections.

3.1 Implementation of a reduced order model based on the Galerkin projection

In this section some issues related to the implementation of the reduced order model for the manufacturing system, as derived in the previous chapter, are discussed. The issues are related to the numerical methods used for calculating integrals and derivatives.

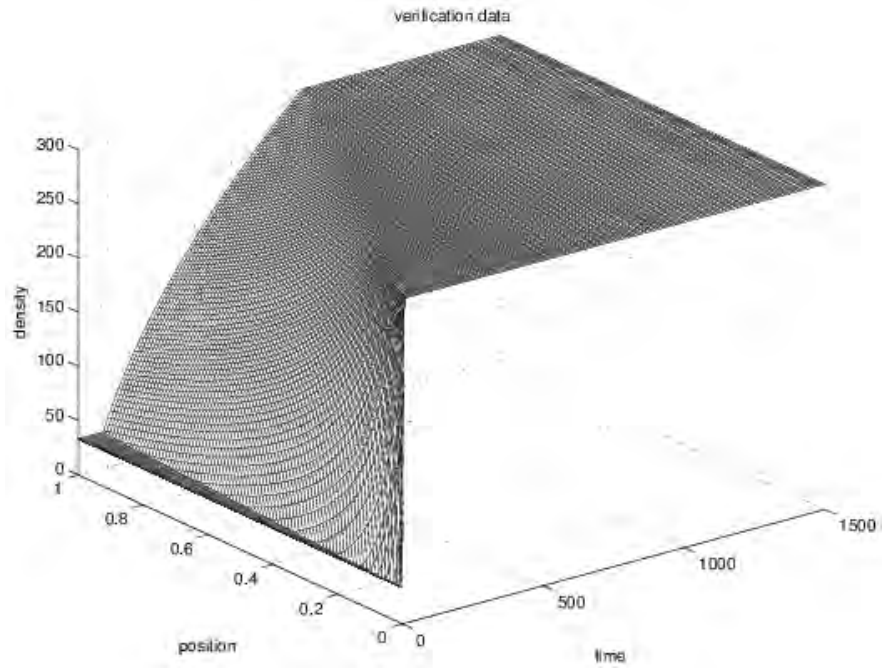


Figure 3.1: Simulation data for the test case, based on the analytical solution for the PDE model

3.1.1 Methods for numerical integration

The basis functions, used for the reduced order model, are determined for a spatial domain $\mathbb{X}$ which is discretized into 100 points (Example 2.1). This means that the integral in (2.29) should be calculated using a numerical integration method. For the first implementation a method known as the trapezoidal method is used for calculating the integral on the right hand side of (2.29). This integration method calculates the integral for a segment between two adjacent discrete points using a first order polynomial for the approximation of the function that is to be integrated. The trapezoidal integration method is discussed in more detail in Appendix C.2.

Figure 3.2 shows the results for the simulation of the test case, where the trapezoidal method is used to calculate the integral on the right hand side of (2.29). When these results are compared to the discretization of the analytical solution for the test case, shown in Figure 3.1, it can be seen that the results do not match the expected behavior. The figure with the results for the implementation based on the trapezoidal integration shows that at both the boundaries $x = 0$ and $x = 1$ the density increases, while for intermediate locations nothing happens. It is however expected that the density first increases at the boundary $x = 0$ and that this increase in density subsequently propagates through the manufacturing line to $x = 1$. It should also be noted that because of the problems at the boundaries the simulation ended prematurely at time

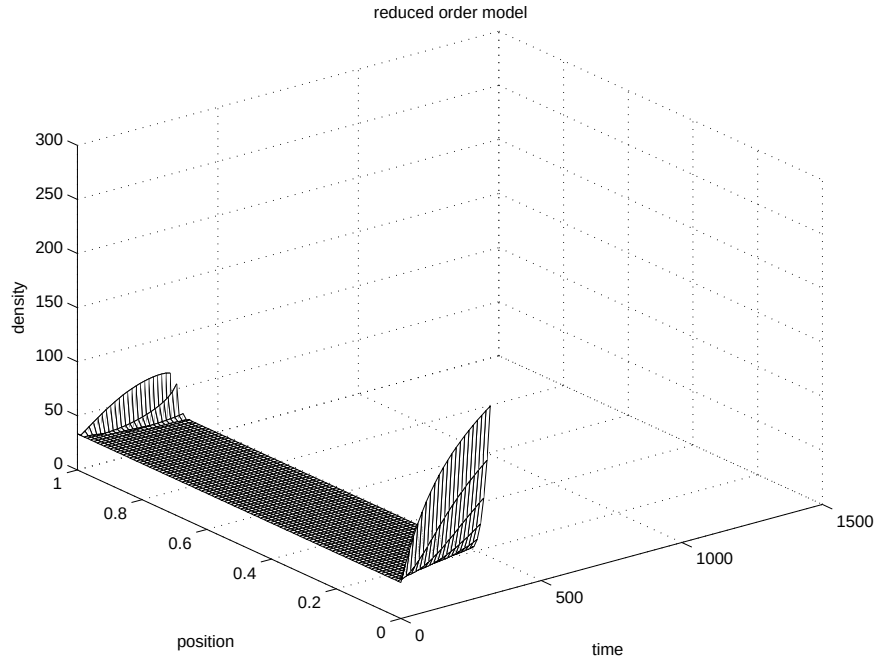


Figure 3.2: Simulation results for implementation using trapezoidal integration

$t \approx 320$.

It is thought that the problems at the boundaries, shown in Figure 3.2, could be caused by the method used for numerical integration. Therefore a more accurate method for numerical integration, a Romberg integration method based on the trapezoidal method for integration, is used for the implementation of the reduced order model for the manufacturing system (2.29). This Romberg method uses information on the error for trapezoidal integration to cancel out the highest order of the error. The highest order of the error is canceled out by combining information on the errors of trapezoidal integrations, performed for different step sizes, using specific weighting factors. These combinations for different step sizes are performed several times to allow for the successive canceling of higher order errors. More information on the Romberg integration method can be found in Appendix C.3. It is expected that, using this method for numerical integration, the problems at the boundaries can be solved.

Figure 3.3 shows the results for the simulation where the integral is calculated using the Romberg method. It can be seen in this figure however, that the results do not improve for this more refined numerical integration method. In fact, the problem becomes worse. The errors at the boundaries now immediately spread across the complete spatial domain. This suggests that a more refined method for numerical integration has a negative effect on the accuracy of the implementation. So the problem at the boundaries might be solved using a less sophisticated method for numerical integration. There-

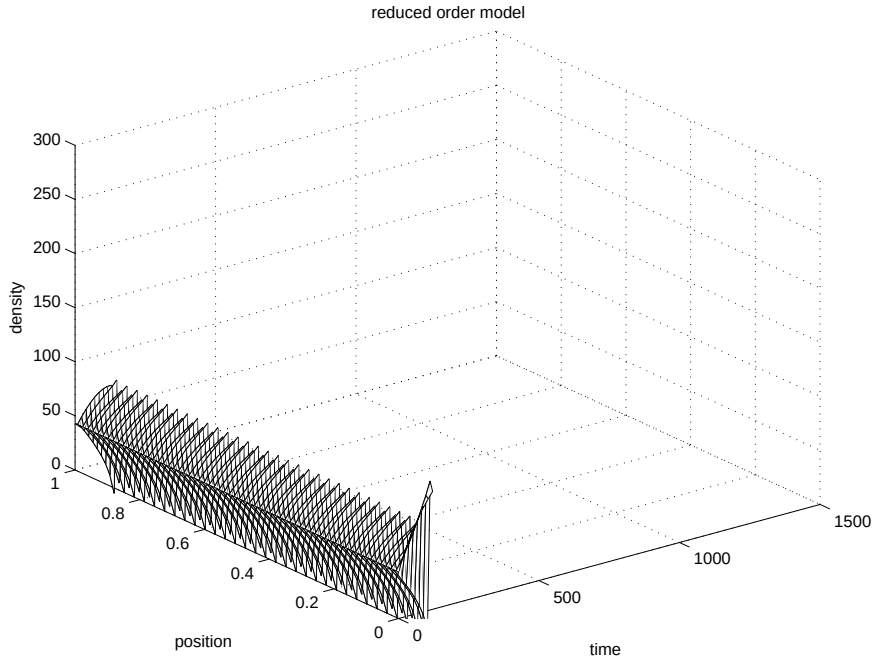
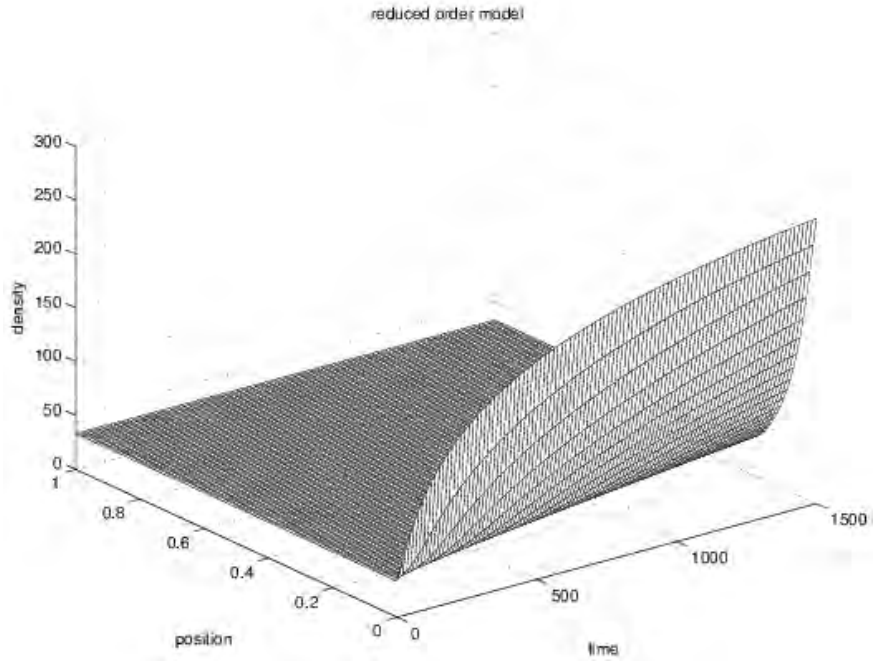


Figure 3.3: Simulation results for implementation using Romberg integration

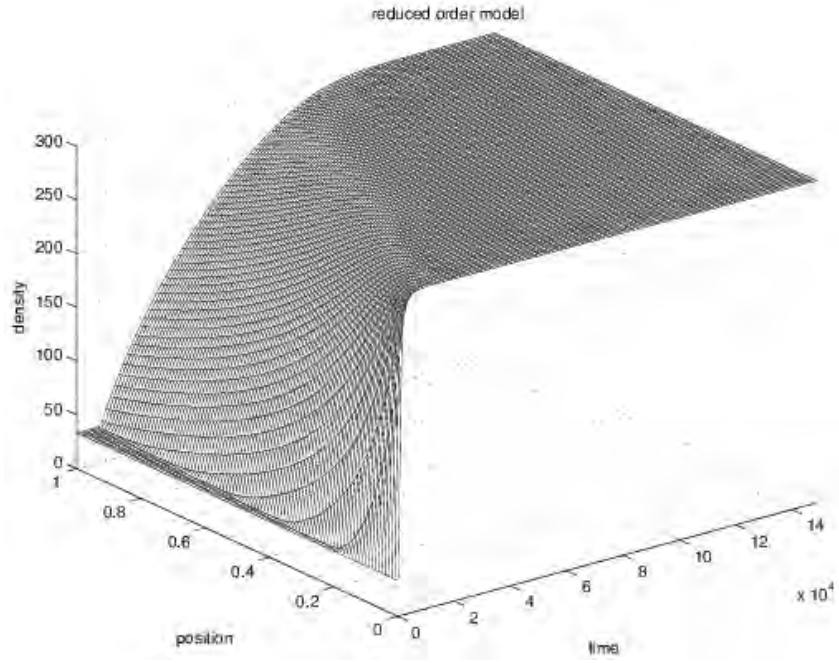
fore a simulation is run using a simpler numerical integration technique. The applied integration method is actually a sum of the function values at the discrete points in $\mathbb{X}$, scaled using the distance between adjacent discrete points and is discussed in more detail in Appendix C.1.

Figure 3.4a shows the simulation results when this simple numerical integration method is used for the implementation of the reduced order model. It can be seen in this figure that, using a simple summation integration technique, the problems at the boundaries are indeed solved.

A more detailed study showed that the problem regarding the boundaries is caused by a mismatch between the boundary conditions, the first term on the right hand side of (2.29), and the internal dynamics, the second term on the right hand side of (2.29). The boundary conditions explicitly define part of the dynamics at $x = 0$ and $x = 1$. For the method of integration by parts (2.26) to hold, these boundary conditions should be complemented by the internal dynamics, given by the integral on the right hand side of (2.29). For the trapezoidal integration method however, the function values at the discrete points $x = 0$ and $x = 1$ are weighed half as much as the values at the other discrete points (Appendix C.2). This means that the internal dynamics at these points is not fully taken into account in the integral, causing the mismatch with the boundary conditions. The boundary conditions at $x = 0$ and $x = 1$ are not fully complemented by the internal dynamics, resulting in the errors at the boundaries as shown in Figure 3.2.



(a) normal time scale



(b) full dynamics

Figure 3.4: Simulation results for implementation using integration by summation

This non uniform weighting also explains the difference between the trapezoidal integration method and the Romberg method for numerical integration. For a trapezoidal integration method, only the values at the boundaries are weighed differently from the rest. The Romberg method however, combines trapezoidal integrations using several different step sizes, which effectively spreads the mismatch across the complete spatial domain $\mathbb{X}$ for a single step in time, as shown in Figure 3.3.

It is important to notice that for the full dynamical response of the implementation using a simple summation method for integration, shown in Figure 3.4b, the time scale does not match the expected time scale (Figure 3.1). This scaling problem occurs since the inner product for the temporal derivatives, the first term on the right hand side of (2.22), needs to be calculated using the same method as used to calculate the inner product for the spatial dynamics, the second term on the right hand side of (2.22). For the previously shown implementation results however, the inner product for the temporal derivatives is calculated using a matrix multiplication, while for the spatial dynamics a numerical integration method is used. As the integration methods take the distance between discrete points into account for the calculation of the inner product, and a matrix multiplication does not, this causes the dynamics to be scaled in time.

The reason that the inner product for the temporal derivatives is calculated using a matrix multiplication is that when this inner product is calculated using a numerical integration technique based on the trapezoidal method, the orthonormality of the basis functions with the basis functions in the reconstructed temporal derivative $\hat{\rho}_t$ does not hold. This means that the inner product of the basis functions for the temporal derivatives in (2.22) does not reduce to $\dot{a}_j(t)$. Therefore in the following section it is investigated if a method can be derived for determining POD basis functions that are orthonormal for a specific method of numerical integration, like for example a trapezoidal integration method.

3.1.2 Inner product in integral form

In this section a method is derived for determining POD basis functions that are orthonormal for a specific method of numerical integration. It is thought that the cause for the lack of orthonormality lies in the different methods used for calculating the inner product in the calculation of the POD eigenfunctions (Section 2.2) on the one hand, and the calculation of the derivative of the Fourier coefficients (Section 2.3) on the other. For the calculation of $C(\varphi_i(x))$ in (2.8) a matrix multiplication is used, while for the application of the method of integration by parts, the inner product used to determine the derivative of the Fourier coefficients is calculated using a numerical integration technique.

The idea that the problem could be these different implementations for the calculation of the inner product, is verified by calculating $C(\varphi_i(x))$ in (2.8) using the same integration technique as used to calculate the integral on the right hand side of (2.29). For

the eigenvalue problem (2.8) to return orthogonal and real eigenvectors however, some restrictions apply to the integration method used. This is shown below.

The eigenvalue problem is given by [Ast04]:

$$C(\varphi_i(x)) = \left\langle \left(\boldsymbol{M}(x,t), \vec{\varphi}_i\right), \boldsymbol{M}(x,t) \right\rangle = \int_{t=0}^{T} \int_{x=0}^{1} \boldsymbol{M}(x,t)\vec{\varphi}_i dx\, \vec{w}_t \boldsymbol{M}(x,t) dt = \lambda_i \vec{\varphi}_i, \tag{3.1}$$

where (2.24) is used to obtain the integrals on the right hand side, the eigenvector $\vec{\varphi}_i$ is the i-th basis function, and $\vec{w}_t$ is a weighting function for averaging over time. To take the effect of the method used for numerical integration into account, the following implementation for the integrals is suggested:

$$\int_a^b f(s)ds = \frac{b-a}{n}\, \vec{c}^{\top} \cdot \vec{f}. \tag{3.2}$$

Here the vector $\vec{c}$ depends on the numerical method used to compute the integral for a vector, and is actually a weighting function. The vector $\vec{f}$ contains the data that is to be integrated.

For the averaging over time $\langle . \rangle$ and inner product $(\cdot, \cdot)$ in (3.1), this results in:

$$\int_{t=0}^{T} f(t)w(t)dt = \frac{T}{n_t}\, \vec{c}_{n_t}^{\top} \cdot \mathrm{diag}(\vec{w}_t) \cdot \vec{f}, \tag{3.3}$$

and

$$\int_{x=0}^{1} f(x)g(x)dx = \frac{1}{n_x}\, \vec{f}^{\top} \cdot \mathrm{diag}(\vec{c}_{n_x}) \cdot \vec{g}, \tag{3.4}$$

where n_x denotes the number of discrete points in $\mathbb{X}$, and n_t the number of discrete points in time. When (3.1) is rewritten using (3.3) and (3.4) this results in:

$$\begin{aligned} \boldsymbol{C}(\varphi_i(x)) &= \frac{T}{n_x n_t} \boldsymbol{M}(x,t) \cdot \mathrm{diag}(\vec{c}_{n_t}) \cdot \mathrm{diag}(\vec{w}_t) \cdot \boldsymbol{M}(x,t)^{\top} \cdot \mathrm{diag}(\vec{c}_{n_x}) \cdot \vec{\varphi}_i \\ &= \boldsymbol{A} \cdot \vec{\varphi}_i. \end{aligned} \tag{3.5}$$

The basis functions can now be determined from the eigenvalue problem:

$$\boldsymbol{A} \cdot \vec{\varphi}_i = \lambda_i \vec{\varphi}_i. \tag{3.6}$$

However, for the eigenvectors to be orthogonal and real, the matrix $\boldsymbol{A}$ should be symmetric.

$$\boldsymbol{A} = \underbrace{\frac{T}{n_x n_t} \boldsymbol{M}(x,t) \cdot \operatorname{diag}(\vec{c}_{n_t}) \cdot \operatorname{diag}(\vec{w}_t) \cdot \boldsymbol{M}(x,t)^\top}_{\text{symmetric}} \cdot \operatorname{diag}(\vec{c}_{n_x}) \tag{3.7}$$

The symmetric part in $\boldsymbol{A}$ suggests that the method used for averaging over time and the weighting function $\vec{w}_t$ in (3.5) have no influence on the symmetry of $\boldsymbol{A}$. It can also be seen that $\boldsymbol{A}$ is completely symmetric when $\operatorname{diag}(\vec{c}_{n_x})$ commutes with the symmetric part in (3.7). In general therefore, the matrix $\boldsymbol{A}$ is symmetric when the elements in the vector $\vec{c}_{n_x}$ have the same value. This is illustrated using a simple example.

