CHALMERS
UNIVERSITY OF TECHNOLOGY

Co-simulation Methods for EPAS and Chassis Systems Development

Master's Thesis in Engineering Mathematics and Computational Science

CANHUI WU

Department of Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Master's thesis 2018

Co-simulation Methods for EPAS and Chassis Systems Development

CANHUI WU

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Mathematical Statistics
Chalmers University of Technology
Gothenburg, Sweden 2018

Co-simulation Methods for EPAS Steering and Chassis Systems Development
Canhui Wu

Supervisor:
Stig Larsson, Department of Mathematical Sciences
Weitao Chen, Volvo Cars, CAE Vehical Dynamics
Shenhai Ran, Volvo Cars, CAE Vehical Dynamics

Examiner:
Stig Larsson, Department of Mathematical Sciences

Master's Thesis 2018
Department of Mathematical Sciences
Division of Applied Mathematics and Mathematical Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Master's Thesis 2018
Department of Mathematics
University of Kaiserslautern
D-67653 Kaiserslautern
Telephone +49 631 205 2251

Typeset in LaTeX
Printed by the Department of Mathematical Sciences
Gothenburg, Sweden 2018

Co-simulation Methods for EPAS and Chassis Systems Development
Canhui Wu
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Co-simulation has become a trending topic of simulation techniques in the recent years with an emerging need for complex system development. However it can show inconsistent results compared with mono-simulation due to the drawbacks of modular integration. Many research works have been done on various types of co-simulation. The scope of this thesis is to have a fundamental understanding including stability and error analysis for parallel co-simulation with *force/displacement* coupling. In the thesis we have shown that the co-simulation error is dominated by the coupling error rather than error by numerical method. In addition the causality and interface selection have an important effect on the robustness of co-simulation results, which can further implies to higher degree extrapolation and multi-subsystems. The study based on a basic dual mass-spring-damper research model can indicate how a complex system can be partitioned in a robust manner.

Besides, a state-of-the-art co-simulation technique called nearly energy-preserving-coupling-element (NEPCE) has been referred and evaluated. The techniques is based on the concept of preserving energy in the power bond and use the residual energy as the error indicator. From the analytical and numerical results of the thesis it is shown that the concept has a limited usage in our research model. A new "causality-based extrapolation" method has been proposed based on the rough knowledge of the system and interface dynamics. It is more accurate than mono-lower-degree extrapolation and faster than mono-higher-degree extrapolation. In certain cases, the improvement can be almost as good as mono-higher-degree extrapolation. This discovery is extendable to multi-subsystems as well.

Lastly the different designs of interface and extrapolation methods have been tested on a co-simulation case study of high fidelity electric power assisted steering (EPAS) model and chassis model.

Keywords: non-iterative co-simulation, causality and interface, force/displacement coupling, error analysis, stability analysis

Acknowledgements

This half year has never been easy. During the journey, there have been a number of people who have assisted, guided and supported me, without their help this thesis would never have happened. First I would like to express my gratidute towards my supervisors, Weitao Chen, Shenhai Ran and Prof. Stig Larsson, who have supported me consistently during this journey.

Also, I would like to thank Dr. Falk Triebsch, Ms Irina Prizigoda, Ms Natalie Scheib, Prof. Dr. Bernd Simeon and Mr. Jamal Nasir, who have helped me before my arrival in Sweden. Thank you for guiding me through the Erasmus exchange program, this would have never happened without your help.

Furthermore, I would like to thank my dear friends Xiaohui Zheng, Jemimah Mae Eugenio and Firoozeh Aga, who have supported through this journey. From moving out to moving in Kaiserslautern, Xiaohui has always offered more than I can expect. I will never forget the rainy evening before my flight to Gothenburg, Jemimah and Firoozeh walked me off to the central train station, they gave me enough support and faith to start this journey.

Last but not least, I would like to thank my dear family, especially my mother. She is always my biggest support in everything, she is my superwoman. I would not be who I am today without her support.

Canhui Wu, Gothenburg, June 2018

Contents

List of Figures

List of Tables

1
Introduction

1.1 Motivation

Automotive industry changes drastically with a trend of electrification and automation, where various needs emerge from research to production. For instance, the vehicle system is getting complex with a combination of mechanical system, electronic system, hydraulic and thermodynamic system, from which it requires a more advanced secure solution. In a complex multidisciplinary system it requires more than one simulation tools, or even a specific tool for each system [4]. As a result, the difficulty of mechanical mathematical model grows. However technological changes realise rarely smoothly but rather in pulses. The complex multidisciplinary problems urge the research of a compatible method, which is able to solve various systems with suitable solvers or tools correspondingly. CAE (computer-aided engineering) simulation is heavily used in vehicle development, due to the fact that it reduces development cost and time. Therefore a general compatible method should be created for the multidisciplinary problems. Co-simulation is the answer for that. Co-simulation is able to model coupled problem and solve them in a distributed manner with the help of FMI (Functional Mock-Up Interface), which is a tool independent standard to support both model exchange and co-simulation of dynamic models [1].

Co-simulation also enables system development in parallel, independently between the clients and suppliers. However, research and development of these multidisciplinary systems are often divided between different suppliers. Each supplier develops its partial solution in a black box manner [13]. Due to the Intellectual Property (IP), most of the details about the black box (including tools, domains, time step, property) are not accessible in co-simulation sharing environment. In order to get a complete solution of the system, one has to couple all the partial solutions or systems from different suppliers. The later the coupling process is done in co-simulation, the less accurate it will be [26] due to the numerical drawback of modular integration. Thus we are interested in researching the propagation of error and change of stability, the preference of coupling design, and further improvements in co-simulation.

1.2 Co-simulation

Model is an abstract expression of an object that only remains relevant meaningful details. Co-simulation is the combination of the theory and techniques to enable

global simulation of a coupled system via the composition of simulators [13]. Overall co-simulation is responsible in coordination and control of the progress of time in the co-simulation and the propagation of information between the constituent models [12].

From a mathematical point of view, modular integration and coupling variables (input and output variables) exchange are the main characteristics in co-simulation. Coupling variables are predefined so are the exchange process, while it only occurs at discrete time points. Before reaching these discrete time points each simulator runs simulation independently. This process in consists of modular integration method in combination with extrapolation or interpolation techniques, depending on the coupling type.

The main differences between co-simulation and mono-simulation are:

- In co-simulation intermediate results are exchanged only in discrete time points. While in mono-simulation all the results are calculated at the same time or intermediate results are exchanged continuously.
- In co-simulation each subsystem can choose different numerical solvers and discrete settings while these are not applicable in mono-simulation.
- Apart from numerical error there is also coupling error in co-simulation due to the usage of extrapolation or interpolation in between the exchange interval.