Example 3.1: ▷ Consider a diagonal matrix and a symmetric matrix that commute:

$$\begin{aligned}
\begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \cdot \begin{bmatrix} d & e & f \\ e & g & h \\ f & h & i \end{bmatrix} &= \begin{bmatrix} d & e & f \\ e & g & h \\ f & h & i \end{bmatrix} \cdot \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \\
\begin{bmatrix} ad & ae & af \\ be & bg & bh \\ cf & ch & ci \end{bmatrix} &= \begin{bmatrix} da & eb & fc \\ ea & gb & hc \\ fa & hb & ic \end{bmatrix} \\
ae &= be \\
af &= cf \\
bh &= ch,
\end{aligned} \tag{3.8}$$

so when $e \neq 0, f \neq 0$ and $h \neq 0$, $a = b = c$ is needed. ◁

Since individual elements in the symmetric part of (3.7) are in general not zero, the following implementation is chosen for vector $\vec{c}_{n_x}$:

$$\vec{c}_{n_x} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^\top, \tag{3.9}$$

which corresponds to the following implementations for integration of a single vector and multiple vectors:

$$\int_a^b f(x)dx = \frac{b-a}{n_x} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \cdot \vec{f}_{n_x} \tag{3.10a}$$

$$\int_a^b f(x)g(x)h(x)dx = \frac{b-a}{n_x} \vec{f}_{n_x}^\top \cdot \operatorname{diag}(\vec{g}_{n_x}) \cdot \vec{h}_{n_x}. \tag{3.10b}$$

It should be noted that this method for numerical integration is exactly the simple summation integration method, as described in Appendix C.1. However, the eigenvectors derived from (3.6) using Matlab are still orthonormal for inner products calculated using vector multiplications. This means that when the inner product in (3.1) is calculated using the simple summation method for integration, the eigenvectors are orthogonal for this integration method, but the result is scaled compared to normality by a factor $\frac{b-a}{n_x}$. Therefore it can be concluded that the inner product in (3.1) is best calculated using matrix multiplications, as the scaling of $C(\varphi_i(x))$ has no effect on the scaling of the resulting eigenvectors. The eigenvectors are normalized for vector multiplications.

3.1.3 Numeric derivative

In the previous sections it was shown that for a proper implementation of the reduced order model (2.29) some restrictions regarding the method used for numerical integration should be taken into account. This indicates that for a proper implementation, the method used for calculating numerical derivatives should also be investigated. Therefore the method for calculating numerical derivatives is investigated in this section.

For the reduced order model of the manufacturing system it is important that the boundary value at $x = 0$ can be explicitly specified since this is the input for the manufacturing system. To make this possible, the method of integration by parts was applied to the spatial dynamics of the reduced order model (2.22). This method of integration by parts however, imposes a constraint on the possible combinations of numerical integration techniques and derivation techniques that can be made. That is, for the method of integration by parts to hold, the method used for calculating numerical derivatives should match the method used for numerical integration. This is worked out below for the numerical integration method suggested in the previous section.

A simple matrix multiplication is suggested that allows the numerical derivative to be found:

$$\vec{f}_{n_x} = \boldsymbol{D}_{n_x} \cdot \vec{F}_{n_x}; \qquad \vec{F}_{n_x}, \vec{f}_{n_x} \in \mathbb{R}^{n_x}. \tag{3.11}$$

A general constraint on the derivative matrix $\boldsymbol{D}_{n_x}$ is that the derivative of a constant is zero i.e.:

$$\boldsymbol{D}_{n_x} \cdot \vec{k}_{n_x} = \vec{0}_{n_x}; \qquad \vec{k}_{n_x} \in \mathcal{N}(\boldsymbol{D}_{n_x}), \tag{3.12}$$

where all the elements in the vector $\vec{k}_{n_x}$ have the same value.

Another constraint is that the method of integration by parts (2.26) has to hold when numerical techniques are used for integration and differentiation. Rewriting (2.26) using

the suggested implementations for calculating numerical integrals (3.10) and derivatives (3.11), this results in:

$$\begin{aligned}
\int_a^b \vec{U}_{n_x}\vec{v}_{n_x} + \int_a^b \vec{V}_{n_x}\vec{u}_{n_x} &= \vec{U}_{n_x}\vec{V}_{n_x}\Big|_a^b \\
\int_a^b \vec{U}_{n_x}\boldsymbol{D}_{n_x}\cdot\vec{V}_{n_x} + \int_a^b \vec{V}_{n_x}\boldsymbol{D}_{n_x}\cdot\vec{U}_{n_x} &= \vec{U}_{n_x}\vec{V}_{n_x}\Big|_a^b \\
\vec{U}_{n_x}^\top\cdot\boldsymbol{D}_{n_x}\cdot\vec{V}_{n_x} + \vec{V}_{n_x}^\top\cdot\boldsymbol{D}_{n_x}\cdot\vec{U}_{n_x} &= \frac{n_x}{b-a}\vec{U}_{n_x}^\top\cdot\mathrm{diag}(-1,0,\ldots,0,1)\cdot\vec{V}_{n_x} \\
\vec{U}_{n_x}^\top\cdot\boldsymbol{D}_{n_x}\cdot\vec{V}_{n_x} + \vec{U}_{n_x}^\top\cdot\boldsymbol{D}_{n_x}^\top\cdot\vec{V}_{n_x} &= \frac{n_x}{b-a}\vec{U}_{n_x}^\top\cdot\mathrm{diag}(-1,0,\ldots,0,1)\cdot\vec{V}_{n_x} \\
\boldsymbol{D}_{n_x} + \boldsymbol{D}_{n_x}^\top &= \frac{n_x}{b-a}\mathrm{diag}(-1,0,\ldots,0,1).
\end{aligned} \tag{3.13}$$

So for the method of integration by parts to hold, the derivative matrix $\boldsymbol{D}$ should be of the following form:

$$\begin{aligned}
\boldsymbol{D}_{n_x}(1,1) &= -\frac{n_x}{2(b-a)} \\
\boldsymbol{D}_{n_x}(n,n) &= \frac{n_x}{2(b-a)} \\
\boldsymbol{D}_{n_x}(i,i) &= 0 \qquad\qquad \{1<i<n_x | i\in\mathbb{N}\} \\
\boldsymbol{D}_{n_x}(i,j) &= -\boldsymbol{D}_{n_x}(j,i) \quad i\neq j,
\end{aligned} \tag{3.14}$$

which also satisfies (3.12).

These necessary conditions still leave many choices for the method of numerical differentiation. Here a simple implementation is chosen:

$$\boldsymbol{D}_{n_x} = \frac{n_x}{2(b-a)}\begin{bmatrix} -1 & 1 & 0 & \ldots & 0 \\ -1 & 0 & 1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 0 & 1 \\ 0 & \ldots & 0 & -1 & 1 \end{bmatrix}. \tag{3.15}$$

With this method for determining a numerical derivative it is possible to write the reduced order model (2.29) in matrix and vector notation. In the following section an implementation for (2.29) is derived where the method for numerical differentiation as discussed in this section is taken into account.

3.1.4 Implementation

In this section the reduced order model (2.29) is written in vector and matrix notation, where the numerical techniques for computing integrals and derivatives, developed in the previous two sections, are implemented. This notation facilitates the implementation of the reduced order model, as matrices and vectors are typical elements used by computational tools like, for example, Matlab.

Consider a matrix that contains the individual basis vectors:

$$\boldsymbol{\Phi} = \begin{bmatrix} \vec{\varphi}_1 & \dots & \vec{\varphi}_n \end{bmatrix}, \tag{3.16}$$

where n denotes the number of basis functions that are taken into account, and the basis vector $\vec{\varphi}_i$ is a column vector that contains the values of the basis function $\varphi_i(x)$ for the discrete points in $\mathbb{X}$. The reconstructed variables $\hat{\rho}$ and $\hat{\rho}_x$ used in (2.27) can now be represented in vector notation:

$$\vec{\rho} = \sum_{i=1}^{n} a_i \vec{\varphi}_i = \boldsymbol{\Phi} \cdot \vec{a} \qquad\qquad \vec{\rho}_x = \sum_{i=1}^{n} a_i \vec{\varphi}_i' = \boldsymbol{D}_{n_x} \cdot \boldsymbol{\Phi} \cdot \vec{a}, \tag{3.17}$$

where $\vec{\rho}$ is a column vector that contains the density for the discrete elements in $\mathbb{X}$ for a specific moment in time, and column vector $\vec{a}$ contains the Fourier coefficients for this specific moment in time. The velocity and its derivatives for the simple model of the manufacturing system can then be written as:

$$v(i) = \frac{\mu}{m + \rho(i)} \qquad v_\rho(i) = \frac{-\mu}{\big(m + \rho(i)\big)^2} \qquad v_{\rho\rho}(i) = \frac{2\mu}{\big(m + \rho(i)\big)^3}, \tag{3.18}$$

where $\vec{\rho}(i)$ denotes the i-th element of the vector $\vec{\rho}$. The reduced order model (2.27) can now be written as:

$$\begin{aligned} \frac{1}{n_x} \dot{\vec{a}} = &- \Big(\mathrm{diag}(\mathrm{diag}(\vec{v}_\rho) \cdot \vec{\rho} + \vec{v}) \cdot \mathrm{diag}(\vec{\rho}) \cdot \boldsymbol{\Phi}\Big)^\top \Big|_{x=0}^{x=1} \\ &+ \frac{1}{n_x} \left(\boldsymbol{D} \cdot \boldsymbol{\Phi}\right)^\top \cdot \Big(\mathrm{diag}(\mathrm{diag}(\vec{v}_\rho) \cdot \vec{\rho} + \vec{v}) \cdot \vec{\rho}\Big) \\ &+ \frac{1}{n_x} \boldsymbol{\Phi}^\top \cdot \Big(\mathrm{diag}(\mathrm{diag}(\vec{v}_{\rho\rho}) \cdot \vec{\rho} + 2\vec{v_\rho}) \cdot \mathrm{diag}(\vec{\rho}) \cdot \vec{\rho}_x\Big). \end{aligned} \tag{3.19}$$

Note that the first element in the vector $\vec{\rho}$ corresponds to the density at $x = 0$, so when the incoming flux $\rho v|_{x=0}$ is written as λ this yields:

$$\begin{aligned}\frac{1}{n_x}\,\dot{\vec{a}} &= \left(\lambda - \vec{v}_\rho(1)\vec{\rho}(1)^2\right)\mathbf{\Phi}(1,:)^\top \\ &\quad - \left(\vec{v}_\rho(n_x)\vec{\rho}(n_x) + \vec{v}(n_x)\right)\vec{\rho}(n_x)\mathbf{\Phi}(n_x,:)^\top \\ &\quad + \frac{1}{n_x}\left(\boldsymbol{D}\cdot\mathbf{\Phi}\right)^\top \cdot \left(\mathrm{diag}(\mathrm{diag}(\vec{v}_\rho)\cdot\vec{\rho} + \vec{v})\cdot\vec{\rho}\right) \\ &\quad + \frac{1}{n_x}\,\mathbf{\Phi}^\top \cdot \left(\mathrm{diag}(\mathrm{diag}(\vec{v}_{\rho\rho})\cdot\vec{\rho} + 2\vec{v_\rho})\cdot\mathrm{diag}(\vec{\rho})\cdot\vec{\rho}_x\right),\end{aligned} \tag{3.20}$$

where $\mathbf{\Phi}(i,:)$ denotes the i-th row of the matrix $\mathbf{\Phi}$ (3.16). Using this implementation it is possible to perform simulations for the reduced order model of the manufacturing system. The results for such a simulation are shown in Figure 3.5.

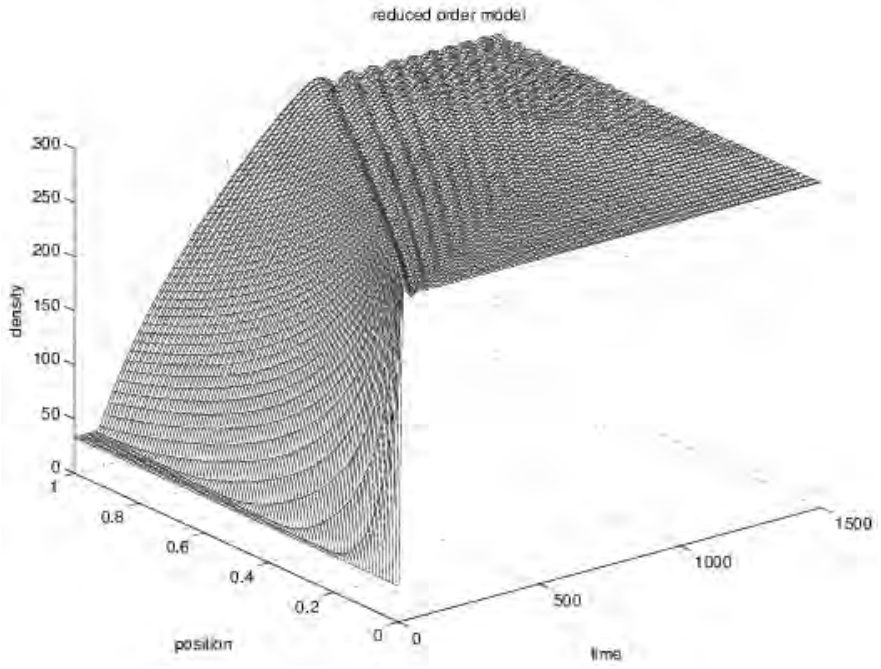


Figure 3.5: Simulation results for implementation using numerical derivative according to (3.15)

When the results shown in this figure are compared to the results for the results based on the analytical solution of the PDE model (Figure 3.1) it can be seen that the implementation as shown in Figure 3.5 captures the general behavior of the PDE model. However, the figure also shows some undesired oscillations in the density. These oscillations can be explained by the fact that the method used to calculate numerical derivatives (3.15) is partially based on the central difference method. This central difference method, combined with a step function that propagates through the system, is

known to cause oscillations [Ber04]. As the system is subject to a step function for the influx, this explains the oscillations in the simulation results.

In the first section of this chapter however, it could be seen that, apart from a scaling in time, the implementation using the integration method based on a simple summation did not show oscillations. This can be explained by the fact that for the implementation in that section a backward difference method was used to calculate the numerical derivative. To check if the oscillations shown in Figure 3.5 can be resolved by applying a backward difference method, a simulation is run where this method is implemented.

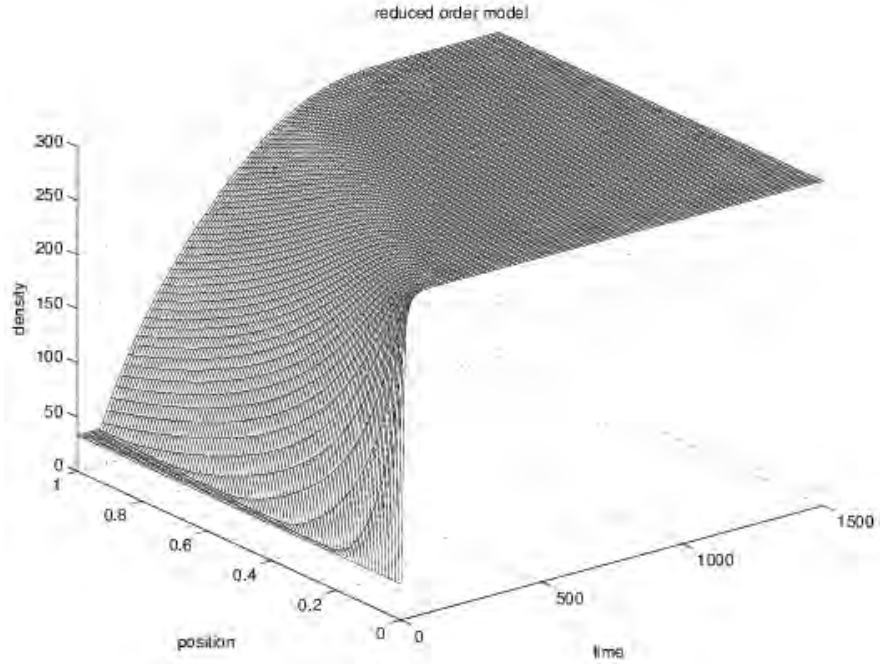


Figure 3.6: Simulation results for implementation using backward difference method

Figure 3.6 shows the results for the simulation based on the backward difference method. The figure shows that the oscillations are indeed caused by the method used to calculate numerical derivatives. That is, using a backward difference method for (3.20), no oscillations occur. Therefore it should be noted that though in section 3.1.3 it was suggested that a backward difference method should not be used, better simulation results are obtained with the backward difference method for numerical derivatives.

In the previous sections an implementation for the reduced order model of the manufacturing system, derived in the previous chapter, was developed. Using this implementation, simulations can be run for the model of the manufacturing system using a reduced set of POD basis functions. This reduced order model however, is of a rather complex form using many first and higher order derivatives. Therefore, in the following section a reduced order model is derived that is of a simpler form than the model observed

previously.

3.2 Alternative reduced order model

In the previous sections the reduced order model, derived in Chapter 2, was implemented allowing simulations to be run for this model. In this section a different reduced order model is derived for the PDE model of the manufacturing system as discussed in Example 2.1, which is of a simpler form than the model discussed until now.