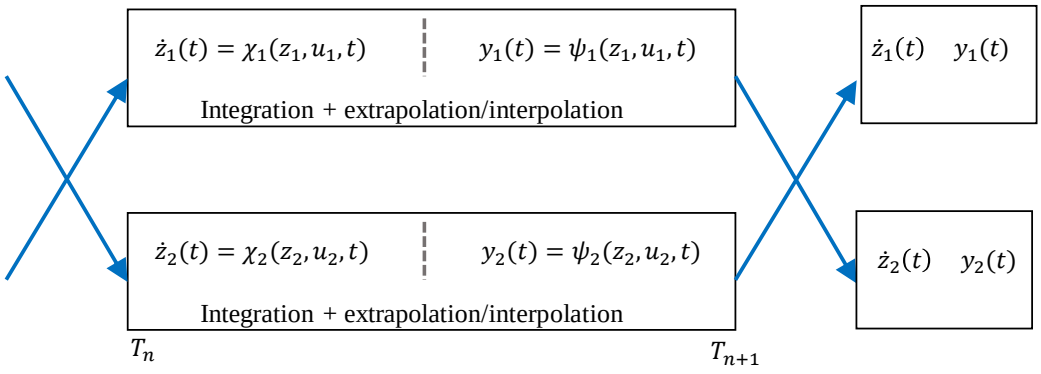

Figure 1.1: Weakly coupled system example

Figure 1.1 demonstrates the differences in co-simulation comparing to mono-simulation. Here we call the co-simulated system as weakly coupled system. The mono-simulated system is the strongly coupled system, which can be seen as the reference.

In addition, systems can be weakly coupled in different ways depending on the choice on modular, communication pattern and numerical level. Figure 1.1 represents one kind of weakly coupled system. More details about the classification will be introduced in the following chapters.

From an engineering point of view, FMI is a model based development standard. It enables model exchange and interconnection in simulation. FMI for co-simulation is covered in MODELISAR project [1]. Systems can be exported as FMU (Functional Mock Unit) to a co-simulation environment, where interfaces are set by FMI definition and intermediate results (input, output variables or other status information) are exchanged.

1.3 Literature Review

Co-simulation survey and taxonomy by Gomes, Thule, Broman, Larsen and Vangheluwe [13] gave a complete summary for co-simulation approaches, challenges, opportunities and classification for the past 5 years. Communication pattern and extrapolation techniques in weakly coupled system were analysed and compared by Busch [6], in terms of stability, error order and computation effort in co-simulation. Error analysis for weakly coupled systems that are free of algebraic loop were investigated by Arnold, Clauss and Schierz [3]. It was proven that error order reduction only occurs in local error of weakly coupled systems that are free of algebraic loop, but not in global error. A master algorithm for step size control in FMI compatible environment based on error analysis was created by Schierz, Arnold and Clauß [23]. Numerical stability of co-simulation with different modular coupling methods, constraints coupling and applied force coupling, were analysed by Li [19] with a linear two-mass oscillator as the research model. A context based extrapolation for multi-core simulation using FMI was created by A. Ben Khaled-El Feki et.al [11][17], where appropriate polynomial is selected for signal forecasting. Stability and convergence for sequential modular time integration in multibody system were investigated by Arnold [2], a linear implicit stabilization technique was proposed to guarantee numerical stability and convergence.

1.4 Contributions

In this thesis *force/displacement* coupling with *Jacobi* communication pattern model has been researched. First, we investigate the stability of our research model where different degrees of extrapolation are used. Furthermore, we compare the stability with and without the usage of numerical methods. As extrapolation degree increases, stability region decreases. It further decreases when the explicit numerical method is involved. Second, we compute different errors' propagation in co-simulation in both cases, namely with and without numerical methods. Input value is approximated by certain degrees of extrapolation polynomial. From the error analysis we know that coupling error is the major error source in co-simulation. It is important to design a robust weakly coupled system. Hence, causality and interface designs are analysed for our research model. A preference of coupling design is given at the end of the analysis, these results can be extended to multi-subsystem and higher degree of extrapolation co-simulation. Besides, an error indicator method called 'Nearly Energy Preserving Coupling Element' method is discussed and analyzed about its

functionality in this thesis as well. Without violating the IP of co-simulation black box, this method requires only input and output information to defein an error indicator. But it turns out to have a limited usage in our research model. Lastly, an economical and well-behaved extrapolation method in co-simulation call *causality-based* extrapolation is introduced. It is a good alternative for *force/displacement* co-simulation, as it requires only very few knowledge in the weakly coupled system, presents lager stability region and delivers good results in co-simulation without huge amount of computation cost.

1.5 Outline

This thesis is composed of 9 chapters. In this chapter a ground knowledge of co-simulation from academic and industrial point view is presented. Chapter 2 presents more mathematical background for co-simulation, co-simulation classification and our research model in both strongly coupled and weakly coupled senses. Classical numerical analysis for co-simulation is demonstrated in chapter 3 and chapter 4. It consists of theoretical stability and error analysis and numerical simulation based on the research model. Chapter 5 presents a preference of causality and interface design for *force/displacement* co-simulation by comparing the weakly coupled solution and the corresponding strongly coupled solution. The numeric experiments are performed to test the theoretical results. In Chapter 6 Nearly Energy Preserving Coupling Element is evaluated from definition to functionality. Chapter 7 introduces the *Causality-based* extrapolation from motivation to functionality and compare with the simulation results with mono-degree extrapolation. Chapter 8 presents the numerical test on EPAS and chassis systems, base on the theoretical results from the previous chapters. A summary, limitation and possible future work for this thesis is given in the last chapter.

2

Co-simulation Model

2.1 Introduction

This chapter presents the fundamental knowledge in different types of differential equation, which is the mathematical background for co-simulation. Besides a detailed classification of co-simulation is presented as well. It exposes weakly coupled systems in three different levels, and the characteristics of different coupling methods. Lastly, based on a clear knowledge on the mathematical and engineering background, the research model is introduced.

2.2 Basics on Differential Equations

Differential equations (DEs) can describe nearly all the fundamental principles that govern physical processes of engineering interest [14]. Therefore research in the well posedness, stability and other properties depending on differential equations are of high interest in mathematical sense.

Modelling a physical phenomenon can differ from case to case in terms of different kinds of DEs. DEs can be classified into several types, most commonly are: linear/ non-linear, homogeneous/ inhomogeneous and ordinary/ partial. Linearity and homogeneity of DEs are rather straight forward, by looking at the coefficients and source term. The focus in this section is on the ordinary/ partial DEs.

2.2.1 Ordinary Differential Equation

Definition 2.2.1. An nth-order ordinary differential equation (ODE) is a relation

$$F(x, u, u', ..., u^{(n)}) = 0 \tag{2.1}$$

connecting a single independent variable x and dependent variable u and its derivatives $u, u', ..., u^{(n)}$ w.r.t to x. Usually the single independent variable is time variable [16].