3.2.1 Integration by parts using dx

In this section a different approach is taken to the derivation of a reduced order model for the PDE model discussed in Example 2.1, which results in a simpler form for the reduced order model. This is achieved by rewriting ρv in (2.10a) as q, the flux of lots, and performing the integration by parts with respect to x in stead of φ. The PDE-model for the manufacturing system then reads:

$$\rho_t = -q_x(\rho), \tag{3.21}$$

with:

$$q(\rho) = \rho v(\rho) \qquad\qquad q_x(\rho) = q_\rho(\rho)\rho_x. \tag{3.22}$$

Using the reconstructed variables from (2.28), The PDE model (3.21) can be written into:

$$\hat{\rho}_t = -q_\rho(\rho)\hat{\rho}_x. \tag{3.23}$$

When the Galerkin projection (2.20) is applied to the residual for (3.23), this results in:

$$\big((\hat{\rho}_t + q_\rho(\rho)\hat{\rho}_x), \varphi_j(x)\big) = 0, \tag{3.24}$$

which using (2.24) and the orthonormality of the basis functions can be rewritten into:

$$\dot{a}_j(t) = -\int_0^1 \varphi_j q_\rho(\rho)\hat{\rho}_x dx. \tag{3.25}$$

Using integration by parts (2.26) with:

$$\begin{aligned} f &= \varphi_j & dg &= -q_\rho(\rho)\hat{\rho}_x dx \\ df &= \varphi_j' dx & g &= -q(\rho), \end{aligned}$$

this results in:

$$\dot{a}_j(t) = -\varphi_j(x)q(\rho)\Big|_0^1 + \int_0^1 q(\rho)\varphi_j' dx. \tag{3.26}$$

Using the boundary condition for $x = 0$ (2.10c) and the relation for the velocity v in (3.18), this can be written as:

$$\dot{a}_j(t) = \varphi_j(0)\lambda - \varphi_j(1)\hat{\rho}(1)\hat{v}(1) + \int_{x=0}^1 \hat{\rho}\hat{v}\varphi_j' dx. \tag{3.27}$$

Using the implementations for numerical integration and derivatives as derived in the previous section, the reduced order model can be written in vector notation:

$$\frac{1}{n_x}\dot{\vec{a}} = \lambda \mathbf{\Phi}(1,:)^\top - \vec{\rho}(n_x)\vec{v}(n_x)\mathbf{\Phi}(n_x,:)^\top + \frac{1}{n_x}\left(\mathbf{D}\cdot\mathbf{\Phi}\right)^\top \cdot (\mathrm{diag}(\vec{\rho})\cdot\vec{v}). \tag{3.28}$$

It should be noted that this reduced order model (3.27) for the manufacturing system is of a less complicated form than the reduced order model derived in Chapter 2. Thereby, where the form of the previous reduced order model depended on the relation used for the velocity, the reduced order model developed here can be easily adapted for different relations for the velocity as these changes do not affect the general form of newly developed reduced order model.

The results for a simulation using the reduced order model as derived in this section is shown in Figure 3.7. The figure shows again some undesired oscillations in the density, caused by the central difference method used for calculating numerical derivatives. In the previous section it was shown that these oscillations could be resolved by applying the backward difference method. For the model derived in this section however, the backward difference method did not allow simulations to be run. This might be explained by the constraints imposed by the method of integration by parts, as discussed in Section 3.1.3.

As a backward difference method can not be used for this model, a different solution is needed to resolve the oscillations as shown in Figure 3.7. In the following section therefore, an alternative method for solving the mass conservation law, which forms the base of the PDE model, is used.

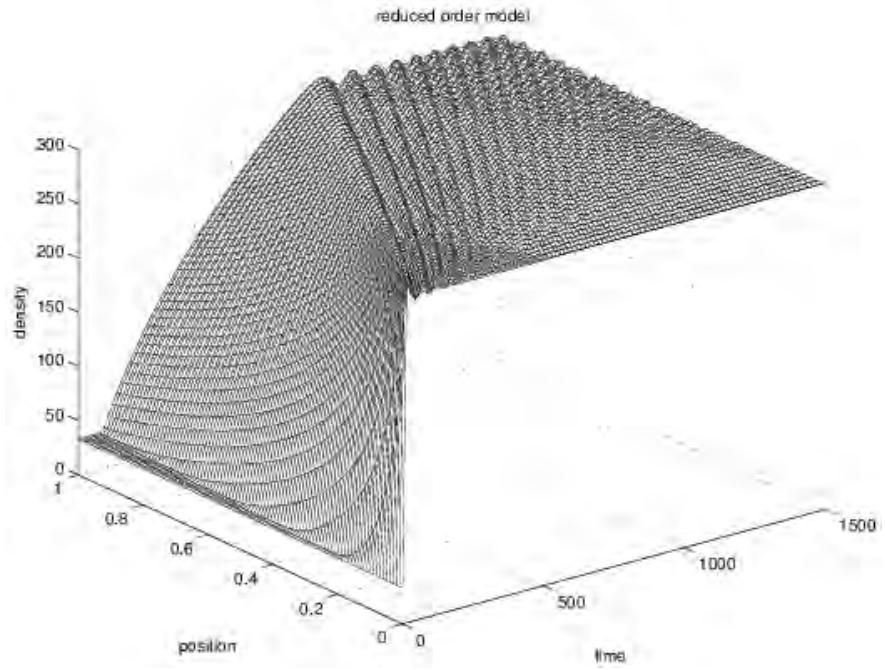


Figure 3.7: Simulation results for implementation of (3.28)

3.2.2 Godunov method

The previous sections discussed several issues related to the use of numerical integration and differentiation methods for solving reduced order models for a manufacturing system. The original model is based on a conservation law in differential form. This form however requires the use of finite integration and derivative methods to obtain solutions for specific cases. In this section a finite volume method is applied to the integral form of the PDE model for the manufacturing system, which removes the need for integration by parts. This means that the constraint on the method used for calculating numerical derivatives, imposed by the method of integration by parts, no longer applies. Another effect is that an alternative method should be used to implement the influx of lots into the manufacturing system at the boundary $x = 0$.

To obtain a model based on a finite volume method, a first order accurate Godunov method [Veq02] is applied to the model discussed in the previous subsection (3.21). It is expected that this finite volume method will solve the numerical problems associated with the models in differential form, as discussed in the previous sections.

To obtain a solution for the PDE model, the spatial domain is again subdivided into grid cells. The finite volume method is now based on keeping track of an approximation to the integral of the density ρ for every single grid cell. This integral denotes the number of lots (WIP) present at an individual grid cell [Arm02]. The values are updated for

every step in time using approximations for the flux through the boundaries of the grid cells. This means that for grid cell i at time t_n the value WIP_i^n is approximated by:

$$\text{WIP}_i^n \approx \int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t_n) dx; \qquad i = 1, 2, \ldots, N. \tag{3.29}$$

It can be seen that when WIP_i^n is determined using a method that is written in conservative form, the discrete sum $\sum_{i=1}^{N} \text{WIP}_i^n \Delta x$ over all the N grid cells will only change due to the fluxes at the outmost boundaries. That is, the amount of WIP in the manufacturing system only changes due to the flux of lots into the system at $x = 0$ and the flux of lots out of the system at $x = 1$. This means that the approximate numerical method still accurately follows the principle of the conservation of mass, or in the case for the manufacturing system, the conservation of lots.

The integral form of the PDE model for the manufacturing system (3.21) is given by:

$$\frac{d}{dt} \int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t) dx = q(\rho(x_{i-1}, t)) - q(\rho(x_i, t)). \tag{3.30}$$

It should be noted that here the flux of lots into grid cell i is the flux of lots that leaves the grid cell $i - 1$. This corresponds to the backward difference technique as used for the differential form of the PDE model, and matches the actual behavior of a manufacturing line as the flux of lots into a workstation is the flux of lots out of the previous workstation. Integrating (3.30) in time from t_n to t_{n+1} results in:

$$\int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t_{n+1}) dx - \int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t_n) dx = \int_{t_n}^{t_{n+1}} q(\rho(x_{i-1}, t)) dt - \int_{t_n}^{t_{n+1}} q(\rho(x_i, t)) dt, \tag{3.31}$$

which can be rewritten as:

$$\int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t_{n+1}) dx = \int_{\frac{i-1}{N}}^{\frac{i}{N}} \rho(x, t_n) dx + \int_{t_n}^{t_{n+1}} q(\rho(x_{i-1}, t)) dt - \int_{t_n}^{t_{n+1}} q(\rho(x_i, t)) dt. \tag{3.32}$$

That is, the total amount of lots present in grid cell i at time t_{n+1} is the amount of lots present at time t_n plus the total amount of lots that entered the grid cell in the time span $t_{n+1} - t_n$, minus the amount of lots that has left the grid cell within this time span. It should be noted however, that the time integrals on the right hand side of (3.32) can only be approximated. These time integrals cannot be calculated exactly since the flux q varies in time along each edge of the grid cell.

The mass conservation law (3.32) can be approximated using a first order Godunov method. This method uses a piecewise constant reconstruction of the density distribution for a specific moment in time (t_n), which means that the density within a grid cell is assumed to be constant. The density only changes at the boundaries of the grid cells, resulting in discontinuities between adjacent grid cells. Now the density within each of the grid cells is allowed to evolve in time until $t = t_{n+1}$. Here it is assumed that the flux of lots does not change within the time span $t_{n+1} - t_n$. Using these assumptions, the mass conservation law (3.32) can be written as:

$$\mathrm{WIP}_i^{n+1} = \mathrm{WIP}_i^n - \Delta t(Q_i^n - Q_{i-1}^n), \tag{3.33}$$

where (3.29) is used to obtain the amount of WIP for a specific grid cell at a specific moment in time, and the flux of lots is assumed constant on the time interval $t_{n+1} - t_n$. This means that the flux at which lots leave the grid cell i within the time span $t_{n+1} - t_n$ only depends on the density within this grid cell at time $t = t_n$:

$$Q_i^n = q(\rho(x_i, t_n)). \tag{3.34}$$

This flux can be calculated using the first relation in (3.22). When relation (3.18) is used for the velocity, the mass conservation law (3.33) can be written as:

$$\mathrm{WIP}_i^{n+1} = \mathrm{WIP}_i^n - \Delta t \left(\frac{\mu\rho(x_i, t_n)}{m + \rho(x_i, t_n)} - \frac{\mu\rho(x_{i-1}, t_n)}{m + \rho(x_{i-1}, t_n)} \right), \tag{3.35}$$

or when written in terms of the amount of WIP:

$$\mathrm{WIP}_i^{n+1} = \mathrm{WIP}_i^n - \Delta t \left(\frac{\mu\mathrm{WIP}_i^n}{1 + \mathrm{WIP}_i^n} - \frac{\mu\mathrm{WIP}_{i-1}^n}{1 + \mathrm{WIP}_{i-1}^n} \right). \tag{3.36}$$

It should be noted that in (3.36), the amount of WIP is related to the density ρ through:

$$\mathrm{WIP}_i^n = \frac{\rho(x_i, t_n)}{m}, \tag{3.37}$$

which corresponds to (3.29) with $N = m$ when the density distribution is assumed to be piecewise constant. Since the number of grid cells is limited to m, every single grid cell in (3.36) corresponds to a single workstation within the manufacturing line. When the limit of (3.36) is taken where Δt approaches 0, this results in an ODE:

$$\frac{d}{dt}\mathrm{WIP}_i = \lim_{\Delta t \to 0} \frac{\mathrm{WIP}_i^{n+1} - \mathrm{WIP}_i^n}{\Delta t} = \frac{\mu\mathrm{WIP}_{i-1}^n}{1 + \mathrm{WIP}_{i-1}^n} - \frac{\mu\mathrm{WIP}_i^n}{1 + \mathrm{WIP}_i^n}. \tag{3.38}$$

The model for the manufacturing system can now be written as a set of ODEs, one ODE for every single workstation. The boundary condition for the first workstation in the line at $x = 0$ can be implemented by using the influx $\lambda(t)$ in stead of the outflux of the "previous" workstation. The full order model then reads:

$$\begin{aligned} \frac{d}{dt}\mathrm{WIP}_1 &= \lambda(t) - \frac{\mu \mathrm{WIP}_1^n}{1 + \mathrm{WIP}_1^n} \\ \frac{d}{dt}\mathrm{WIP}_i &= \frac{\mu \mathrm{WIP}_{i-1}^n}{1 + \mathrm{WIP}_{i-1}^n} - \frac{\mu \mathrm{WIP}_i^n}{1 + \mathrm{WIP}_i^n} \qquad i = 2, \ldots, m. \end{aligned} \tag{3.39}$$

This full order model can be reduced using a reduction method similar to the POD reduction method as discussed in Chapter 2. The POD method however, is now applied to a model that consists of a set of ODEs in stead of a model based on a PDE. This method is discussed in the following section.

3.2.3 Model reduction for a set of ODEs

In the previous section a full order finite volume model was derived for the manufacturing system. To obtain a reduced order model however, the POD reduction technique should be applied to the set of ODEs that make up the model for the manufacturing system (3.39).

The POD reduction method for a set of ODEs is similar to the reduction method for a PDE, as discussed in Chapter 2. The basis functions can be derived using exactly the same method as discussed in Section 2.2. To obtain a reduced order model however, the Galerkin projection (2.20) now has to be applied to a set of ODEs.

Given a set of variables $p(t) \in \mathbb{R}^l$ and the set of ODEs $\dot{p}(t) = f(p)$. It can be seen that the structure of this set of ODEs matches the from:

$$M(f(x,t)) = D(f(x,t)), \tag{3.40}$$

as discussed in Section 2.3, with:

$$\begin{aligned} M(f(x,t)) &= \dot{p}(t) \\ D(f(x,t)) &= f(p). \end{aligned}$$

That is, $M(f(x,t))$ contains only a first order temporal derivative and $D(f(x,t))$ does not contain any derivatives in x at all, it is simply a function of the variables in p. Again a residual function can be defined as:

$$R(f(x,t)) = M(f(x,t)) - D(f(x,t)). \tag{3.41}$$

Now using the POD basis as defined in (2.7) and the truncated spectral expansion (2.3), the variables $p(t)$ in $D(f(x,t))$ can be reconstructed using:

$$p(t) = \sum_{i=1}^{n} a_i(t)\varphi_i(x), \tag{3.42}$$

and the temporal derivatives in $M(f(x,t))$ can be reconstructed using:

$$\dot{p}(t) = \sum_{i=1}^{n} \dot{a}_i(t)\varphi_i(x). \tag{3.43}$$

Using the Galerkin projection as defined in (2.20) on the residual (3.41) then results in a reduced order model for the set of ODEs:

$$\dot{a}_j(t) = \left(\varphi_j(x), f\Big(\sum\nolimits_{i=1}^{n} a_i(t)\varphi_i(x)\Big)\right); \qquad j = 1, \ldots, n, \tag{3.44}$$

where n denotes the number of basis functions that are taken into account for the reduction. For the implementation of a reduced order model, the inner product $(.,.)$ and the reconstruction of p (3.42) are written using matrix multiplications:

$$(\vec{s}, \vec{t}) = \vec{s}^{\top} \cdot \vec{t} \tag{3.45}$$

$$\vec{p}(t) = \mathbf{\Phi} \cdot \vec{a}(t), \tag{3.46}$$

where the elements from a,p,s and t are captured in the corresponding vectors $\vec{a}$,$\vec{p}$,$\vec{s}$ and $\vec{t}$ respectively and $\mathbf{\Phi}$ contains the individual basis functions as defined in (3.16). The implementation for a reduced order model (3.44) can then be written as:

$$\dot{\vec{a}}(t) = \mathbf{\Phi}^{\top} \cdot f\big(\mathbf{\Phi} \cdot \vec{a}(t)\big), \tag{3.47}$$

It should be noted however, that for this reduced order model no boundary conditions are taken into account as these are not necessary for a set of ODEs. For the model of the manufacturing system, as discussed in the previous section, this does not provide a problem since the flux of lots into the system is specified directly in the first ODE of this model. The reduced version of the model for the manufacturing system can be obtained using (3.47), where $f(p)$ is the right hand side of (3.39) and p is reconstructed using (3.46).

In this chapter two implementations were discussed for different reduced order models. For the simulation results shown in this chapter however, no actual reduction was performed. For these simulations all the individual basis functions were taken into account. The performance of the reduced order models, where the number of POD basis functions that are taken into account is actually reduced, is discussed in the following chapter.

Chapter 4

Performance of the reduced order models

In the previous chapters two different reduced order models were derived for the manufacturing line, discussed in Example 2.1. One model was based on the method of integration by parts (3.20), and the other was derived using a first order Godunov method (section 3.2.2). In this chapter the performance of these models is evaluated. This is done by comparing the simulation results of these models for a specific test case with the results of simulations performed using a discrete event model (DEM), a proven method for simulating the behavior of manufacturing systems. The first section discusses the method used to obtain the validation data. In the second section this validation data is used to analyze the performance of the models when no reduction is performed. The third section discusses the effect of an actual reduction on the performance of the observed models, and in the last section it is investigated if the performance of the reduced order models can be improved by using data obtained from DEM simulations to determine the POD basis functions.

4.1 Discrete event model

The purpose of the PDE model for the manufacturing system, discussed in Example 2.1, is to derive a control strategy for this system. Therefore it is important that the continuous approximation model is able to capture the dynamical behavior of the manufacturing system. To validate the performance of the reduced order models, based on the method of integration by parts and a first order Godunov method, simulations are performed using a discrete event model (DEM) that describes the behavior of the observed manufacturing line. The manufacturing line discussed in Example 2.1 consists of 100 identical workstations (Figure 4.1).

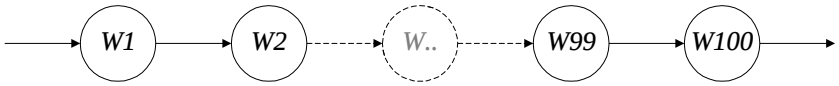


Figure 4.1: Observed manufacturing system

Each workstation within this manufacturing line consists of a buffer with infinite capacity, followed by a single machine with exponentially distributed process times t_e (Figure 4.2).

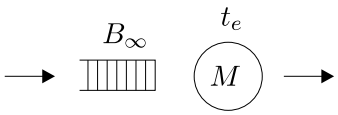


Figure 4.2: Single machine workstation

In queueing theory this corresponds to a network of $M/M/1$ systems. It should be noted that the relation between the density ρ and the velocity v in the PDE model used throughout this thesis is actually based on the queuing theory for these $M/M/1$ systems. The DEM used to simulate the behavior of the observed system is implemented in χ and is discussed in more detail in Appendix A.1.

Since the observed manufacturing system is based on machines with stochastic process times, it is not possible to accurately determine the general behavior of the system from one simulation instance. The variability present in the system causes a different outcome to occur for every single simulation. To obtain suitable measurements, the outcome of individual simulations should be averaged to obtain information on the general behavior of the manufacturing system. The method used to average the measured data is illustrated in Figure 4.3.

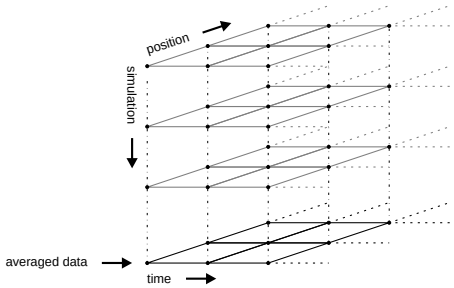


Figure 4.3: Method used for averaging multiple simulation results

For every single simulation run, samples on the amount of WIP within each workstation are taken at fixed intervals in time. As the individual simulation runs are independent, different values for a specific point in place and time can be averaged [Pla04]. This process is automated in a python script that performs new simulations until for the averaged results of every single point in place and time, a specified relative error from the true mean is met (Appendix B).