Example 1. Logistic differential equation

$$\frac{d}{dx} f(x) = f(x)\,(1 - f(x)) \tag{2.2}$$

2.2.2 Partial Differential Equation

Recall the definition of ODE, a differential equation with the amount of independent variables is greater than 1 is partial differential equation [16].

Definition 2.2.2. A first order partial differential equation (PDE) is a relation

$$F(x_1, ..., x_n, u, p_1, ..., p_n) = 0 \tag{2.3}$$

where

$$p_i = \frac{\partial u}{\partial x_i}, \quad i = 1, ...n$$

connecting a vector of variables $x_i, i = 1, ...n$ and dependent variable u and its partial derivatives $\frac{\partial u}{\partial x_i}, \quad i = 1, ...n$. The order of PDE is the highest derivative order occur in the equation.

Example 2. Continuity equation(fluid dynamics)

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0, \tag{2.4}$$

where ρ is fluid density, t is time, $\boldsymbol{u}$ is flow velocity vector field, and ∇ is divergence.

2.2.3 Differential Algebraic Equation

Differential algebraic equation (DAE) has similar definition like ODE but with extra algebraic constraints involved.

Definition 2.2.3. A system of equations given in implicit form

$$F(t, \boldsymbol{x}, \dot{\boldsymbol{x}}(t)) = 0, \tag{2.5}$$

is called DAE, if the jacobian matrix $\frac{\partial F}{\partial \dot{\boldsymbol{x}}}$ is singular, where $t \in \mathbb{R}$, $\boldsymbol{x}(t) \in \mathbb{R}^n$.

Definition 2.2.4. The differential index of a DAE is the minimum differentiation steps it requires to transform a DAE [24]

$$F(t, x, \dot{x}(t)) = 0, \tag{2.6}$$

to a ODE

$$\dot{x}(t) = f(t, x). \tag{2.7}$$

Example 3. Constrained mechanical model (CMM)
Consider a mathematical pendulum problem,

$$\begin{aligned} m\ddot{x} + 2\lambda x &= 0, \\ m\ddot{y} - my + 2\lambda y &= 0, \\ x^2 + y^2 - l^2 &= 0. \end{aligned} \tag{2.8}$$

where λ is Lagrange multiplier, $(x(t), y(t))$ is the position at time t. As known, this is a index 3 problem.

Roughly speaking, non-linearity, high index represents the difficulty in solving DAE. There are no general results available for the solvability and stability of nonlinear DAE. Therefore, one often considers a linearized DAE model [15].

2.2.4 State Space Representation

State space representation is a mathematical model that generates the a physical phenomenon by first order DE with a set of input, output and state variables.

Definition 2.2.5. State space representation

$$\begin{aligned} \dot{\boldsymbol{z}}(t) &= f(\boldsymbol{z}(t), \boldsymbol{u}(t)), \\ \boldsymbol{y}(t) &= g(\boldsymbol{z}(t), \boldsymbol{u}(t)) \end{aligned} \tag{2.9}$$

where $\boldsymbol{z}(t) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{y}(t) \in \mathbb{R}^q$ is the output vector and $\boldsymbol{u}(t) \in \mathbb{R}^p$ is the input vector. If it is linear system, one can rewrite the state space representation as:

$$\begin{aligned} \dot{\boldsymbol{z}}(t) &= \boldsymbol{A}(t)\boldsymbol{z}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}(t)\boldsymbol{z}(t) + \boldsymbol{D}(t)\boldsymbol{u}(t) \end{aligned} \tag{2.10}$$

where $\boldsymbol{A}(t) \in \mathbb{R}^{n\times n}$ is the state matrix, $\boldsymbol{B}(t) \in \mathbb{R}^{n\times p}$ is the input matrix, $\boldsymbol{C}(t) \in \mathbb{R}^{q\times n}$ is the output matrix and $\boldsymbol{D}(t) \in \mathbb{R}^{q\times p}$ is the feed-through matrix. Feed-through matrix describes the dependency between system output and input.

Remark. *In our coupling mechanical systems, we used state space representations as our mathematical model.*

2.3 Co-simulation Classification

For a coupled system, the most clear difference between weakly coupled and strongly coupled system is the interface. The interface defines different subsystems. And each subsystem communicates at discrete time point through an interface while subsystem in strongly coupled system communicates continuously. We classify the weakly coupling techniques into 3 different perspectives: modular, communication pattern and numerical approach.

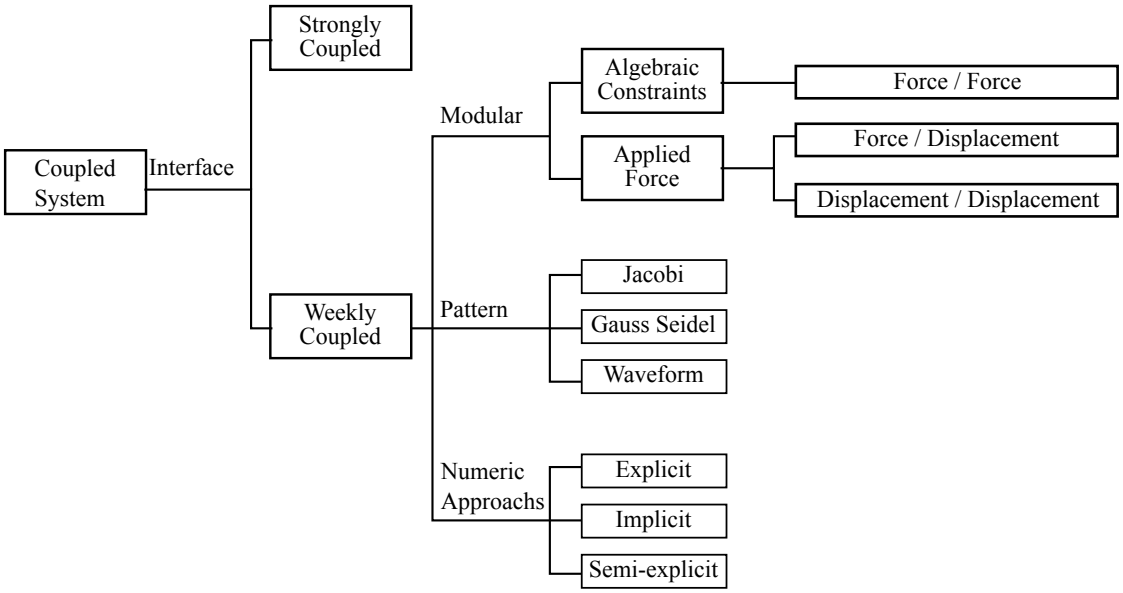

Figure 2.1: Co-simulation classification

In weakly coupled system, modular level is about how to design input and output. Algebraic constraint coupling means each subsystem are coupled via constraint forces (i.e. via Lagrange multipliers) [6]. It can be having force as input and output in each subsystem, which leads us to DAE model by summarizing the state space equations. In the well-known DAE there are stability, index, computation problems. Furthermore there will be an algebraic loop in the co-simulation, it is a challenge to deal with algebraic constraint coupling. One example of algebraic constraint coupling is *force/force* coupling, where each subsystem has force as input and output.

Remark. *Algebraic loop occurs when a variable indirectly depends on itself. To see how it arise in co-simulation, let us assume that there are two subsystems, take the output equation in general state space representations an example:*

$$\begin{aligned}
\boldsymbol{y}_1 &= g_1(\boldsymbol{z}_1, \boldsymbol{u}_1),\\
\boldsymbol{u}_1 &= \boldsymbol{y}_2,\\
\boldsymbol{y}_2 &= g_2(\boldsymbol{z}_2, \boldsymbol{u}_2),\\
\boldsymbol{u}_2 &= \boldsymbol{y}_1.
\end{aligned}$$

therefore $\boldsymbol{y}_1$ *indirectly depends on itself, if*

$$\frac{\partial g_i(\boldsymbol{z}_i(t).\boldsymbol{u}_i(t))}{\partial \boldsymbol{u}_i(t)} \neq 0, \quad i = 1, 2 \quad \forall t \in \mathbb{T},$$

where $\mathbb{T}$ *is total simulation time set. This result can be extended to multi-subsystems co-simulation.*

In applied force coupling, we have *force/displacement* coupling and *displacement /displacement* coupling, both of them are free of algebraic loop. In *force/displacement* coupling, each subsystem has either force as input or output and displacement

as output or input. Hence there is a so called causality problem, whether the subsystem should output force or input force. Force as input in different subsystem changes the co-simulation performance. Therefore, one has to be careful when it comes to causality problem. While in *displacement/displacement* coupling, each subsystem has displacement as input and output. This is more accurate than *force/displacement* [6], more details will be given in Chapter 4.

Remark. *In modular coupling level, a complementary term called causality (causal or acausal)need to be introduced. Causal approach describes the system's physics in form where the direction (causality) of signal flows explicitly, while acausal approach is not [10].*

After chosen the modular level, we take a look at the communication pattern between subsystems. The common choices are: *Jacobi*, *Gauss Seidel* and *waveform*.

In *Jacobi* pattern, each subsystem integrates its own dynamics in parallel and independently. For details one can see it from Figure 2.2.

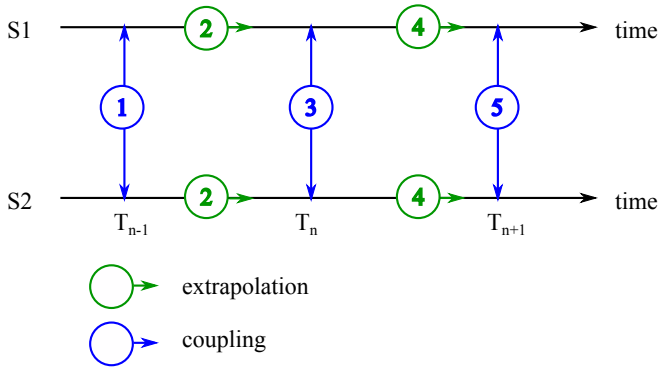

Figure 2.2: *Jacobi* communication pattern

Assume there are two different subsystems S_1 and S_2 in co-simulation, at each communication instant T_{n-1}, T_n and T_{n+1}, subsystems communicate with each other. After the communication action, each subsystem integrates within itself independently and in parallel till the next communication instant. As indicated in Figure 2.2, the number represents the order of time for co-simulation. $1, 3, 5$ are coupling steps at communication instant, $2, 4$ represent independent integration steps on each subsystem, where extrapolation is needed for unknown input value at each micro time step. This communication pattern is usually more efficient in terms of computational cost but less accurate in comparison with *Gauss Seidel* [6].

In *Gauss Seidel*, the coupled subsystems integrate its own dynamics sequentially. It is slightly more complicated than *Jacobi* communication pattern, as the simulators are integrated in a sequential order where extrapolation and interpolation techniques are both needed for unknown input value.

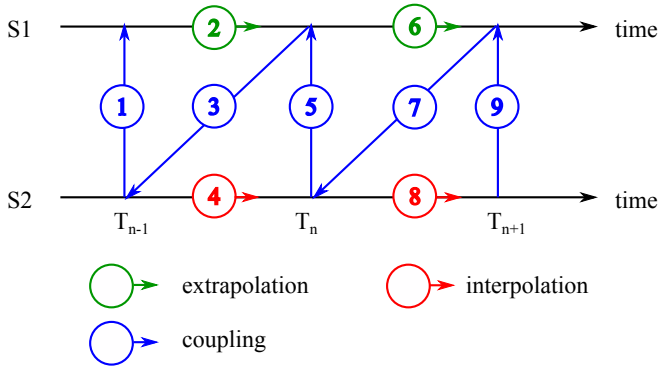

Figure 2.3: *Gauss Seidel* communication pattern

Again the number indicated in Figure 2.3 represents the order of time for co-simulation, where red color represents interpolation and green color represents extrapolation(the same rule applies in *waveform* pattern). At communication instant T_{n-1} subsystem S_1 obtains a new input value from subsystem S_2, then integrates within itself till next communication instant T_n, where extrapolation for unknown input value at each micro time step is needed. At communication instant T_n subsystem S_2 obtains a new input value from subsystem S_1, then integrates within itself at macro step $(T_{n-1}, T_n]$, where interpolation is needed for input value approximation. There is a sequential order of integration in Figure 2.3, which means *Gauss Seidel* communication pattern takes longer time for simulation but also is more accurate than *Jacobi* [6].

Remark. *Figure 2.3 has explicitly indicated the sequential order of calculation for each subsystem, but this is not the case in practise in general. There is a so called master algorithm controls the sequential order for the subsystems.*

In *waveform* pattern, the coupled subsystems communicate in an interactive way, which means it has the highest level in terms of difficulty, computation cost and stability [6].