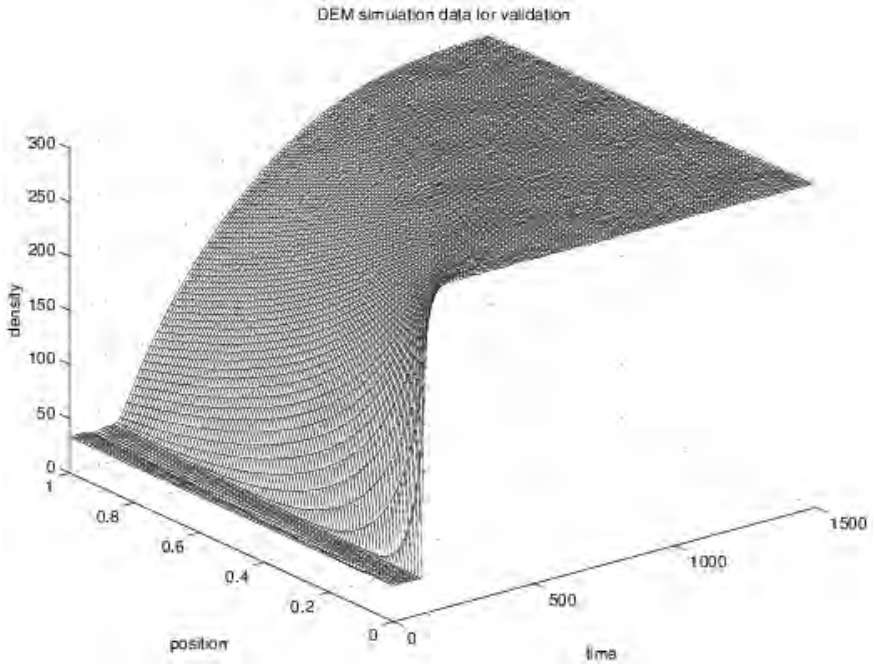


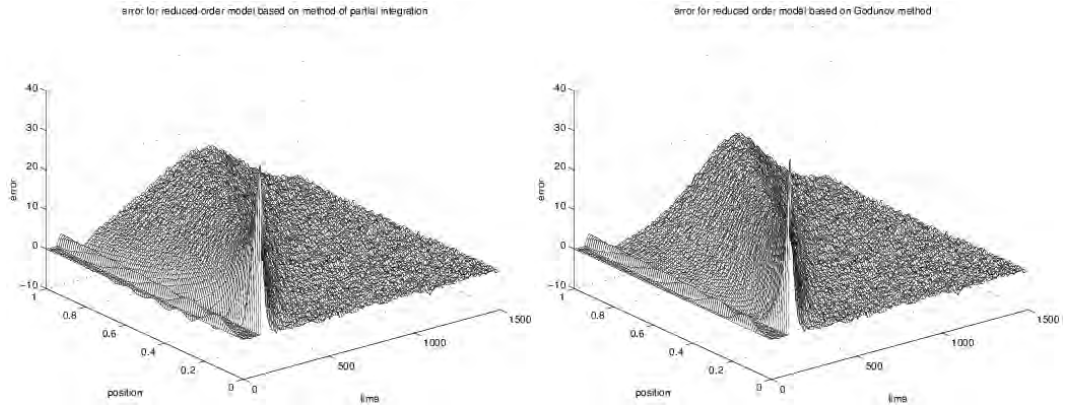
Figure 4.4: Averaged DEM simulation results for validation test case

Figure 4.4 shows the averaged simulation results of the DEM for the test case as discussed in the previous chapter. It should be noted that the relation between the WIP and density ρ is used as stated in (3.37), to determine the density distribution from the DEM simulation results. It can be seen in the figure that the system is ramped up at time $t = 100$, this is done to make sure that the system is initially in steady state when the system is ramped up. Using these averaged results of the different simulation runs, the performance of the reduced order models can be validated. This is discussed in the following sections.

4.2 Performance of full order models

In the previous section verification data for the test case, as discussed in the previous chapter, was obtained from multiple DEM simulations. Using this verification data, the performance of the reduced order models can be validated. In this section however,

no actual reduction is performed, the models are validated using all the available POD basis functions.



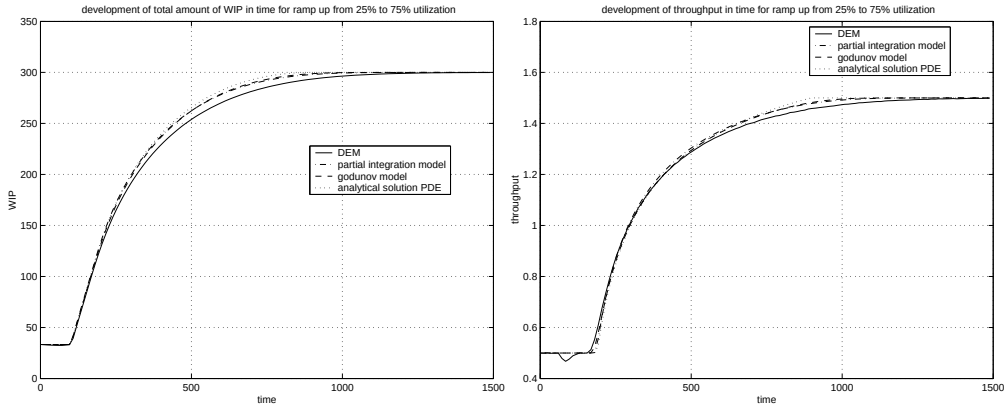
(a) integration by parts (b) Godunov method

Figure 4.5: Error of reduced order models with DEM validation results

Figures 4.5a and 4.5b show the difference with the DEM simulation results for the model based on the method of integration by parts and the model based on a first order Godunov method respectively. It can be seen in these figures that the largest deviation occurs near the entrance of the manufacturing line, when the influx is changed instantly ($x = 0$, $t = 100$). The relative large error at this location can be explained by the fact that as the density at this location changes dramatically, a small error in the response time results in a rather large error for the density. The figures 4.5a and 4.5b also show that this error propagates throughout the complete transient state.

Figure 4.6a and Figure 4.6b respectively show the effect of this error in the transient state on the development of the total amount of WIP and the development of the throughput. It can be seen in these figures that the transient behavior of the manufacturing line is slightly better approximated using the implementations for the reduced order models than by the discretized analytical solution for the PDE model. It should be noted however, that the error with the DEM validation data does not differ much for the different implementations for the reduced order models. The maximal error, or difference in density with the validation data, for the model based on the method of integration by parts is 42.62 while the maximal error for the model based on the Godunov method is 44.43. These relative large errors however (approximately 13% of the desired density), are not representative for the general difference in densities. The averaged errors over the observed spatial and temporal domain are 4.53 and 4.48 respectively. As both the maximal errors and the averaged errors do not differ much, the remaining part of this thesis will discuss the model based on the Godunov method, since this model is of a simpler form than the model based on the method of integration by parts.

The results in this section show that the PDE model used in this thesis is not able to



(a) WIP (b) Throughput

Figure 4.6: Performance measures for reduced order models

accurately describe the transient behavior of the observed manufacturing system. This conclusion however, was already made in [Ber04]. Though more accurate PDE models are already suggested in [Pla04], the basic PDE model used in this thesis is still suitable to show the effect of model reduction. This effect is discussed in the following section.

4.3 Performance of reduced order model

In the previous section the performance of the reduced order models, derived in section 2.3 and section 3.2.2, was discussed when no actual reduction was performed. As the difference in performance between these two reduced order models for the manufacturing system was negligible, this section will discuss the performance of the reduced order model that has the simplest form, the model based on the Godunov method.

The performance is again tested for the first 12 POD basis functions, the same basis functions as used to illustrate the effect of POD reduction in section 2.2. The performance is tested for the case as discussed in the previous chapter. That is, a simulation is performed for both the full order model and the reduced order model, where the system is ramped up from a state corresponding to a 25% utilization level to a state that corresponds to a 75% utilization level.

The difference between the full order model and reduced order model results for this test case is shown in Figure 4.7. It can be seen in this figure that the largest difference in density (13.01 [lots/place]) occurs at the entrance of the manufacturing system ($x = 0$) at the moment that the influx is changed instantly ($t = 100$). This can again be explained by the fact that the POD basis functions are determined for the time averaged behavior shown in Figure 2.1. As for this data the discontinuity in the density only

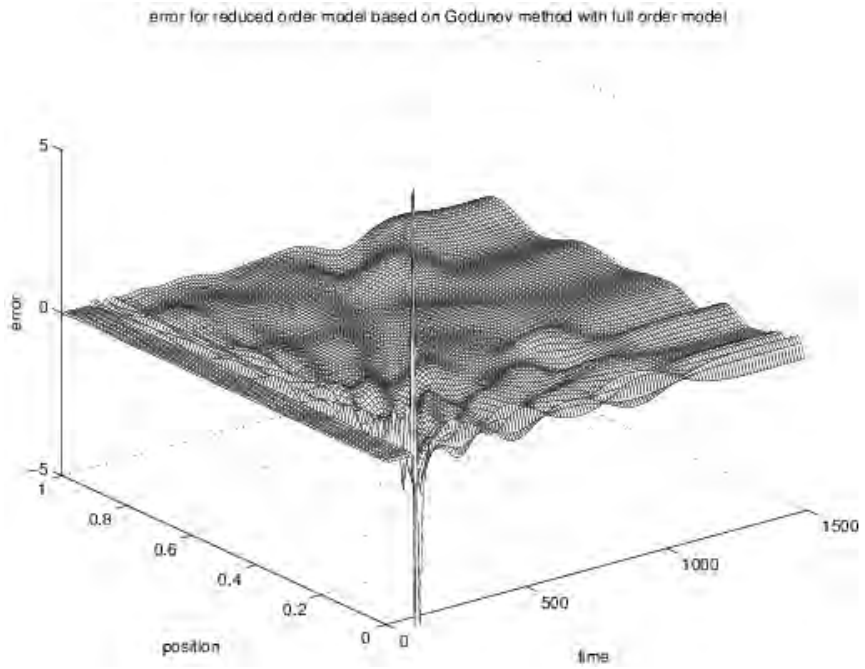


Figure 4.7: Effect of reduction on validation test case for the model based on the Godunov method

occurs at time $t = 0$, a reduced set of POD basis functions based on this data will not be able to reconstruct the discontinuity accurately.

Figure 4.7 also shows that the errors in density for other locations in place and time are not larger than 1.0 [lots/place]. In fact, the averaged absolute error, over the complete temporal and spatial domain is only 0.13 [lots/place]. This indicates that the error introduced by the reduction is of a smaller order than the error between the simulation results for the full order model and the DEM simulation results. This means that the reduced order model, using only roughly a tenth of the available POD basis functions, performs similar to the full order model.

Simulation	time [s]
Simulation based on exact solution	2.31
Full order model based on method of integration by parts	13.60
Reduced order model based on method of integration by parts	0.71
Model based on Godunov method	0.77
Full order model based on Godunov method	0.83
Reduced order model based on Godunov method	0.30

Table 4.1: Characteristic simulation times for different implementations

Table 4.1 shows some typical simulation times for both full order and reduced order models of the manufacturing system based on the first 12 basis functions. It can be seen in this table that the largest reduction in simulation time, both absolute and relative, is realized for the reduced order model based on the method of integration by parts. This can partially be explained by the fact that for the implementation of these models (3.20) more calculations are performed with $\mathbf{\Phi}$, the matrix that contains the basis functions, than for the implementations of the reduced order models based on the Godunov method (3.44). The table also shows that the simulation for the reduced order model based on the method of integration by parts requires more computation time than the reduced order model based on the Godunov method. This result matches the observation that the reduced order model based on the Godunov method is of a simpler form than the model based on the method of integration by parts.

It should be noted that the performance of the reduced order model is tested for a case *different* from the case used to obtain the POD basis functions. For the training and verification cases used in this thesis, this does not introduce large errors as these cases are very similar. The only difference is the scale of the ramp up that is performed, the basis functions are obtained from a training case where the system is ramped up to a 50% utilization level, while in the validation case the system is ramped up to a 75% utilization level. That is, the POD basis functions allow optimal reduction for models that show similar spatial dynamical behavior as present in the data used to obtain these basis functions. This could mean that the performance of the reduced order model, compared to the DEM validation data, might be improved by obtaining the POD basis functions from DEM simulation results for the training case as discussed in section 2.2. The reduced order model, using POD basis functions determined from DEM simulation results, might perform better as these basis functions are determined from data that better matches the dynamical behavior captured in the validation data. This is tested in the following section.

4.4 POD based on DEM simulations

In the previous section the effect of reduction on the accuracy of the reduced order model was discussed. It was suggested that the accuracy the reduced order model based on the Godunov method, compared to the DEM validation data, might be improved by using POD basis functions that are determined from DEM simulation results. As a single simulation instance does not provide accurate information on the general behavior of the manufacturing system, again many simulations need to be performed to determine the general behavior of the manufacturing system.

Figure 4.8 shows the averaged results of the multiple simulations for the DEM. It should be noted that, similar to the DEM validation data obtained in section 4.1, the ramp up of the system is not performed immediately. The DEM is first allowed to reach the initial steady state before the ramp up is performed. This initialization data however,

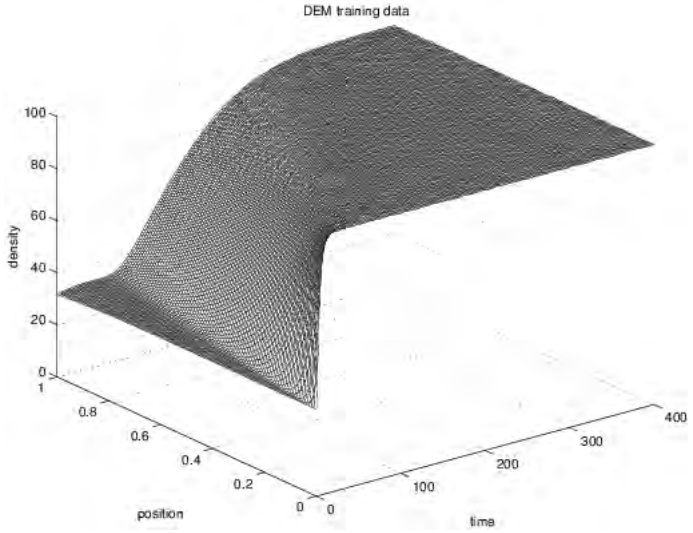


Figure 4.8: Averaged DEM data of the manufacturing model, used to determine the POD basis

is not taken into account for the calculation of the POD basis functions.

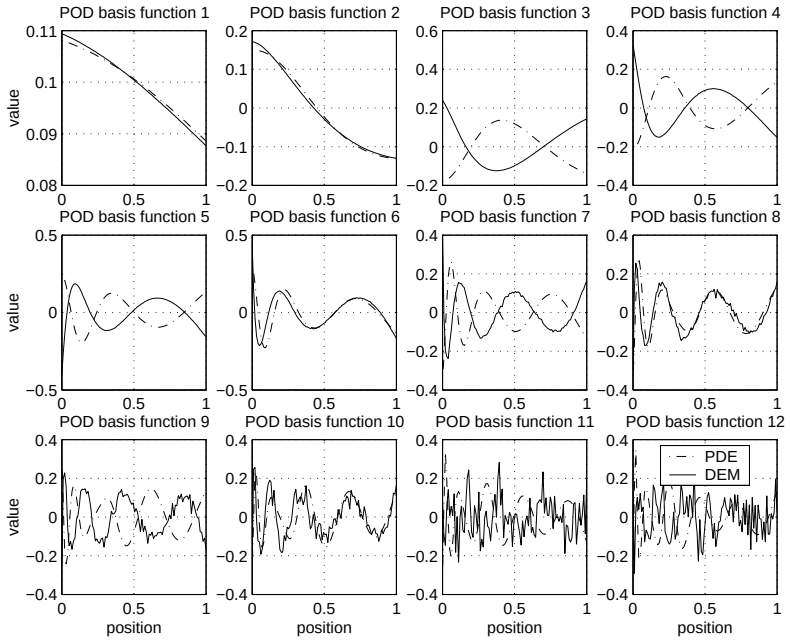
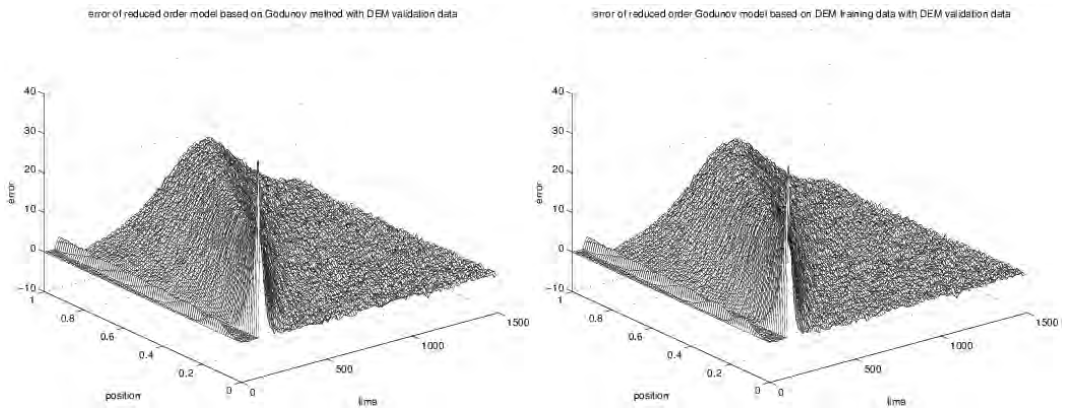


Figure 4.9: The first 9 POD basis used to approximate the data of the DEM simulations.

The first 12 POD basis functions derived from the DEM simulation data, are shown

in Figure 4.9, together with the POD basis functions derived in section 2.2. Though the figure shows that the first modes of the DEM resemble those determined for the analytical solution to the PDE, there are some small differences. It should be kept in mind however, that the PDE-model, proposed in Example 2.1, was found to be unsuitable for modelling the transient behavior of the observed manufacturing system accurately. This could account for the difference in the first POD basis functions. Apart from these small differences for the first basis functions, it can be seen that higher order basis functions show some noise. This noise is the result of the averaging multiple simulation results and indicates that more simulations need to be performed to obtain more smooth results. It should be noted however, that the number of simulations run to obtain the data used here to determine DEM based POD basis functions lies in the order of $\mathcal{O}10^6$. To test if the performance of the reduced order model can be improved by using the POD basis functions determined from the DEM simulation results, the performance of the reduced order model is analyzed for both sets of POD basis functions.

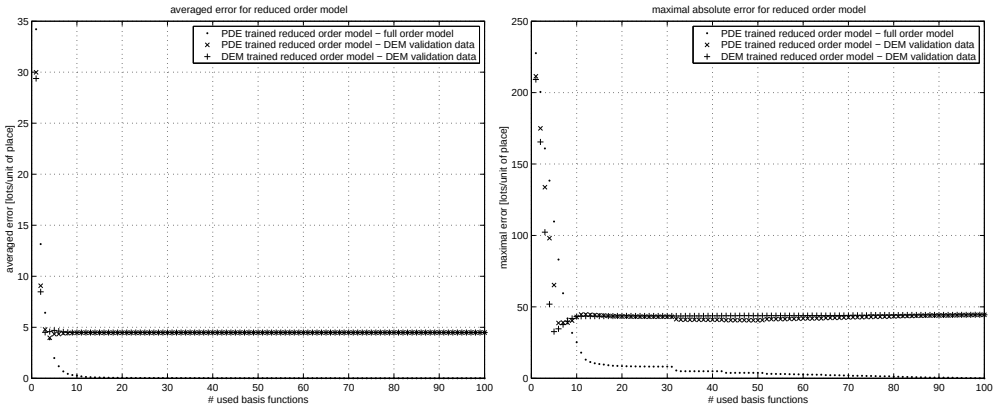


(a) PDE based (b) DEM based

Figure 4.10: Error of reduced order model based on Godunov method with DEM validation data

Figure 4.10 shows the differences of the reduced order model based on both the POD basis functions derived from the PDE based data and the DEM based data with the validation data, using the first 12 basis functions. It can be seen that no improvement is obtained when the POD basis functions are determined from DEM simulation results. The maximal absolute error changes from 44.66 [lots/place] for the model based on the PDE data to 43.50 [lots/place] for the model based on DEM simulation results, while the averaged error changes from 4.46 [lots/place] to 4.48 [lots/place] respectively.