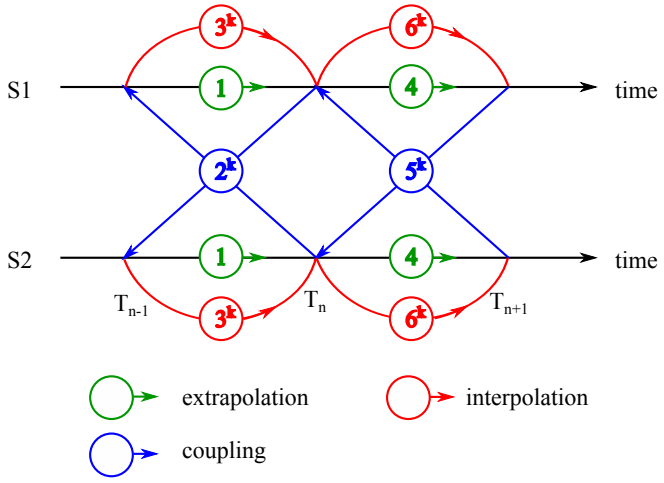

Figure 2.4: *Waveform* communication pattern

In *waveform* pattern, after communication instant T_{n-1} each subsystem starts like *Jacobi* pattern integrating independently and in parallel, where extrapolation for unknown input value at each micro time step is involved, marked as number 1 in green color in Figure 2.4. At communication instant T_n, two subsystems communicate with each other, the obtaining new input value is used again from T_{n-1}, the difference is that instead of extrapolation technique, interpolation technique is involved for input value approximation at macro time step $(T_{n-1}, T_n]$, marked as number 3 in red color on in Figure 2.4. This interactive communication between subsystems marked as number 2 and interpolation procedure marked as number 3 are ongoing repeatedly k times till the error is lower than predefined error tolerance, which is controlled by the master algorithm. Then the subsystems starts again the same procedure on $(T_n, T_{n+1}]$ like $(T_{n-1}, T_n]$.

The last in weakly coupled classification is from numerical approach level, this differs from case to case depending on the differential equation. Ordinary Differential Equation (ODE) and Differential Algebraic Equation (DAE) solvers can be chosen differently, as well as specific time step.

2.4 Research Model

Due to the fact that *Jacobi* communication patter is easy to implement (without master algorithm), fast to compute and does not require the backward calculation ability for simulator. Thus it is widely used in industry. The main focus of our research is *force/displacement* coupling approach in combination with *Jacobi* communication pattern.

The difference of macro time step, micro time step and extrapolation in co-simulation can be found in Figure 2.5.

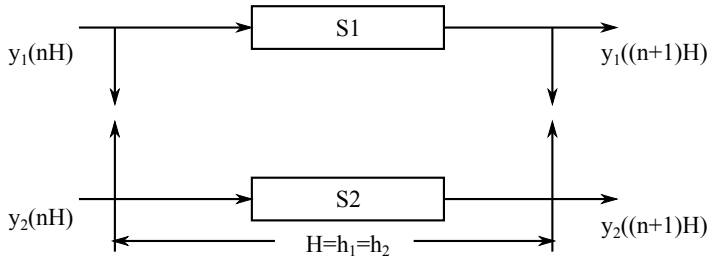

Figure 2.5: *Jacobi*:macro step size = micro step size

H, h_1, h_2 are denoted as macro step size, micro step size in subsystem S_1 and micro step size in subsystem S_2 respectively. For $H = h_1 = h_2$, subsystems communicate with each other after one calculation, and each time there would be one extrapolation needed. This is more efficient comparing to *Jacobi* in combination with $H \neq h_1 \neq h_2$, but more computational expense at the same time.

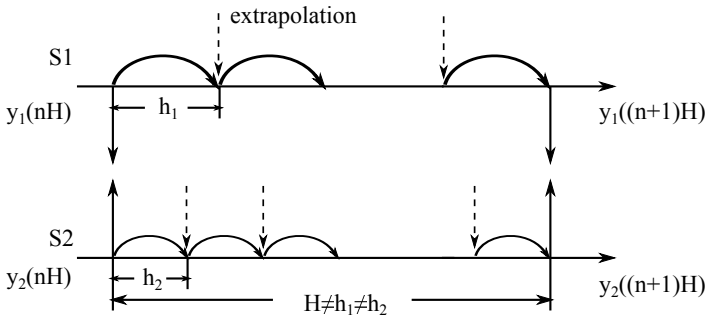

Figure 2.6: *Jacobi*: macro step size unequal to micro step size

From Figure 2.6 one can see that between each communication instant, input value is unknown as there is no communication between two subsystems. Therefore, extrapolation for unknown input value is needed at each micro step time point. The larger macro time step is, the more extrapolation approximation is needed. This turns out to be an important source of global error, the details will be explained in Chapter 4.

For research purposes, we choose dual mass spring damping system in Figure 2.7 as the research model, because dual mass spring system can represent most of the mechanical systems. For instance, the vehicle model and steering system can be seen as a dual mass spring system or a steering system can be also decoupled into a dual mass spring system.

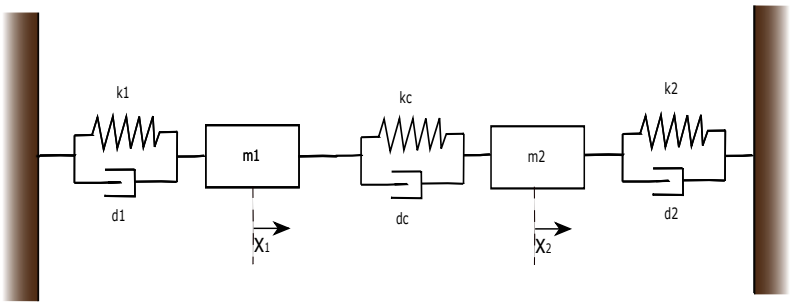

Figure 2.7: Strongly coupled system

We can generate the dynamics as follow:

$$\begin{aligned} \dot{\boldsymbol{z}} &= \boldsymbol{A}\boldsymbol{z}, \\ \boldsymbol{z} &= \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix}, \\ \boldsymbol{A} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_c}{m_1} & -\frac{d_1+d_c}{m_1} & \frac{k_c}{m_1} & \frac{d_c}{m_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_c}{m_2} & \frac{d_c}{m_2} & -\frac{k_c+k_2}{m_2} & -\frac{d_c+d_2}{m_2} \end{bmatrix}. \end{aligned} \tag{2.11}$$

As we mentioned before, strongly coupled system has no interface, and all the state variables are integrated together. In order to compare weakly coupled system with strongly coupled system more easily, we can understand that there is an imaginary interface exists in the strongly coupled system. But subsystems exchange intermediate results continuously, which means no extrapolation approximation for unknown input value is needed.

Here we are interested in using a *force/displacement coupling* approach in combination with a *Jacobi* scheme to connect two subsystems and researching the effect on global error and stability. Therefore we decoupled this mass spring damping system it into two subsystems S_1 and S_2 as in Figure 2.8 and take this as our research model.