Figure 4.11 shows these averaged and maximal errors for different levels of reduction. It can again be seen in this figure that the performance of the reduced order model does not improve when the POD basis functions are determined from DEM simulation data. When more basis functions are taken into account, the errors quickly stabilize onto the



(a) averaged error (b) maximal absolute error

Figure 4.11: Errors for reduced order models

errors of the full order model with the DEM validation data. In fact, Figure 4.11b suggests that a stronger reduction results in a smaller maximal error compared to the DEM validation data. This improvement might not occur for other cases, but the figure does show that the reduced order model could as well be based on less than 12 basis functions, as in this case the error with the DEM validation data is mainly the result of the performance of the original PDE model.

In this chapter the effect of reduction on the Godunov based model was discussed. It was shown that the number of ODEs that need to be solved to perform simulations for the reduced order model could be reduced, while still obtaining acceptable accurate results. It was also shown that the performance of the reduced order model did not improve when the POD basis functions were derived from DEM simulation results. The DEM based POD basis functions do not improve the dynamical behavior of the governing equations, they only (though minimally) affect the performance of the reduction.

Apart from the reduction in the computation time needed to perform simulations for these reduced order models, the reduction in the number of ODEs can also be useful for the design of a controller. Therefore in the following chapter it is investigated if it is possible to determine a control strategy for manufacturing lines by applying a backstepping method to the model derived using the Godunov method (3.39).

Chapter 5

Backstepping

In Chapter 3 an approximation model was proposed for a manufacturing line that is based on a first order Godunov method (3.39). This approximation model consists of a set of nonlinear Ordinary Differential Equations (ODEs), where every single ODE represents the behavior of a single workstation within the manufacturing line. This form allows the use of a backstepping procedure to find a stabilizing feedback controller for the approximation model.

The goal for the control strategy is to stabilize the throughput of the continuous approximation model onto a reference trajectory and to compensate for permanent backlog. Backlog is formed when the system can not keep up with a sudden change in the requested throughput of the system, which results in a shortage or excess of lots. The control strategy is derived using a backstepping approach [Sas99] and can be adapted for models of arbitrary length.

First the general structure of the model is discussed, after which the control strategy is derived for a model that consists of a single workstation. The performance of this control strategy is analyzed using a test case where the demand suffers from seasonal influences. To study the effect of variability on the performance of the control strategy, the strategy is applied to a discrete event model (DEM) that represents the stochastic behavior of a manufacturing system. The procedure regarding the single workstation manufacturing line is then repeated for a system consisting of two workstations. Finally, a general structure for the control strategy is given, which allows control strategies for longer manufacturing lines to be derived.

5.1 Model structure

The continuous approximation model (3.39) is rewritten into the following form:

$$\begin{aligned}
\dot{x}_m &= u - \frac{\mu x_m}{x_m + 1} \\
\dot{x}_{m-1} &= \frac{\mu x_m}{x_m + 1} - \frac{\mu x_{m-1}}{x_{m-1} + 1} \\
&\vdots \\
\dot{x}_1 &= \frac{\mu x_2}{x_2 + 1} - \frac{\mu x_1}{x_1 + 1} \\
\dot{y} &= \frac{\mu x_1}{x_1 + 1}.
\end{aligned} \tag{5.1}$$

In this set of ODEs the variables x_i denote the number of lots present at a specific workstation, or the work in progress (WIP). It is important to notice that the variables x_i are in reversed order compared to the model derived in Chapter 3. That is, x_1 denotes the WIP for the last workstation in the manufacturing line and x_m denotes the WIP for the first workstation in the system. These reversed coordinates match the order in which the backstepping technique is applied to the model, and therefore simplify the backstepping procedure.

Though a better approximation to the original PDE might be obtained with a smaller step size, it is assumed here that a single ODE represents the behavior of a single workstation. A single ODE in (5.1) therefore represents the change of the amount of lots within a particular workstation. This change is the result of lots leaving and entering the workstation at a particular rate. It can be seen that the structure of (5.1) matches the idea that in a manufacturing line the flow of lots into a workstation is the outflux of the preceding workstation, where the rate at which lots leave the workstation is dependent on the amount of lots present in the workstation. For practical reasons the relation between the amount of WIP in a workstation and the outgoing flux is captured in a separate variable z:

$$z_i = \frac{\mu x_i}{x_i + 1}. \tag{5.2}$$

The model can now be written as:

$$\begin{aligned}
\dot{x}_m &= u - z_m \\
\dot{x}_{m-1} &= z_m - z_{m-1} \\
&\vdots \\
\dot{x}_1 &= z_2 - z_1 \\
\dot{y} &= z_1
\end{aligned} \tag{5.3}$$

It should be noted that the actual output y of the system is the total amount of lots that have left the manufacturing system (the integrand of the outflux z_1 of the system). By controlling this output it is possible to compensate for permanent backlog.

5.2 Controller design for a single workstation manufacturing line

In this section the first step is taken to develop a control strategy for a manufacturing line that consists of identical workstations. The control strategy is derived for a model of a single workstation manufacturing line:

$$\begin{aligned} \dot{x}_1 &= u - z_1 \\ \dot{y} &= z_1. \end{aligned} \tag{5.4}$$

The output y of the manufacturing system is supposed to follow a reference trajectory y_r. Therefore, the first error variable in the backstepping procedure is defined as:

$$\varepsilon_1 = y - y_r. \tag{5.5}$$

The derivative for this error variable can be rewritten using the second relation from (5.4):

$$\begin{aligned} \dot{\varepsilon}_1 &= \dot{y} - \dot{y}_r \\ &= z_1 - \dot{y}_r. \end{aligned} \tag{5.6}$$

To stabilize the difference between the output and the requested output (5.5) at zero, a stabilizing control action is needed. Therefore, the following simple Lyapunov function is chosen:

$$V_1 = \frac{1}{2}\varepsilon_1^2, \tag{5.7}$$

with derivative:

$$\begin{aligned} \dot{V}_1 &= \varepsilon_1 \dot{\varepsilon}_1 \\ &= \varepsilon_1 (z_1 - \dot{y}_r). \end{aligned} \tag{5.8}$$

For error (5.5) to be stable at $\varepsilon_1 = 0$, (5.8) should be negative except for $\varepsilon_1 = 0$. To achieve this, z_1 is taken as a virtual input. With the following control strategy for this virtual input, ε_1 can be stabilized at 0:

$$\alpha_1 = \dot{y}_r - k_1\varepsilon_1 \qquad k_1 > 0, \tag{5.9}$$

since the derivative of the Lyapunov function (5.7) then becomes:

$$\dot{V}_1 = -k_1\varepsilon_1^2 \leq 0. \tag{5.10}$$

To develop a control strategy for the input u that ensures that the virtual input z_1 follows the virtual control strategy α_1, a second error variable is defined:

$$\varepsilon_2 = z_1 - \alpha_1. \tag{5.11}$$

Rewriting this for z_1 and substitution in (5.6) yields:

$$\begin{aligned} \dot{\varepsilon}_1 &= \varepsilon_2 + \alpha_1 - \dot{y}_r \\ &= \varepsilon_2 - k_1\varepsilon_1. \end{aligned} \tag{5.12}$$

Using the derivative for the outgoing flux:

$$\dot{z}_i = \frac{\mu\dot{x}_i}{(x_i+1)^2}, \tag{5.13}$$

the derivative for (5.11) can be written as:

$$\begin{aligned} \dot{\varepsilon}_2 &= \dot{z}_1 - \dot{\alpha}_1 \\ &= \frac{\mu\dot{x}_1}{(x_1+1)^2} - \ddot{y}_r + k_1\dot{\varepsilon}_1 \\ &= \frac{\mu}{(x_1+1)^2}(u - z_1) - \ddot{y}_r + k_1(\varepsilon_2 - k_1\varepsilon_1). \end{aligned} \tag{5.14}$$

For the stabilization of (5.12,5.14) the previously stated Lyapunov function (5.7) is extended to:

$$V_2 = \frac{1}{2}\varepsilon_1^2 + \frac{1}{2}\varepsilon_2^2, \tag{5.15}$$

with derivative:

$$\begin{aligned}\dot{V}_2 &= \varepsilon_1\dot{\varepsilon}_1 + \varepsilon_2\dot{\varepsilon}_2 \\ &= -k_1\varepsilon_1^2 + \varepsilon_2(\varepsilon_1 + \dot{z}_1 - \dot{\alpha}_1) \\ &= -k_1\varepsilon_1^2 + \varepsilon_2\left(\varepsilon_1 + \frac{\mu}{(x_1+1)^2}(u - z_1) - \ddot{y}_r + k_1(\varepsilon_2 - k_1\varepsilon_1)\right).\end{aligned} \tag{5.16}$$

For (5.16) to be negative, the following input u is chosen:

$$\begin{aligned}u &= z_1 + \frac{(x_1+1)^2}{\mu}\big(\ddot{y}_r + \varepsilon_1(k_1^2 - 1) - \varepsilon_2(k_1 + k_2)\big) \qquad\qquad k_2 > 0 \\ &= \underbrace{\frac{\overbrace{\mu x_1}^{\dot{y}}}{x_1+1} + \frac{\ddot{y}_r(x_1+1)^2}{\mu}}_{u_1} \underbrace{- \frac{(x_1+1)^2}{\mu}\left((k_1+k_2)\Big(\frac{\overbrace{\mu x_1}^{\dot{y}}}{x_1+1} - \dot{y}_r\Big) + (k_1k_2+1)(y - y_r)\right)}_{u_2},\end{aligned} \tag{5.17}$$

so that the derivative for the Lyapunov function (5.15) becomes:

$$\dot{V}_2 = -k_1\varepsilon_1^2 - k_2\varepsilon_2^2 \leq 0. \tag{5.18}$$

The resulting control strategy u stabilizes both errors ε_1 and ε_2 at zero, and therefore stabilizes the complete single workstation approximation model onto the reference trajectory y_r.

The control strategy derived above is a feedback controller that stabilizes the model for a single workstation manufacturing line on a reference trajectory. However, the manufacturing line by itself is stable. That is, the flux of lots into the system eventually becomes the flux of lots out of the system, provided that the influx is smaller than the average processing rate of the machines in the system ($u < \mu$). This suggests that a control action can be found for the manufacturing line, that is purely based on the reference trajectory and its derivatives.

The reference dynamics of the system is given by:

$$\begin{aligned}\dot{x}_{1r} &= u_r - \frac{\mu x_{1r}}{x_{1r}+1} \\ \dot{y}_r &= \frac{\mu x_{1r}}{x_{1r}+1},\end{aligned} \tag{5.19}$$

which describes the behavior of the manufacturing line when the output y and its derivatives exactly match the reference trajectory y_r and its derivatives. This means

that no feedback control action is needed to stabilize the system onto the reference trajectory.

Using (5.13), the following relation can be determined for $\ddot{y}_r$:

$$\ddot{y}_r = \frac{\mu}{(x_{1r}+1)^2}\Big(u_r - \frac{\mu x_{1r}}{x_{1r}+1}\Big). \tag{5.20}$$

Substituting this relation into (5.17) shows that the control strategy (5.17) indeed reduces to u_r once the output of the system is stabilized at y_r, as u_2 becomes 0 and:

$$u_1 = \frac{\mu x_{1r}}{x_{1r}+1} + \frac{\ddot{y}_r(x_{1r}+1)^2}{\mu} = u_r. \tag{5.21}$$

Now u_r can be written in terms of $\dot{y}_r$ and $\ddot{y}_r$ by solving the second equation in (5.19) for x_{1r}, which results in:

$$x_{1r} = \frac{\dot{y}_r}{\mu - \dot{y}_r}. \tag{5.22}$$

Substituting this into (5.21) then results in:

$$u_r = \dot{y}_r + \frac{\mu \ddot{y}_r}{(\mu - \dot{y}_r)^2}. \tag{5.23}$$

Since the manufacturing line by itself is stable, it should be possible to control the manufacturing line using (5.23) when the system initially matches the reference state imposed by the reference trajectory y_r and its derivatives. If this initial reference state is not matched however, u_r will allow the system to follow the requested changes in the outgoing flux $\dot{y}_r$ of the system, but with an offset (backlog) compared to y_r. This is shown in the following section, where the performance of both the control strategy u (5.17) and u_r (5.23) is discussed.

5.3 Performance of the control strategy for a single workstation manufacturing line

In this section the performance of the feedback control strategy (5.17) applied to the approximation model for a single workstation manufacturing system (5.4) is discussed.

The control strategy is tested using a case, where the demand $\dot{y}_r$ is subject to seasonal influences. The demand is assumed to follow a sine curve with an average throughput corresponding to a 47.5% utilization level for the manufacturing system, and minimum and maximum reaching respectively the 0% and 95% utilization levels. The seasonal

influences have a time period of 365 days and the machine in the production system has a mean processing rate μ of 10 lots a day. The manufacturing system is initially left empty while the demand at this point is maximal.

It is expected that since the initial conditions do not match the conditions that correspond to the demand at time $t = 0$, the system that is controlled solely by u_r (5.23) will develop a fixed offset (backlog) compared to the reference trajectory. For the complete control strategy (5.17) this will probably result in a temporal backlog because of its stabilizing effect.

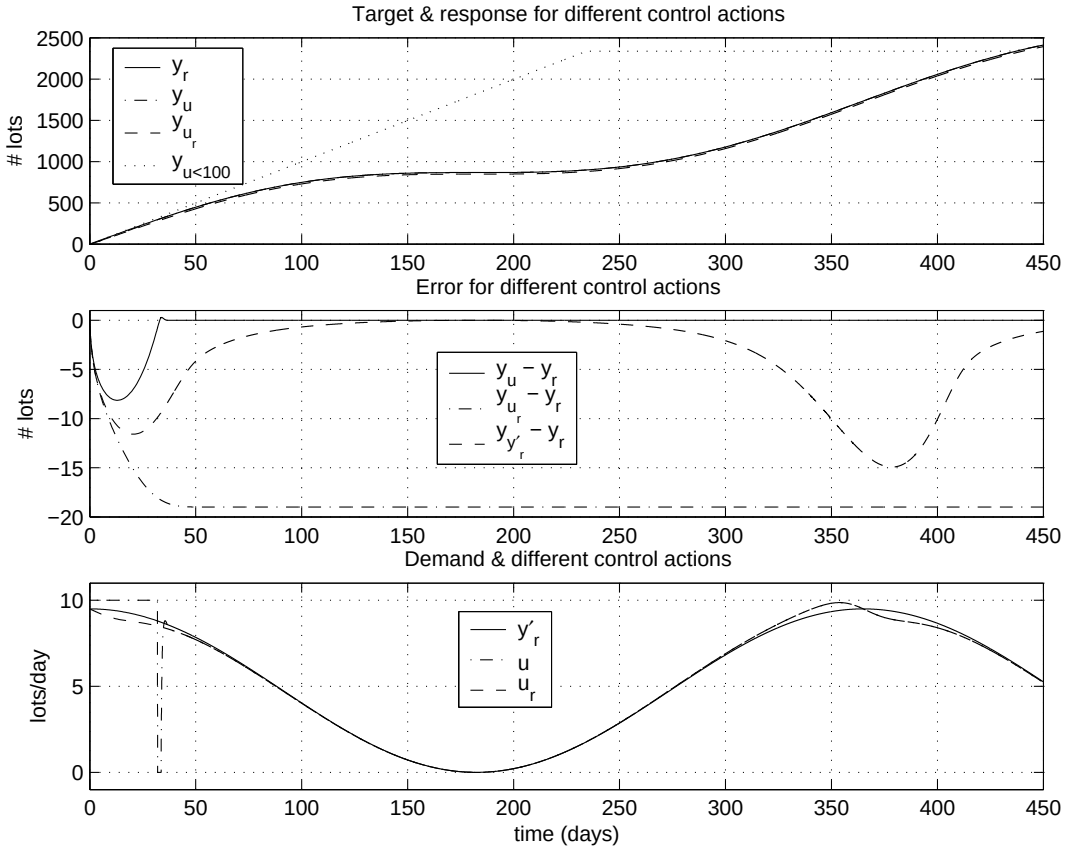


Figure 5.1: Response and control action for 1 workstation test case

Figure 5.1 shows the results for the test case, where the gains k_1 and k_2 in (5.17) are set to 1. As expected the control strategy (u_r) proves to be able to control the system, apart from some amount of backlog that is formed since the initial state of the system does not match the initial reference state. This initial condition problem can be seen as a discontinuity in the demand. Since the system is initially empty, the throughput of the system is at that point zero, while the demand $\dot{y}_r$ at that moment is actually 9.5 lots a day. In the time span it takes the system with the given influx u_r to reach

a state that matches the reference state at that time, a shortage of 19 lots is formed. This effect can be clearly seen in the second plot of Figure 5.1 ($y_{u_r} - y_r$).

When the feedback control action (5.17) is applied for the test case however, some problems occur with the feasibility regarding the obtained controller actions. The effects of the discontinuity in demand cause the controller to apply a very high rate at which lots are released into the system. This results in a throughput for the system that is higher than the demand, to make up for the backlog. In fact, the response of the control action is so strong that Matlab is not able to solve the ODE model for this test case, even when the gains are set to zero. This problem is solved by providing an upper limit for the control signal.

For high limits however, this still causes problems, as can be seen in the first plot of Figure 5.1, where an upper limit of 100 lots an hour is applied ($y_{u<100}$). The problem is that in order to make up for the backlog, the controller raises the throughput of the system to a level higher than the demand by increasing the number of lots in the system. The controller subsequently tries to remove the excess of lots by applying a negative inflow, to match the throughput of the system to the actual demand. However, lots are not allowed to leave the manufacturing line at the entrance of the system as this behavior does not match the behavior of a real life manufacturing line. By applying this constraint to the allowed flux into the system, the controller does not have the means to remove the excess of lots. This results in a positive backlog which, depending on the upper limit for the control action, might not be resolved within a reasonable period of time as simply too many lots are released into the system.

For the test case an upper limit for the control signal is used that corresponds to the average processing rate of a workstation. Figure 5.1 shows that the saturated controller action u is able to deal with the discontinuity at $t = 0$ and compensate for the resulting backlog. However, some time is needed to reduce the amount of backlog to zero. This time could be reduced by using a different upper limit for the rate at which lots are released into the system. Finding an optimum for the maximal release rate is not easy as it is actually the WIP of the system that has to be limited. This means that for different discontinuities in the demand, different upper limits are needed to reduce the amount of backlog to zero in the shortest time possible. Therefore, it might be better to use a constraint based on the amount of WIP in the system.

5.4 Controller performance for a real life single workstation manufacturing line

In the previous section the performance of the control strategy developed for the single workstation approximation model was discussed. To this end the control strategy was applied to the continuous approximation model. In real life however, manufacturing systems suffer from variability. This means that the processing rate of a workstation varies in time. It is therefore interesting to test the performance of the control strategy

for systems that are subject to variability. In this section the control strategy is therefore linked directly to a discrete event model (DEM) that represents a manufacturing line where the processing rate of the machine is exponentially distributed with mean μ (Appendix A.2).

The performance is tested using the same test case as used in the previous section. It is expected that the control strategy (5.17) will be able to stabilize the output of the DEM onto the reference trajectory. Because of the variability present in the system the error will probably not be exactly 0, but vary around 0. Because the DEM only works with whole numbers of lots, the resulting figure will show discontinuities in the results.

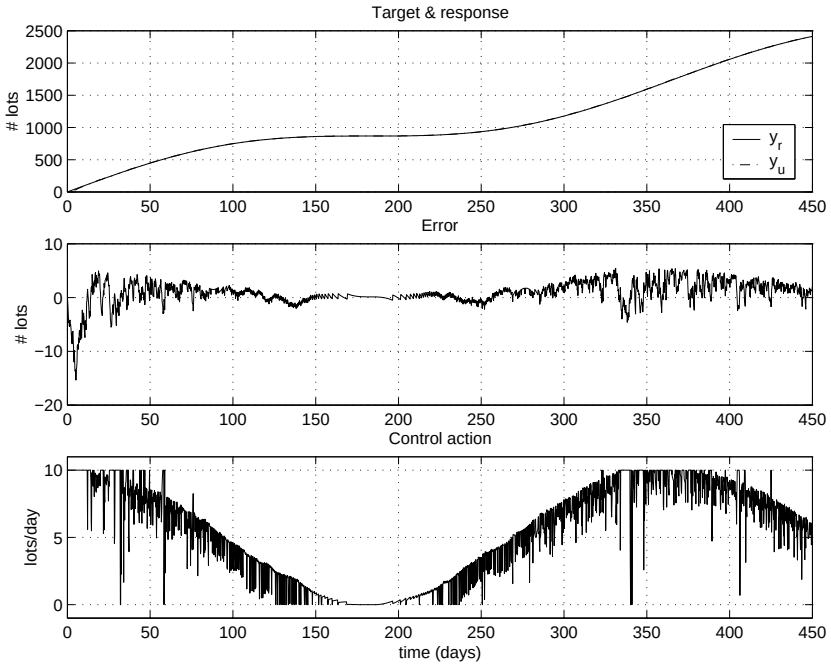


Figure 5.2: Response of DEM for 1 workstation test case

The results for a single simulation run are shown in Figure 5.2. It can be seen that the controller is indeed capable of stabilizing the system around the reference trajectory y_r. When the error $(y_u - y_r)$ is observed more closely in the second plot however, it can be seen that the error is not exactly zero. The stochastic behavior of the DEM results in an error that varies around zero. This stochastic behavior can also be observed in the control signal u since u is derived directly from the state of the DEM.

Because of the stochastic behavior of the DEM every single simulation will show different results. Therefore, several simulations are performed to study the overall effect of the control strategy. As each simulation is performed independently, the results of the individual simulations can be analyzed using basic statistics (Appendix B).

The averaged results for the multiple simulation runs are shown in Figure 5.3. Surpris-

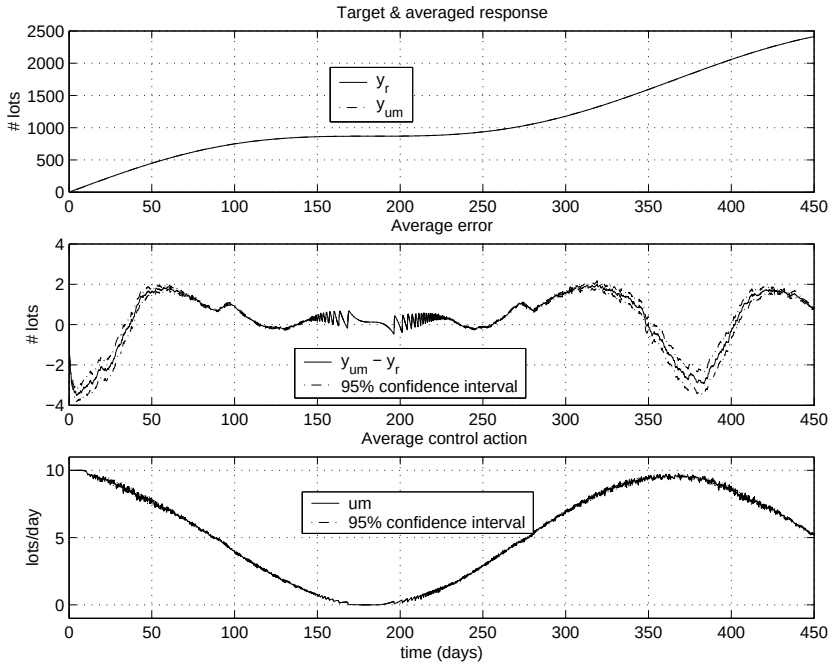


Figure 5.3: Averaged response of DEM for 1 workstation test case

ingly the averaged error $y_{um} - y_r$ shows some trends, which are not observed in the single experiment shown in Figure 5.2. For periods with high utilization some negative backlog seems to be formed, while for medium utilization levels the backlog is on average positive. It should also be noted that for periods of high utilization the confidence interval on the mean error is relatively large. This matches the idea that variability is especially important for high utilization.

In the following section the control strategy for the single workstation manufacturing line is expanded to control a manufacturing line that consists of two workstations.

5.5 Controller design for a two workstation manufacturing line

Now that a control strategy for a single workstation manufacturing system is designed and tested, the backstepping technique is continued to derive a control strategy for a continuous approximation model of a two workstation manufacturing line:

$$\begin{aligned} \dot{x}_2 &= u - z_2 \\ \dot{x}_1 &= z_2 - z_1 \\ \dot{y} &= z_1. \end{aligned} \tag{5.24}$$

Since the two workstation model is actually a single workstation model with an extra workstation added, the control strategy discussed in Section 5.2 still applies to a part of the two workstation model. The only difference is that the stabilizing input for the single workstation model is now a control strategy for the virtual input z_2, the flow of lots between the two workstations:

$$\alpha_2 = z_1 + \frac{(x_1+1)^2}{\mu}(\dot{\alpha}_1 - \varepsilon_1 - k_2\varepsilon_2) \qquad k_2 > 0. \tag{5.25}$$

To develop a control strategy for the input u that ensures that the virtual input z_2 follows the virtual control strategy α_2, a third error variable is defined:

$$\varepsilon_3 = z_2 - \alpha_2. \tag{5.26}$$

Rewriting this for z_2 and substitution into (5.14) yields the following relation for the derivative of the second error variable:

$$\begin{aligned} \dot{\varepsilon}_2 &= \frac{\mu}{(x_1+1)^2}(z_2 - z_1) - \dot{\alpha}_1 \\ &= \frac{\mu}{(x_1+1)^2}(\varepsilon_3 + \alpha_2 - z_1) - \dot{\alpha}_1 \\ &= \frac{\mu}{(x_1+1)^2}\left(\varepsilon_3 + \frac{(x_1+1)^2}{\mu}(\dot{\alpha}_1 - \varepsilon_1 - k_2\varepsilon_2)\right) - \dot{\alpha}_1 \\ &= -\varepsilon_1 - k_2\varepsilon_2 + \frac{\mu}{(x_1+1)^2}\varepsilon_3. \end{aligned} \tag{5.27}$$

The derivative for the new error variable defined in (5.26) can be written as:

$$\begin{aligned} \dot{\varepsilon}_3 &= \dot{z}_2 - \dot{\alpha}_2 \\ &= \frac{\mu}{(x_2+1)^2}(u - z_2) - \dot{\alpha}_2. \end{aligned} \tag{5.28}$$

For stabilizing (5.26) at zero, the Lyapunov function from (5.15) is extended to:

$$V_3 = \frac{1}{2}\varepsilon_1^2 + \frac{1}{2}\varepsilon_2^2 + \frac{1}{2}\varepsilon_3^2, \tag{5.29}$$

with derivative:

$$\begin{aligned}\dot{V}_3 &= \varepsilon_1\dot{\varepsilon}_1 + \varepsilon_2\dot{\varepsilon}_2 + \varepsilon_3\dot{\varepsilon}_3 \\ &= -k_1\varepsilon_1^2 - k_2\varepsilon_2^2 + \varepsilon_3\left(\frac{\mu}{(x_1+1)^2}\varepsilon_2 + \frac{\mu}{(x_2+1)^2}(u-z_2) - \dot{\alpha}_2\right).\end{aligned} \tag{5.30}$$

Now the output of the two workstation model can be stabilized onto the reference trajectory y_r by choosing the following control strategy for the release rate of lots into system:

$$\begin{aligned}u &= z_2 - \frac{(x_2+1)^2}{(x_1+1)^2}\varepsilon_2 + \frac{(x_2+1)^2}{\mu}(\dot{\alpha}_2 - k_3\varepsilon_3) \qquad k_3 > 0 \\ &= \frac{x_2\mu}{x_2+1} + \left(\frac{(x_2+1)^2}{(x_1+1)^2} + \frac{2(x_1+1)(x_2+1)^2}{\mu^2}\ddot{y}_r\right)\left(\frac{x_2\mu}{x_2+1} - \frac{x_1\mu}{x_1+1}\right) + \frac{(x_1+1)^2(x_2+1)^2}{\mu^2}\dddot{y}_r \Bigg\} u_1 \\ &\quad - \Bigg(\underbrace{\frac{\mu}{(x_1+1)^2}\left(\frac{x_2\mu}{x_2+1} - \frac{x_1\mu}{x_1+1}\right)}_{\ddot{y}} - \ddot{y}_r\Bigg)\frac{(x_1+1)^2(x_2+1)^2}{\mu^2}(k_1+k_2+k_3) \\ &\quad - \Bigg(\underbrace{\frac{x_1\mu}{x_1+1}}_{\dot{y}} - \dot{y}_r\Bigg)\left(\frac{(x_2+1)^2}{(x_1+1)^2} + \frac{(x_1+1)(x_2+1)^2}{\mu^2}\left(\frac{x_2\mu}{x_2+1} - \frac{x_1\mu}{x_1+1}\right)2(k_1+k_2)\right. \\ &\quad \left. + \frac{(x_1+1)^2(x_2+1)^2}{\mu^2}(k_1k_2 + k_2k_3 + k_1k_3 + 1)\right) \\ &\quad - (y - y_r)\left(\frac{(x_2+1)^2}{(x_1+1)^2}k_1 + \frac{2(x_1+1)(x_2+1)^2}{\mu^2}\left(\frac{x_2\mu}{x_2+1} - \frac{x_1\mu}{x_1+1}\right)(k_1k_2+1)\right. \\ &\quad \left. + \frac{(x_1+1)^2(x_2+1)^2}{\mu^2}(k_3 + k_1k_2k_3)\right), \qquad \Bigg\} u_2\end{aligned} \tag{5.31}$$

which results in the following derivative for the Lyapunov function (5.29):

$$\dot{V}_3 = -k_1\varepsilon_1^2 - k_2\varepsilon_2^2 - k_3\varepsilon_3^2 \leq 0. \tag{5.32}$$

Similar to the single workstation control strategy it can be shown that the system can be controlled by a strategy purely based on the reference trajectory and its derivatives provided that the system initially matches the reference state. This control strategy is given by:

$$u_r = \dot{y}_r + \frac{\mu\ddot{y}_r}{(\mu-\dot{y}_r)^2} + \frac{\frac{\mu^2\dddot{y}_r}{(\mu-\dot{y}_r)^2} + \frac{2\mu^2\ddot{y}_r^2}{(\mu-\dot{y}_r)^3} + \mu\ddot{y}_r}{\left(\mu - \frac{\mu\ddot{y}_r}{(\mu-\dot{y}_r)^2} - \dot{y}_r\right)^2}. \tag{5.33}$$

The performance of both these control actions is discussed in the following section.

5.6 Performance of control strategy for two workstation manufacturing line

The performance of the control strategy for the two workstation approximation model (5.31) is analyzed using the same test case as used to analyze the single workstation control strategy (Section 5.3). Because of the longer manufacturing line, the system will react slower to changes in the influx. This means that the controller will probably apply some stronger control actions to compensate for this extra delay, especially in the regions with high utilization levels.

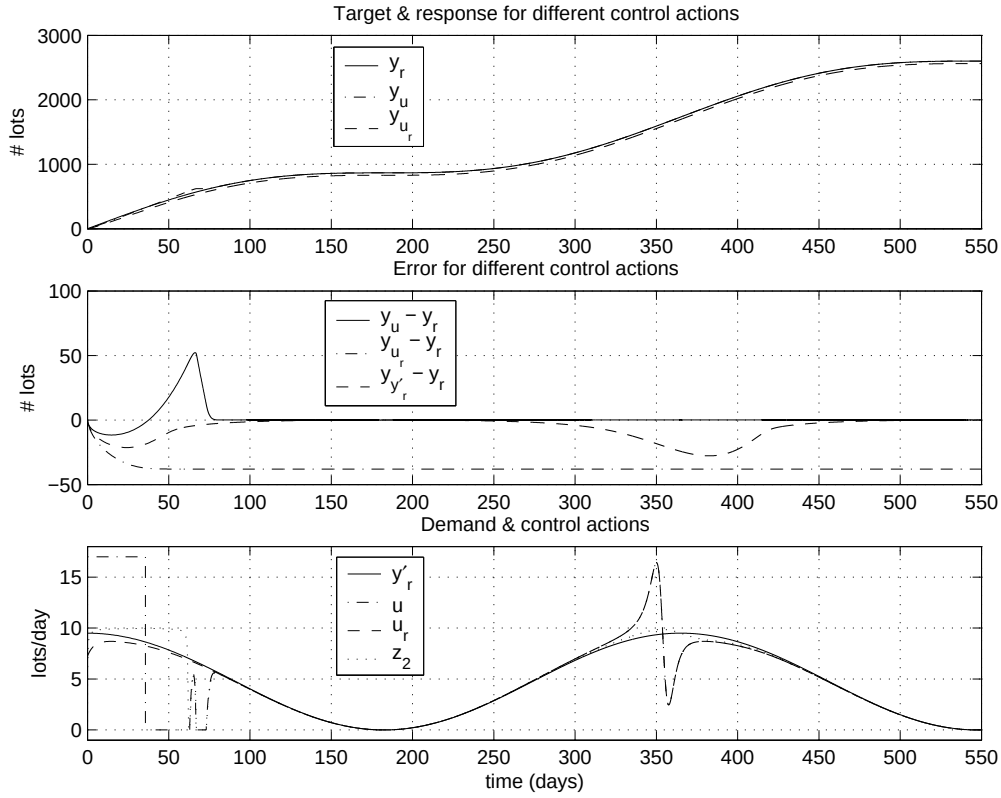


Figure 5.4: Response and control action for 2 workstation test case

Figure 5.4 shows the results for the test case. It can be seen in the second plot of the figure that u_r is again able to match the throughput of the manufacturing system to the demand $\dot{y}_r$. The amount of backlog that is formed in this case however, is higher than for the single workstation model. This can simply be explained by the fact that

the response of this system is slower than the single workstation model. In fact, the resulting backlog is exactly 38 lots, twice as many as for the single workstation model.

Looking at the resulting control signal u in the third plot of Figure 5.4, it should be noted that the upper limit for the release rate of lots into the system is chosen higher than the mean processing rate μ, because of the stronger controller actions near regions with high utilization levels. This is done to prevent the temporal formation of backlog in these high utilization regions, as the control signal in these regions would be capped off at 10 lots an hour. Apart from the resulting control signal u, the third plot of Figure 5.4 also shows the flux z_2 of lots from the first workstation in the line to the next workstation. This is the flux that was actually controlled for the single workstation model. Comparing this flux to the control signal needed for the 2 workstation model shows the effect of adding another workstation to the manufacturing line on the control actions. Since the system reacts slower, the controller for this longer manufacturing line needs to compensate for the extra delay. Looking at Figure 5.4 around about 350 days into the test case, it can be seen that the control signal u is already compensating for the expected change in direction of the demand, while z_2 follows the control strategy for the single workstation model, and reacts later. Apart from this compensation for the delay it can also be seen that the control action is stronger compared to the controller for the single workstation model, to compensate for the damping effect of a longer system.

Another effect that can be seen in the first and second plot of Figure 5.4, is that since the controller is allowed to release lots into the system at a higher rate compared to the single workstation controller, temporarily a positive backlog is formed. This is a result of the chosen upper limit for the flux of lots into the system, as discussed in Section 5.3.

5.7 Control of a two workstation DEM

In this section the effect of variability on the performance of the control strategy for the two workstation manufacturing line (5.31) is discussed. To study the effect of variability, the control strategy is again applied directly to a DEM that represents the behavior of a two workstation manufacturing line (Appendix A.2) with exponentially distributed processing rates for the machines. To compare the results for the discrete simulation to the results for the continuous approximation model, the performance of the control strategy for the DEM is tested using the same test case (Section 5.3) as used for the continuous model.

Apart from the stochastic nature of the output of the system, it is expected that the results for the test case will generally show the behavior seen in the previous section. That is, the system may initially show some overshoot because of the raised upper limit for the control signal, but this positive backlog will probably be of temporal nature and stabilize around 0.