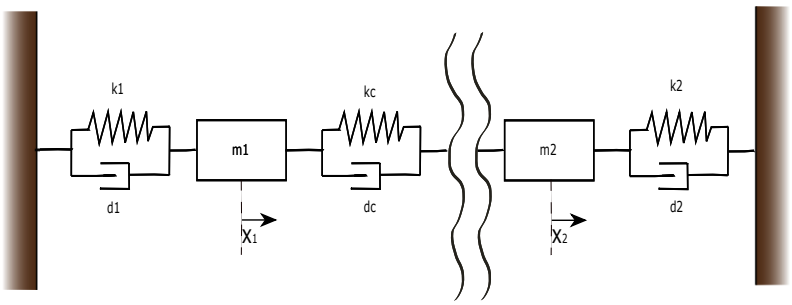

Figure 2.8: Research model

In each subsystem, state motion can be described as a second order ODE by Newton's law:

$$m\ddot{x} + d\dot{x} + kx = f \tag{2.12}$$

and an output linear equation

$$\boldsymbol{y} = g(\boldsymbol{z}, \boldsymbol{u}). \tag{2.13}$$

Thus, we can summarize the state space equations for each subsystem as below. For subsystem S_1:

$$\begin{aligned} \dot{\boldsymbol{z}}_{\boldsymbol{1}} &= f_1(\boldsymbol{z}_{\boldsymbol{1}}, \boldsymbol{u}_{\boldsymbol{1}}) = \boldsymbol{A}_{\boldsymbol{1}}\boldsymbol{z}_{\boldsymbol{1}} + \boldsymbol{B}_{\boldsymbol{1}}\boldsymbol{u}_{\boldsymbol{1}}, \\ \boldsymbol{y}_{\boldsymbol{1}} &= g_1(\boldsymbol{z}_{\boldsymbol{1}}, \boldsymbol{u}_{\boldsymbol{1}}) = \boldsymbol{C}_{\boldsymbol{1}}\boldsymbol{z}_{\boldsymbol{1}} + \boldsymbol{D}_{\boldsymbol{1}}\boldsymbol{u}_{\boldsymbol{1}}, \end{aligned} \tag{2.14}$$

where

$$\boldsymbol{z}_{\boldsymbol{1}} = \begin{bmatrix} x_1 \\ \dot{x}_1 \end{bmatrix}, \quad \boldsymbol{y}_{\boldsymbol{1}}, \boldsymbol{u}_{\boldsymbol{1}} \in \mathbb{R}^2. \tag{2.15}$$

Analogously for subsystem S_2:

$$\begin{aligned} \dot{\boldsymbol{z}}_{\boldsymbol{2}} &= f_2(\boldsymbol{z}_{\boldsymbol{2}}, \boldsymbol{u}_{\boldsymbol{2}}) = \boldsymbol{A}_{\boldsymbol{2}}\boldsymbol{z}_{\boldsymbol{2}} + \boldsymbol{B}_{\boldsymbol{2}}\boldsymbol{u}_{\boldsymbol{2}}, \\ \boldsymbol{y}_{\boldsymbol{2}} &= g_2(\boldsymbol{z}_{\boldsymbol{2}}, \boldsymbol{u}_{\boldsymbol{2}}) = \boldsymbol{C}_{\boldsymbol{2}}\boldsymbol{z}_{\boldsymbol{2}} + \boldsymbol{D}_{\boldsymbol{2}}\boldsymbol{u}_{\boldsymbol{2}}, \end{aligned} \tag{2.16}$$

where

$$\boldsymbol{z}_{\boldsymbol{2}} = \begin{bmatrix} x_2 \\ \dot{x}_2 \end{bmatrix}, \quad \boldsymbol{y}_{\boldsymbol{2}}, \boldsymbol{u}_{\boldsymbol{2}} \in \mathbb{R}^2. \tag{2.17}$$

For the coupling process between two subsystems:

$$\begin{bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \end{bmatrix} = \boldsymbol{u} = L\boldsymbol{y} = L \begin{bmatrix} \boldsymbol{y_1} \\ \boldsymbol{y_2} \end{bmatrix} \\ \boldsymbol{L} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{I} & \boldsymbol{O} \end{bmatrix}, \tag{2.18}$$

where $\boldsymbol{O} \in \mathbb{R}^{2\times 2}$ is a zero matrix, $\boldsymbol{I} \in \mathbb{R}^{2\times 2}$ is identity matrix.

In the research model, subsystem S_1 and subsystem S_2 are coupled via *force/displacement*, where subsystem S_2 has an external force f as input, subsystem S_1 has the force f as output:

$$\boldsymbol{u_1} = \begin{bmatrix} x_2 \\ \dot{x}_2 \end{bmatrix} \qquad y_1 = f$$
$$u_2 = f \qquad \boldsymbol{y_2} = \begin{bmatrix} x_2 \\ \dot{x}_2 \end{bmatrix}$$

where

$$f = k_1(x_1 - x_2) + d_1(\dot{x}_1 - \dot{x}_2).$$

In other literature [6], subsystem S_1 is also called base point excited 1-DOF oscillator and subsystem S_2 is called force driven 1-DOF oscillator. The matrix coefficient can be formulated as follows:

$$\boldsymbol{A_1} = \begin{bmatrix} 0 & 1 \\ -\frac{k_1+k_c}{m_1} & -\frac{d_1+d_c}{m_1} \end{bmatrix}, \quad \boldsymbol{B_1} = \begin{bmatrix} 0 & 0 \\ \frac{k_c}{m_1} & \frac{d_c}{m_1} \end{bmatrix}, \quad \boldsymbol{C_1} = \begin{bmatrix} 0 & 0 \\ k_c & d_c \end{bmatrix}, \quad \boldsymbol{D_1} = \begin{bmatrix} 0 & 0 \\ -k_c & -d_c \end{bmatrix},$$

$$\boldsymbol{A_2} = \begin{bmatrix} 0 & 1 \\ -\frac{k_2}{m_2} & -\frac{d_2}{m_2} \end{bmatrix}, \quad \boldsymbol{B_2} = \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix}, \quad \boldsymbol{C_2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{D_2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

It is not hard to see that, when one system outputs displacement/velocity, it has $\boldsymbol{D}$ matrix is a zero matrix. $\boldsymbol{D}$ matrix describes the dependency between system output and input. We call the subsystem direct feed-through, if $\boldsymbol{D}$ is not a zero matrix. Vice versa, the subsystem is nondirect feed-through, if $\boldsymbol{D}$ is a zero matrix.

Remark. *In* displacement/displacement *weakly coupled system, each subsystem has displacement/velocity as input and output. Therefore, every subsystem in* displacement/displacement *coupling is nondirect feed-through. This is a property that affect local error order, more details will be exposed in Chapter 4.*

2.5 Research Questions

Given the background of co-simulation and our research model, in this thesis we aim to find out:

- How do local solver, extrapolation degree and step size evolve in co-simulation?
- How to choose causality for a given model?
- How does NEPCE method work as error indicator in co-simulation?

3

Weakly Coupled System Stability Analysis

3.1 Introduction

The strongly coupled dual mass spring damping system is stable, it will become static after sometime. But in weakly coupled system an approximated input is added, how the stability will change needs to be known. In this chapter weakly coupled model linear stability, numerical stability and zero stability will be discussed. In our weakly coupled research model input value extrapolation can be chosen from different extrapolation techniques and degrees. In this thesis, we analyse the stability in combination with *Lagrange polynomials*. Analysis processes are similar for other extrapolation technique e.g. *Hermite approximation*, the result can be found in M.Busch [6].

3.1.1 Zero Stability

A numerical method is zero stable if the solution remains bounded as simulation step size converges to 0, for finite final time T. In co-simulation, a coupling method is called zero stable if the co-simulation results converge for an infinitesimal macro step size, i.e. $H \to 0$ [6]. Zero stability may be seen directly in the coupling structure. If all the subsystems are direct feed-through, an algebraic loop will arise [18], which leads to stability problems in DAE systems [6]. Therefore, zero stability can not be guaranteed [18].

3.1.2 Numerical Stability

Getting an arbitrary small simulation step size is not practical, thus we want to know whether a relative small step size will lead to convergence as well. This is the well-known numerical stability. In co-simulation numerical stability is always studied by assuming that the system being co-simulated is stable [13]. There is no point to discuss an unstable strongly coupled system and an corresponding unstable weakly coupled system. One way to check stability is to calculate the spectral radius of the weakly coupled system. Numerical stability is a desired property of a numerical method.

3.2 Stability Analysis

We have discrete weakly coupled system:

$$\begin{aligned} \dot{\tilde{\boldsymbol{z}}}_n &= \boldsymbol{A}\tilde{\boldsymbol{z}}_n + \boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}_n), \\ \tilde{\boldsymbol{y}}_n &= \boldsymbol{C}\tilde{\boldsymbol{z}}_n + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\ \tilde{\boldsymbol{u}}_n &= \boldsymbol{L}\tilde{\boldsymbol{y}}_n, \end{aligned} \tag{3.1}$$

where $\tilde{\boldsymbol{z}}, \tilde{\boldsymbol{y}}, \tilde{\boldsymbol{u}}$ is the weakly coupled notation for the state variable $\boldsymbol{z}$, the output variable $\boldsymbol{y}$ and the input variable $\boldsymbol{u}$ respectively, Φ is the extrapolation operator, and

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & & & \\ & \boldsymbol{A}_2 & & \boldsymbol{O} \\ & & \ddots & \\ \boldsymbol{O} & & & \boldsymbol{A}_{n-1} \\ & & & & \boldsymbol{A}_n \end{bmatrix}$$

is a *Jordan* matrix, each *Jordan* block is the $\boldsymbol{A}_i$ in i-th subsystem. $\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}$ are *Jordan* matrices as well, and the *Jordan* blocks are the corresponding matrix presentations in each subsystem.

We want to know the stability and numerical stability for weakly coupled system in combination of constant, linear and quadratic extrapolation respectively. Thus we need to transform equation (3.1) into a linear system:

$$\begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{y} \end{bmatrix}_{n+1} = \boldsymbol{A}^* \begin{bmatrix} \boldsymbol{z} \\ \boldsymbol{y} \end{bmatrix}_n$$

and compute the spectral radius in $\boldsymbol{A}^*$. For the differential equation, we will solve it exactly and with *Forward Euler* method respectively. Then we compare the spectral radius in both cases in combination with different degrees of extrapolation.

3.2.1 Constant Extrapolation

Under constant extrapolation approximation:

$$\Phi(\tilde{\boldsymbol{u}}(\tau)) = \tilde{\boldsymbol{u}}_n, \quad \tau \in (nH, (n+1)H]. \tag{3.2}$$

First let us solve the differential equation exactly by using *Variation of Constant Formula*. The discrete system can be written as:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}(\tau))d\tau, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}\boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n + \boldsymbol{C}e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \boldsymbol{D}\boldsymbol{L}\tilde{\boldsymbol{y}}_n, \\
\text{where}\quad \boldsymbol{K}(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau
\end{aligned}
\tag{3.3}
$$

To combine above equations together:

$$
\begin{bmatrix}\tilde{\boldsymbol{z}}_{n+1}\\ \tilde{\boldsymbol{y}}_{n+1}\end{bmatrix} = \begin{bmatrix} e^{\boldsymbol{A}H} & \boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L} \\ \boldsymbol{C}e^{\boldsymbol{A}H} & \boldsymbol{C}\boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L} + \boldsymbol{D}\boldsymbol{L}\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{z}}_n\\ \tilde{\boldsymbol{y}}_n.\end{bmatrix} \tag{3.4}
$$

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

Assuming there is *Forward Euler* method involved in combination with constant extrapolation, the discrete weakly coupled system can be written as:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= \tilde{\boldsymbol{z}}_n + \dot{\tilde{\boldsymbol{z}}}_n H, \\
&= \tilde{\boldsymbol{z}}_n + (\boldsymbol{A}\tilde{\boldsymbol{z}}_n + \boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}_n))H, \\
&= (\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_n, \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_n + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + (\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_n.
\end{aligned}
\tag{3.5}
$$

To combine above equations together:

$$
\begin{bmatrix}\tilde{\boldsymbol{z}}_{n+1}\\ \tilde{\boldsymbol{y}}_{n+1}\end{bmatrix} = \begin{bmatrix} \boldsymbol{I} + \boldsymbol{A}H & \boldsymbol{B}\boldsymbol{L}H \\ \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H) & \boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L}\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{z}}_n\\ \tilde{\boldsymbol{y}}_n\end{bmatrix}. \tag{3.6}
$$

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

3.2.2 Linear Extrapolation

Under linear extrapolation approximation:

$$
\Phi(\tilde{\boldsymbol{u}}(\tau)) = \tilde{\boldsymbol{u}}_n + (\tau - nH)\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H}, \quad \tau \in (nH, (n+1)H]. \tag{3.7}
$$

First let us solve the differential equation exactly by using *Variation of Constant Formula*. The discrete system can be written as:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}(\tau))d\tau, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\left(\tilde{\boldsymbol{u}}_n + (\tau - nH)\,\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H}\right)d\tau, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n \\
&\quad + (\int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\frac{(\tau - nH)}{H}d\tau)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n \\
&\quad - \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\frac{(\tau - nH)}{H}d\tau)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_{n-1}, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + (\boldsymbol{K}(H) + \boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n - \boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_{n-1}, \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}(\tilde{\boldsymbol{u}}_n + H\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H}), \\
&= (\boldsymbol{C}(\boldsymbol{K}(H) + \boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L} + 2\boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_n \\
&\quad - (\boldsymbol{C}\boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L} + \boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_{n-1} + \boldsymbol{C}e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n,
\end{aligned}
\tag{3.8}
$$

where

$$
\begin{aligned}
\boldsymbol{K}(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau, \\
\boldsymbol{J}(H) &= \int_{nH}^{(n+1)H} \frac{e^{\boldsymbol{A}((n+1)H-\tau)}(\tau - nH)}{H}d\tau.
\end{aligned}
$$