The results are shown in Figure 5.5. It can be seen that, as expected, the output of the system suffers from overshoot caused by the high upper limit for the influx. Apart

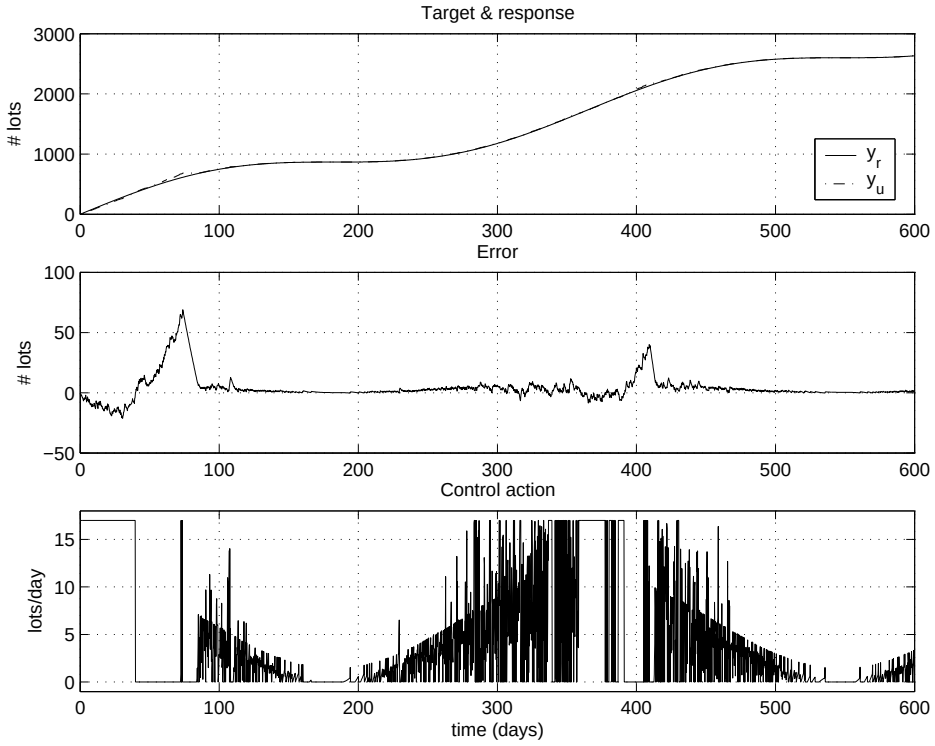


Figure 5.5: Response of DEM for 2 workstation test case

from the overshoot at the beginning of the simulation however, the error plot shows that the overshoot also occurs in the second region with high utilization. To check if this behavior is incidental, the simulation is repeated several times so that the general behavior of the controlled system can be observed. The results for these simulations are shown in Figure 5.6.

Figure 5.6 indeed shows that for several independent simulations the same effects occur. After a period of high utilization, generally a positive backlog is formed and the control signal shows a sharp descent. This indicates that too many lots are released into the system in the regions with relatively high utilization and that the control strategy tries to compensate for the positive backlog by reducing the influx into the system. It seems that the high upper limit for the control signal combined with the stochastic behavior of the system reduce the performance of the control strategy, as the trend shown in the second plot of Figure 5.6 suggests that the amount of backlog increases with every period of high utilization. To test if the performance can be improved by reducing the upper limit for the control signal, simulations are performed where the upper limit is lowered to a release rate of lots that corresponds to the average processing rate μ.

The results are shown in Figure 5.7. It can be seen in this figure that the amount of backlog does not grow after a period of high utilization. Clearly the performance of the

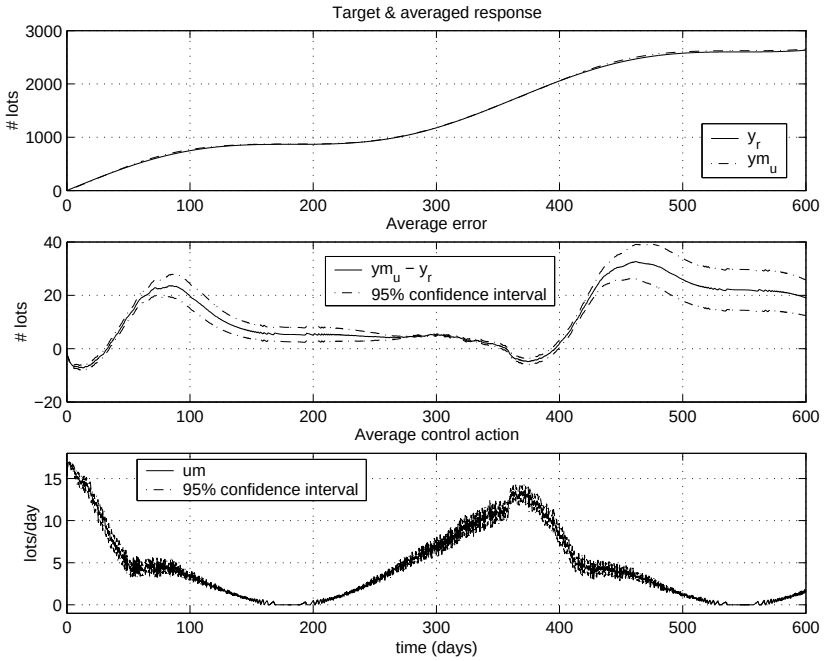


Figure 5.6: Averaged response of DEM for 2 workstation test case

control strategy is improved by reducing the upper limit for the control signal. However, the remaining backlog does show some trends similar to those observed for the single workstation system (Section 5.4).

Now that a control strategy is derived for a two workstation manufacturing line, this strategy can be extended to allow for the control of longer manufacturing lines. This is discussed in the following section.

5.8 Controller design for longer manufacturing lines

In the previous sections the backstepping approach was applied to derive control strategies for continuous approximation models consisting of one or two workstations. For models that consist of more than two workstations, the backstepping procedure can be continued.

Each step where another workstation is added to the model, is similar to the step from a single workstation to a two workstation model as discussed in Section 5.5. The following building blocks therefore, allow the design of a control strategy for models of arbitrary length:

$$\alpha_1 = \dot{y}_r - k_1\varepsilon_1 \qquad k_1 > 0, \tag{5.34}$$

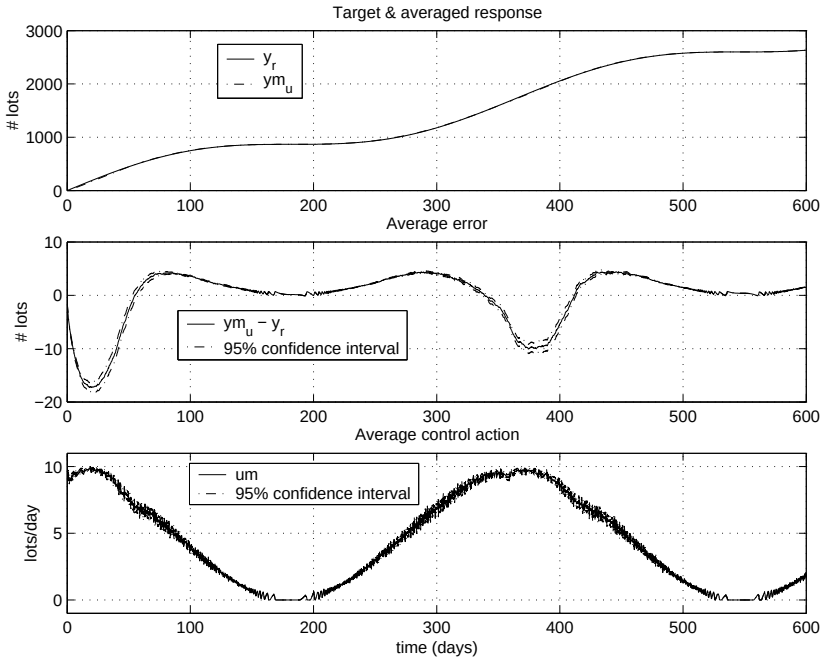


Figure 5.7: Response of DEM for 2 workstation test case, where the input signal is limited to the average processing rate μ

$$\varepsilon_1 = y - y_r, \tag{5.35}$$

$$\alpha_2 = z_1 + \frac{(x_1+1)^2}{\mu}(\dot{\alpha}_1 - \varepsilon_1 - k_2\varepsilon_2) \qquad k_2 > 0, \tag{5.36}$$

$$\varepsilon_i = z_{i-1} - \alpha_{i-1} \qquad i > 1, \tag{5.37}$$

$$\alpha_i = z_{i-1} - \frac{(x_{i-1}+1)^2}{(x_{i-2}+1)^2}\varepsilon_{i-1} + \frac{(x_{i-1}+1)^2}{\mu}(\dot{\alpha}_{i-1} - k_i\varepsilon_i) \qquad k_i > 0, i > 2. \tag{5.38}$$

Though the structure seems relatively simple, the reference to the derivative of the control strategy for order $i-1$ causes the number of terms necessary to fully describe the complete control sequence to grow explosively. For each step in the backstepping procedure all the control strategies for the virtual inputs and derivatives obtained so far have to be differentiated again, resulting in this 'explosion of terms'. This effect could already be observed in Section 5.5, where the step from a single workstation model

(5.17) to a two workstation model control strategy (5.31) showed a strong increase in terms that make up the control strategy.

In this chapter a control strategy was derived for the Godunov based approximation model for the manufacturing system as discussed in section 3.2.2. It was shown that the derived strategy allowed a single and two workstation model to follow a predefined trajectory for the demand, compensating for permanent backlog. The performance of the control strategy in situations where the throughput of the system does not match the demand, seemed to depend on the applied saturation level for the maximal allowed flux of lots into the system. That is, in a ramp up situation, the controller might be able to stabilize the throughput of the system on the demand faster if a higher upper limit is used for the release rate of lots into the system. However, the limit should not be raised to high as this can result in the formation of positive backlog. It was also shown that the structure of the control strategy results in the 'explosion' of the number of terms that make up the control strategy for longer manufacturing lines. This is a well known problem related to the use of a backstepping procedure to derive a control strategy. When the derived control strategies where applied directly to DEM which represent the stochastic behavior of the observed manufacturing line, the performance of the control strategies decreased. This can partially be explained by the fact that it is impossible to correct for the variability present in the system, but the averaged results of multiple simulations showed that still some trends can be observed in the error for the controlled DEM.

Chapter 6

Conclusions

In this thesis it was investigated if the computational effort needed to perform simulations for a partial differential equation (PDE) model of a manufacturing line could be reduced by applying a reduction technique based on the method of proper orthogonal decomposition (POD). It was shown that for a PDE model that is discretized into 100 grid cells for the spatial domain, using roughly a tenth of this number for the reduced set of POD basis functions still resulted in acceptably accurate simulation results. However, the effective reduction in computation time is of a smaller order.
This reduction in computation time is of a far less magnitude than reductions realized for a computational fluid dynamics (CFD) model of glass melt tank furnace ($\mathcal{O}10^5$) as discussed in [Hui05]. The spatial domain for this model contains a number of grid cells in the order $\mathcal{O}10^5$ which can be reduced to a set of 10 to 20 POD basis functions while still obtaining sufficiently accurate results.
It is however expected that the number of POD basis functions used to describe the spatial dynamics of the PDE model used in this thesis, might also describe the spatial dynamics of such a model where the spatial domain is discretized into more than the 100 grid cells as used here, with similar accuracy. That is, if the characteristic spatial dynamics do not change much for a higher number of grid cells, this characteristic behavior will still be accurately described by the same number of POD basis functions.

In section 3.2.2 a different implementation for the model of the manufacturing system was derived, using a first order Godunov method resulting in a model based on a set of ODEs. This model is of a simple form where every single ODE represents the behavior of one workstation within the manufacturing line. Control strategies were derived for these type of models using a backstepping approach, where the systems consisted of one or two workstations. When saturated versions of these control strategies were applied to the corresponding approximation models, it was shown that the strategies were able to stabilize the throughput of the system onto the demand and correct for backlog. The performance of the control strategies however, seemed to depend on the allowed upper limit for the control signal. That is, a higher upper limit allows for a faster correction if the actual throughput of the system is below the current demand, but if too many

lots are released into the system, the resulting overshoot can not be compensated for by applying a negative influx as such an input signal is not allowed. This suggests that the performance of the control strategy can be improved by tuning the maximal number of lots that are allowed to be released into the system for particular situations. Better results might be obtained by using a control strategy that does not allow an excess of lots to be released into the system.

When the control strategies developed for the continuous approximation models are applied directly to the discrete event models (DEM) which include variability, the performance of the control strategies decreases. This can partially be explained by the fact that the variability in the system can not be compensated for as the variability can not be predicted. However, averaged results for different simulations show some trends in the error, which might be eliminated using a different control strategy. Further research needs to be done to determine the cause for these trends and to find a control strategy that eliminates these trends.

Another problem regarding the control strategies derived using a backstepping approach, is that for longer manufacturing lines the number of terms that make up the control strategy grows explosively. This is a common problem associated with the backstepping approach. It could therefore be interesting to investigate the application of a backstepping procedure to obtain a control strategy for a reduced order model of a manufacturing system, as this could drastically reduce the number of terms needed for the derived control strategy for the observed system.

Chapter 7

Recommendations

In this thesis a basic PDE model that can be used to simulate the behavior of a manufacturing line was used to illustrate model reduction based on the method of Proper Orthogonal Decomposition (POD) and the development of a control strategy using a backstepping approach. Previous research however, suggested that this basic model is not able to accurately describe the transient behavior of manufacturing lines [Ber04]. This problem regarding the accuracy of the transient state is also observed in Chapter 4. Previous research also suggested some adaptations to the PDE model, improving the accuracy of the approximation of the transient state of a manufacturing line [Pla04]. It might therefore be interesting to apply the POD reduction technique to these improved PDE models. For the improved PDE models however, some parameters are unknown. Therefore the reduction technique should be applied in combination with system identification (SID) techniques to identify the unknown parameters. Apart from the effect on the accuracy of the transient behavior described by these improved models, it should also be investigated if the use of SID techniques allows the models to describe the behavior of more realistic manufacturing systems. For example systems where individual workstations have different processing rates, or systems where the assumption of exponentially distributed processing times do not hold.

The model for the manufacturing system, derived using a first order Godunov method in Chapter 3, consists of a set of ODEs where every single ODE corresponds to a single workstation within the manufacturing line. This suggests that using this model, it is possible to describe the behavior of a manufacturing line where the individual workstations have different average processing rates. As the combination of different types of workstations allows for the modelling of more realistic manufacturing lines, it is interesting to investigate the effect that these different types of workstations have on the developed control strategy and how the performance of these control strategies is affected.

The number of terms that make up the control strategy derived for the Godunov based model of the manufacturing system, was shown to increase dramatically for longer

manufacturing lines. This explosion of terms is a common problem for control strategies derived using a backstepping approach. It is therefore interesting to investigate the use of different control methods for sets of nonlinear ODEs, which do not result in an excessive number of term that make up the control strategy. It might for example be possible to apply the backstepping method to a model which is reduced using the POD reduction method. As this reduction method can effectively reduce the number of ODEs that make up the approximation model, a drastic reduction in the terms that make up the control strategy for a model of a manufacturing line might be realized. In fact, when the derived control strategy is robust enough, the number of POD basis functions might be reduced even further while still obtaining similar performance for the controlled system. It should therefore be investigated how the backstepping approach can be applied to reduced order models, and what the effect is on the complexity and the performance of the developed controller.

In Chapter 5 it was suggested that a control strategy could be derived that allows the system to stabilize on the reference trajectory more quickly, without resulting in a large overshoot. For this improvement in performance, a different approach should be taken to limit the number of lots released into the system by the controller. It is therefore interesting to investigate the performance of control strategies that use different methods to limit the released number of lots into the system. This might for example be realized using a control strategy that stabilizes the complete state of the manufacturing line on a desired reference state.

The manufacturing system observed in this thesis consists of 100 workstations. It was shown that the number of basis functions used to describe the spatial dynamics of the system could roughly be reduced by a factor 10 while still obtaining sufficiently accurate results. In the conclusions it was mentioned that this reduced number of POD basis functions could be the same number of basis functions used to describe the spatial dynamics of a more fine discretized system with similar accuracy. This suggests that the spatial dynamics of a model for a manufacturing line consisting of 1000 workstations might also be described with a number of basis functions similar to the number used for the 100 workstation manufacturing line. After all, the longer manufacturing line will show characteristic behavior in a ramp up situation similar to the behavior of a shorter manufacturing line, the largest difference is the scale of this dynamical behavior. This means that the POD reduction method will probably be more efficient for longer manufacturing lines. This should be tested by applying the POD reduction method to PDE models for manufacturing lines that consist of different numbers of workstations and comparing the obtained reduction levels for similar levels of accuracy.

Bibliography

[Arm02] H.D. Armbruster, D. Marthaler, and C. Ringhofer. Efficient simulations of supply chain. In *Proceedings of the Winter Simulation Conference*, pages 1345–1348, 2002.

[Ast04] P. Astrid. *Reduction of Process Simulation Models: a proper orthogonal decomposition approach*. PhD thesis, Eindhoven University of Technology, 2004.

[Ber04] R.A. van den Berg. Partial differential equations in modelling and control of manufacturing systems. Master's thesis, Eindhoven University of Technology, Department of Mechanical Engineering, Systems Engineering Group, March 2004.

[Dav84] P.J. Davis and P. Rabinowitz. *Methods of numerical integration*. London : Academic Press, second edition, 1984.

[Hui05] L. Huisman. *Control of Glass Melting Processes Based on Reduced CFD Models*. PhD thesis, Eindhoven University of Technology, 2005.

[Law00] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Singapore, third edition, 2000.

[Pla04] S. Platschorre. Modelling of manufacturing lines using higher order PDEs. Master's thesis, Eindhoven University of Technology, Department of Mechanical Engineering, Systems Engineering Group, December 2004.

[Sas99] S. Sastry. *Nonlinear Systems*. Springer-Verlag New York, Inc, first edition, 1999.

[Veq02] R.J. Le Veque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, first edition, 2002.

[Wei] E.W. Weisstein. Newton-Cotes Formulas. MathWorld, a Wolfram Web Resource. http://mathworld.wolfram.com/Newton-CotesFormulas.html.

Appendix A

Discrete event models

In this appendix the discrete event models (DEM) are discussed that are used in this research. The first section discusses the model that is used to validate the PDE based models and measure density distributions to obtain POD basis functions derived from measured data as discussed in section 4.4. The second section shows the model used to test the performance of the developed control strategies for discrete systems that suffer from variability as discussed in section 5.4 and 5.7 together with the Python script and Matlab code used to implement the control strategies. Both discrete event models are written in χ0.8.

A.1 Model used to measure density distributions

This section shows the DEM used to obtain measurements on the density distribution within a manufacturing line that consists of 100 identical workstations. The measurements are used for the calculation of the verification data for the performance of the PDE based models (section 4.1) and for the calculation of POD basis functions as discussed in section 4.4.

```
from random import *
from std    import *
from fileio import *

type lot  = real # nat
type info = nat # real

proc G(a: !lot, li, ti, lt: real)=
|[ i: nat, tai, tat: ->real
 | i:= 1
```

```
 ; tai:= exponential(1.0/li)
 ; tat:= exponential(1.0/lt)
 ; *[ time < ti -> delta sample tai; a!<time,i>; i:= i+1 ]
 ; *[ true      -> delta sample tat; a!<time,i>; i:= i+1 ]
]|

proc B(a: ?lot, b: !lot, c: !info, d: ?void, e: ?nat, m: ?real) =
|[ x: lot, xs: lot*, bi, mb: nat, ct: real
 | xs:= []; mb:=0; ct:= 0.0; e?bi
 ; *[ len(xs) < bi -> xs:= xs ++ [<0.0,0>] ]
 ; *[ true       ; a?x      -> xs:= xs ++ [x]
    | len(xs) > 0; b!hd(xs) -> xs:= tl(xs); mb:= 1
    | true       ; d?       -> c!(<len(xs)+mb,ct>)
    | true       ; m?ct     -> mb:= 0
    ]
]|

proc M(a: ?lot, b: !lot, m: !real, mu: real) =
|[ x: lot, td: ->real
 | td:= exponential(1.0/mu)
 ; *[ true -> a?x; delta sample td; m!(time-x.0); x.0:= time; b!x]
]|

proc E(a: ?lot) =
|[ x:lot
 | *[ true -> a?x ]
]|

proc S(a: (?info)^100, b: (!void)^100, c: (!nat)^100, d: ?file, ts: real) =
|[ m: info, i: nat, cts: real*, wips: nat*, bi: nat^100
 | d?bi; i:=0; *[ i < 100 -> c.i!bi.i; i:= i + 1]
 ; *[ true -> delta ts; i:= 0; cts:= []; wips:= []
    ; *[ i < 100 -> b.i!; a.i?m; cts:= cts ++ [m.1]
                  ; wips:= wips ++ [m.0]; i:= i + 1 ]
    ; ! time, nl(), cts, nl(), wips, nl()
    ]
]|

clus Sys(li,ti,lt,mu,ts: real)=
|[ a: (-lot)^101, b: (-lot)^100, c: (-info)^100
 , d: (-void)^100, e: (-nat)^100, m: (-real)^100
 | G(a.0,li,ti,lt)
|| i: nat <- 0..100: B(a.i,b.i,c.i,d.i,e.i,m.i)
|| i: nat <- 0..100: M(b.i,a.(i+1),m.i,mu)
```

```
|| S(c,d,e, filein("ibuf.txt"),ts)
|| E(a.100)
]|

xper(li,ti,lt,mu,ts: real) = |[ Sys( li, ti, lt, mu, ts ) ]|
```

The inputs for the simulation model are the initial arrival rate $li(\lambda_i[\text{lots/hour}])$, the target arrival rate $lt(\lambda_t[\text{lots/hour}])$ and the time $ti(t_i[\text{hour}])$ when the arrival rate changes from li to lt. This delay in change of setpoint is used to allow the DEM to reach an initial steady state before the setpoint is changed. The average $mu(\mu[\text{lots/hour}])$ for the exponentially distributed processing rate of the individual machines and the sampling time $ts(t_s[\text{hour}])$ also need to be specified. The structure of the model is shown in Figure A.1.

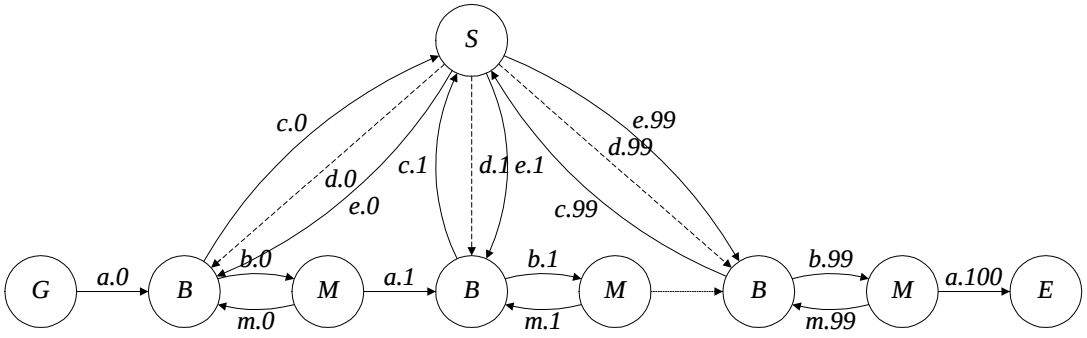


Figure A.1: Structure of DEM

The generator G releases lots into the system at an exponentially distributed rate. Lots are initially generated at a mean rate of λ_i and when the time reaches t_i the mean rate is changed to λ_t. The time that the release rate of the lots is kept at the initial rate is used to let the production system reach the initial steady state before the setpoint is changed.

Each workstation in the production system consists of a buffer B and a machine M. The buffer is of infinite capacity and also contains functionality to collect information on the state of the workstation. The buffer can also be initiated to a specified level that corresponds with an initial arrival rate. When information on the state of the workstation at a particular point in time is requested, the buffer returns the number of lots present in the workstation and the flow time of the last lot that left the workstation. It therefore needs to keep track of the state of the machine. When a lot is being processed on the machine it is still located in the observed workstation. Only when processing is finished it no longer accounts for the work in process on the workstation. At this time the machine returns the flow time of the lot to the buffer and passes the lot to the next workstation. To keep track of the flow time, the machine passes the lot to the

next workstation together with the time that the lot left the current machine. The next machine can then determine the flow time by comparing the time that the lot leaves this next machine to the time that the lot left the current machine.

To collect information on the different workstations at specific intervals in time, a separate process S is used. This process is also used to pass the initial buffer levels to the buffers. The initial buffer levels are specified in a separate file *'ibuf.txt'* which contains an array with the initial values. Samples on the WIP and flow time of the individual workstations are collected every t_s hour and are printed to screen together with the current time.

A.2 Model used to analyze the performance of the developed control strategies

In this section the model is shown that is used to analyze the performance of the control strategies developed in Chapter 5. Together with the Python script and Matlab code to implement the control strategies. The DEM is shown below:

```
from random   import *
from std      import *
from control1 import *

type lot  = real # nat
type info = nat # real

proc G(a: !lot, b: ?real)=
|[ i: nat, lt, ltnew, t: real
 | i:= 1
 ; b?lt; t:=1/lt+time
 ; *[ true; delta t-time -> a!<time,i>; i:= i+1; t:=1/lt+time
    | true; b?ltnew      -> t:=(t-time)*lt/ltnew+time; lt:=ltnew
    ]
]|

proc B(a: ?lot, b: !lot, c: !info, d: ?void, e: ?nat, m: ?real) =
|[ x: lot, xs: lot*, bi, mb: nat, ct: real
 | xs:= []; mb:=0; ct:= 0.0; e?bi
 ; *[ len(xs) < bi -> xs:= xs ++ [<0.0,0>] ]
 ; *[ true       ; a?x      -> xs:= xs ++ [x]
    | len(xs) > 0; b!hd(xs) -> xs:= tl(xs); mb:= 1
    | true       ; d?       -> c!(<len(xs)+mb,ct>)
    | true       ; m?ct     -> mb:= 0
```

```
    ]
]|

proc M(a: ?lot, b: !lot, m: !real, c: ?real) =
|[ x: lot, td: ->real, mu: real
 | c?mu; td:= exponential(1.0/mu)
 ; *[ true -> a?x; delta sample td; m!(time-x.0); x.0:= time; b!x]
]|

proc E(a: ?lot, b: ?void, c: !info) =
|[ x:lot, xs: lot*
 | *[ true; a?x -> xs:= xs ++ [x]
    | true; b?  -> c!(<len(xs),0.0>)
    ]
]|

proc S(a: (?info)^2, b: (!void)^2, c: (!nat)^1, d: !real, e: (!real)^1) =
|[ m: info, i: nat, ct: real^2, wip: nat^2, bi: nat^1, mu, ts, lt, tend: real
 | mu:=procrate(); bi:=bufinit(); ts:=sampletime(); tend:=term(); i:=0
 ; *[ i < 1 -> e.i!mu; c.i!bi.i; i:= i + 1]
 ; *[ tend > time  -> i:= 0
                     ; *[ i < 2 -> b.i!; a.i?m; ct.i:= m.1; wip.i:= m.0; i:= i + 1 ]
                     ; lt:=control(time,wip); d!lt; delta ts
    | tend <= time -> terminate
    ]
]|

clus Sys()=
|[ a: (-lot)^2, b: (-lot)^1, c: (-info)^2, d: (-void)^2, e: (-nat)^1, f: -real
 , g, m: (-real)^1
 | G(a.0,f)
|| i: nat <- 0..1: B(a.i,b.i,c.i,d.i,e.i,m.i)
|| i: nat <- 0..1: M(b.i,a.(i+1),m.i,g.i)
|| S(c,d,e,f,g)
|| E(a.1,d.1,c.1)
]|

xper= |[ Sys() ]|
```

The model is similar to the model discussed in Appendix A.1, but the model shown here consists of only one workstation and is build to communicate with a Python script that provides simulation settings and the control signal.

The generator process G is set up to communicate with the sampling process S, to obtain the current release rate of lots into the system provided by the control strategy. A repetitive selective waiting structure is chosen that allows the countdown to the release of a new lot into the system to be altered on the fly. This means that as soon as the control signal changes, the time until the next releases is changed accordingly. The conversion from release rate to inter departure time however, causes problems when the release rate becomes 0. Therefore the release rate should be greater than 0, which is accomplished by limiting the minimum release rate to a very low number, just above 0.

The buffer process B is exactly the same as for the model discussed in Appendix A.1. The same applies to the machine process M, except for the fact that this process obtains the average processing rate μ from the sample process S during initialization.

The exit process E is changed so that it can report the number of lots that have left the system to the sample process S. The sample process S is actually the central process that controls the behavior of the complete system. First it collects data from the Python script like the average processing rate μ for the machines, the initial buffer levels, the sampling time and the end time for the simulation. It then initializes the machine by supplying the average processing rate and initializes the buffer levels to the appropriate initial levels. After data is collected on the state of the system it is passed to the Python script together with the time. The Python script then returns the control signal, which is passed to the generator. This cycle of collecting data and returning the control signal is repeated until the end time for the simulation is reached.

The model that is discussed here consists of a single workstation. This model however, can be easily adapted to a system with multiple workstations.

The Python script that provides the χ-model with the necessary settings and the control signal is shown below:

```
import sys,os,string
from mlabwrap import mlab
from Numeric import *

# simulation settings
mu=10.0
simtime=450.0
samplet=0.2
ibuf=tuple([0])
k=(1.0, 1.0)

def control(t,wip):

    global output
```

```
    lt, yref = mlab.control1(t,wip,mu,k,nout=2)
    lt=lt[0][0]
    if lt < 0.001:
        lt=0.001

    output = output + str(t) + '\t'
    for x in range(len(wip)):
        output = output + '\t' + str(wip[x])
    output = output + '\t' + str(yref[0][0]) + '\t' + str(lt) + '\n'

    if t>=simtime-samplet:
        fp = open('control1.dat','w')
        fp.write(output)
        fp.close
    return lt

def sampletime():
    return samplet

def bufinit():
    return ibuf

def procrate():
    return mu

def term():
    return simtime
```

It can be seen in the function 'control', that the control signal is actually calculated using a Matlab function. The Matlab control function as used for the one workstation test case, discussed in section 5.4, is shown below:

```
function [lt,yr] = control(t,x,mu,k)

usat    = 1;
umax    = .95;

% reference trajectory
a1      = 365/(2*pi)*mu*umax/2;
a2      = mu*umax/2;
[yr,yrdot,yrdot2] = yref(t,a1,a2);

% variables
```

```
k1 = k(1);
k2 = k(2);
x1 = x(1);
y  = x(2);

% control action
lt =  mu*x1/(x1+1)...
    + yrdot2*(x1+1)^2/mu...
    - (x1+1)^2/mu*((k1+k2)*(mu*x1/(x1+1)-yrdot) + (k1*k2+1)*(y-yr));
% saturation
if lt<0
    lt=0;
elseif lt > usat*mu
    lt = usat*mu;
end

function [yr,yrdot,yrdot2] = yref(t,a1,a2)

yr     = a1*sin(t*2*pi/365)              + a2*t;
yrdot  = a1*cos(t*2*pi/365)*2*pi/365     + a2;
yrdot2 =-a1*sin(t*2*pi/365)*(2*pi/365)^2;
```

Appendix B

Variability

To obtain information on the general behavior of a manufacturing system that is subject to variability, multiple simulations need to be performed. From these simulation results it is then possible to determine the average behavior of the system.

As the results from different simulations are independent, it is possible to apply classical statistics to quantify the general behavior of the manufacturing system. The general behavior of the system is obtained by taking the sample mean of the different simulation results. The sample mean is defined as:

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}. \tag{B.1}$$

To determine the accuracy of the obtained mean, a $100(1-\alpha)$ percent confidence interval on the mean can be determined:

$$\overline{x} - t_{\alpha/2,n-1} S(n)/\sqrt{n} \leq \mu \leq \overline{x} + t_{\alpha/2,n-1} S(n)/\sqrt{n}, \tag{B.2}$$

where the sample variance $S(n)^2$ is given by:

$$S(n)^2 = \frac{\sum_{i=1}^{n} (X_i - \overline{X})^2}{n-1}. \tag{B.3}$$

Equation B.2 shows that the size of the confidence interval depends on the number of simulations n that are performed. For a larger set of simulations that is performed the confidence interval on the mean is smaller, i.e. the error on the mean is smaller.

The error of the obtained sample mean compared to the population mean μ is defined by [Law00]:

$$\gamma = \frac{|\overline{X}(n) - \mu|}{\mu}. \tag{B.4}$$

However, the population mean μ is unknown and it is therefore impossible to determine the relative error directly. Suppose that n simulations are performed until the relative error can be estimated by:

$$\frac{t_{\alpha/2,n-1} S(n)/\sqrt{n}}{|\overline{X}(n)|} \leq \gamma \tag{B.5}$$

Then it can be shown that $\overline{X}$ has a relative error of at most $\gamma/(1-\gamma)$ with a probability of $1-\alpha$. So to get an actual relative error of at most γ an adjusted relative error of $\gamma' = \gamma/(1+\gamma)$ is needed.

$$\begin{aligned}
1-\alpha &\approx P\big(|\overline{X} - \mu|/|\overline{X}| \leq t_{\alpha/2,n-1} S(n)/\sqrt{n}\big) \\
&\leq P\big(|\overline{X} - \mu| \leq \gamma |\overline{X}|\big) \\
&= P\big(|\overline{X} - \mu| \leq \gamma |\overline{X} - \mu + \mu|\big) \\
&\leq P\big(|\overline{X} - \mu| \leq \gamma |\overline{X} - \mu| + |\mu|\big) \\
&= P\big((1-\gamma)|\overline{X} - \mu| \leq \gamma |\mu|\big) \\
&= P\big(|\overline{X} - \mu|/|\mu| \leq \gamma/(1-\gamma)\big)
\end{aligned} \tag{B.6}$$

Appendix C

Numerical integration

In this appendix the numerical integration techniques used to improve the implementation of the reduced order model (Section 3.1.1) are discussed.

C.1 Simple summation

A numerical method used to calculate the integral on the right hand side of (2.29) is a simple summation:

$$\int_{x=0}^{1} f(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i), \tag{C.1}$$

where n denotes the total number of uniformly distributed discrete points in x and $f(x_i)$ is the function value at point i. The method is visualized in Figure C.1.

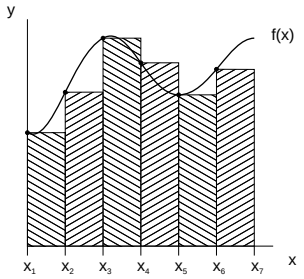


Figure C.1: Integration by summation

It can be seen that given a fixed number of grid points this method for numerical integration is a rough approximation to the exact integral. For an increasing part of

the function the integral is underestimated, while for a decreasing part the integral is overestimated.

C.2 Trapezoidal integration

A more refined method for numerical integration is the method of trapezoidal integration [Wei]:

$$\int_{x=0}^{1} f(x)dx \approx \frac{1}{2(n-1)} \sum_{i=1}^{n-1} (f(x_i) + f(x_{i+1})), \tag{C.2}$$

shown in Figure C.2

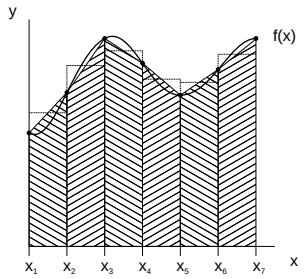


Figure C.2: Trapezoidal integration

The figure shows that, using the same number of grid points as used for the summation method, the method of trapezoidal integration gives a better approximation to the actual integral. Where the method of summation assumed the function $f(x)$ to be piecewise constant, this method approximates the function between two subsequent points using a linear approximation, a 2 point Newton Cotes formula.

C.3 Romberg method

The Romberg method [Dav84] is a method that uses information on the error of the trapezoidal method to reduce this error. The trapezoidal method can be written as:

$$\int_{x=0}^{1} f(x)dx = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) + f(x_{i+1})) + c\left(\tfrac{1}{m^2}\right) + \mathcal{O}\left(\tfrac{1}{m^4}\right), \tag{C.3}$$

where m denotes the number of steps on the interval $x = [0, \ldots, 1]$, c is a constant for the error of order $\frac{1}{m^2}$ and $\mathcal{O}$ contains the higher order errors which are all even powers of $\frac{1}{m}$. This can also be written as:

$$\mathcal{T}_0\big(\tfrac{1}{m}\big) = \mathcal{I} + c\big(\tfrac{1}{m^2}\big) + \mathcal{O}\big(\tfrac{1}{m^4}\big), \tag{C.4}$$

where $\mathcal{T}_0(\frac{1}{m})$ is the trapezoidal approximation of the integral $\mathcal{I}$ on the left hand side of (C.3) with step size $\frac{1}{m}$.

The Romberg method now relies on a combination of trapezoidal approximations with different step sizes to reduce the error. Suppose that the approximation (C.4) is combined with a trapezoidal approximation where the number of steps is doubled according to:

$$\begin{aligned}\mathcal{T}_1\big(\tfrac{1}{2m}\big) &= \alpha\mathcal{T}_0\big(\tfrac{1}{2m}\big) + (1-\alpha)\mathcal{T}_0\big(\tfrac{1}{m}\big) \\ &= \alpha\Big(\mathcal{I} + c\big(\tfrac{1}{4m^2}\big) + \mathcal{O}\big(\tfrac{1}{m^4}\big)\Big) + (1-\alpha)\Big(\mathcal{I} + c\big(\tfrac{1}{m^2}\big) + \mathcal{O}\big(\tfrac{1}{m^4}\big)\Big).\end{aligned} \tag{C.5}$$

By choosing the weighting factor α to be $\frac{4}{3}$ it can be seen that the combination of the two trapezoidal approximations with different step sizes reduces to:

$$\mathcal{T}_1\big(\tfrac{1}{2m}\big) = \mathcal{I} + \mathcal{O}\big(\tfrac{1}{m^4}\big), \tag{C.6}$$

where the error is now of order $\frac{1}{m^4}$, compared to the original error of order $\frac{1}{m^2}$. Since the higher order errors are all even powers of $\frac{1}{m}$, the Romberg procedure can be applied repeatedly to further reduce the error.

Suppose the integral $\mathcal{I}$ is approximated on the interval $[0,1]$ using K trapezoidal approximations $\mathcal{T}_{k,0}; k = 1,\ldots,K$ with step sizes $\frac{1}{2^k}$. Now when $\mathcal{T}_{k,j}$ is the combination of the approximations $\mathcal{T}_{k-1,j-1}$ and $\mathcal{T}_{k,j-1}$ according to:

$$\mathcal{T}_{k,j} = \tfrac{1}{2^{2j}-1}\big(2^{(2j)}\mathcal{T}_{k,j-1} - \mathcal{T}_{k-1,j-1}\big); \qquad j = 1,\ldots,k, \tag{C.7}$$

a triangular set with elements $\mathcal{T}_{k,j}$ is formed where $\mathcal{T}_{K,K}$ is the approximation with the smallest error.