To combine above equations together:

$$
\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \\ \tilde{\boldsymbol{y}}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{S} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_n \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \end{bmatrix}. \tag{3.9}
$$

where

$$
\boldsymbol{S} = \begin{bmatrix} e^{\boldsymbol{A}H} & (\boldsymbol{K}(H) + \boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L} & -\boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L} \\ \boldsymbol{C}e^{\boldsymbol{A}H} & \boldsymbol{C}(\boldsymbol{K}(H) + \boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L} + 2\boldsymbol{D}\boldsymbol{L} & -(\boldsymbol{C}\boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L} + \boldsymbol{D}\boldsymbol{L}) \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{bmatrix}
$$

$\boldsymbol{O}, \boldsymbol{I} \in \mathbb{R}^{4\times 4}$

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

Remark. *In equation* (3.9)*, the extrapolation points are chosen at time instant* $(n-1)H$ *and* nH*. In fact, it is also possible to do it in a smaller time step, e.g.* $nH - h$ *and* nH*, where* h *is the micro step size.*

Assuming there is *Forward Euler* method involved in combination with linear ex-

trapolation, the discrete weakly coupled system can be written as:

$$\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= \tilde{\boldsymbol{z}}_n + \dot{\tilde{\boldsymbol{z}}}_n H, \\
&= \tilde{\boldsymbol{z}}_n + (\boldsymbol{A}\tilde{\boldsymbol{z}}_n + \boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}_n))H, \\
&= (\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + 2\boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_n - \boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_{n-1}, \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}(\tilde{\boldsymbol{u}}_n + H\frac{\boldsymbol{u}_n - \tilde{\boldsymbol{u}}_{n-1}}{H}), \\
&= \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + (2\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + 2\boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_n - (\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_{n-1}.
\end{aligned} \tag{3.10}$$

To combine above equations together:

$$\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \\ \tilde{\boldsymbol{y}}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} + \boldsymbol{A}H & 2\boldsymbol{B}\boldsymbol{L}H & -\boldsymbol{B}\boldsymbol{L}H \\ \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H) & 2\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + 2\boldsymbol{D}\boldsymbol{L} & -(\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L}) \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_n \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \end{bmatrix}, \tag{3.11}$$

where $\boldsymbol{O}, \boldsymbol{I} \in \mathbb{R}^{4\times 4}$.

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

3.2.3 Quadratic Extrapolation

Under quadratic extrapolation approximation:

$$\begin{aligned}
\Phi(\tilde{\boldsymbol{u}}(\tau)) = \tilde{\boldsymbol{u}}_n &+ (\tau - nH)\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H} \\
&+ \frac{(\tau - nH)(\tau - (n-1)H)}{2}\frac{\tilde{\boldsymbol{u}}_n - 2\tilde{\boldsymbol{u}}_{n-1} + \tilde{\boldsymbol{u}}_{n-2}}{H^2}, \quad \tau \in (nH, (n+1)H].
\end{aligned} \tag{3.12}$$

First let us solve the differential equation exactly by using *Variation of Constant Formula*. The discrete system can be written as:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}(\tau))d\tau, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\left(\tilde{\boldsymbol{u}}_n + (\tau - nH)\,\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H}\right. \\
&\quad \left. + \frac{(\tau - nH)\,(\tau - (n-1)\,H)}{2}\,\frac{(\tilde{\boldsymbol{u}}_n - 2\tilde{\boldsymbol{u}}_{n-1} + \tilde{\boldsymbol{u}}_{n-2})}{H^2}\right)d\tau, \\
&= e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + (\boldsymbol{K}(H) + \boldsymbol{J}(H) + \boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_n \\
&\quad - (\boldsymbol{J}(H) + 2\boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_{n-1} + \boldsymbol{G}(H)\boldsymbol{B}\boldsymbol{L}\tilde{\boldsymbol{y}}_{n-2}, \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}(\tilde{\boldsymbol{z}}_{n+1}) + \boldsymbol{D}\left(\tilde{\boldsymbol{u}}_n + H\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H} + \frac{H^2}{2}\frac{(\tilde{\boldsymbol{u}}_n - 2\tilde{\boldsymbol{u}}_{n-1} + \tilde{\boldsymbol{u}}_{n-2})}{H^2}\right), \\
&= (\boldsymbol{C}(\boldsymbol{K}(H) + \boldsymbol{J}(H) + \boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L} + \frac{5}{2}\boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_n \\
&\quad - ((\boldsymbol{J}(H) + 2\boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L} + 2\boldsymbol{D}\boldsymbol{L}))\tilde{\boldsymbol{y}}_{n-1} \\
&\quad + (\boldsymbol{G}(H)\boldsymbol{B}\boldsymbol{L} + \frac{1}{2}\boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_{n-2} + \boldsymbol{C}e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n,
\end{aligned}
\tag{3.13}
$$

where

$$
\begin{aligned}
\boldsymbol{K}(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau, \\
\boldsymbol{J}(H) &= \int_{nH}^{(n+1)H} \frac{e^{\boldsymbol{A}((n+1)H-\tau)}(\tau - nH)}{H}d\tau, \\
\boldsymbol{G}(H) &= \int_{nH}^{(n+1)H} \frac{e^{\boldsymbol{A}((n+1)H-\tau)}(\tau - nH)(\tau - (n-1)H)}{2H^2}d\tau.
\end{aligned}
$$

To combine above equations together:

$$
\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \end{bmatrix} = \begin{bmatrix} e^{\boldsymbol{A}H} & \boldsymbol{S}_{1,2} & \boldsymbol{S}_{1,3} & \boldsymbol{S}_{1,4} \\ \boldsymbol{C}e^{\boldsymbol{A}H} & \boldsymbol{S}_{2,2} & \boldsymbol{S}_{2,3} & \boldsymbol{S}_{2,4} \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_n \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \\ \tilde{\boldsymbol{y}}_{n-2} \end{bmatrix}, \tag{3.14}
$$

where

$$
\begin{aligned}
\boldsymbol{S}_{1,2} &= (\boldsymbol{K}(H) + \boldsymbol{J}(H) + \boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L}, \\
\boldsymbol{S}_{1,3} &= (\boldsymbol{J}(H) + 2\boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L}, \\
\boldsymbol{S}_{1,4} &= \boldsymbol{G}(H)\boldsymbol{B}\boldsymbol{L}, \\
\boldsymbol{S}_{2,2} &= \boldsymbol{C}(\boldsymbol{K}(H) + \boldsymbol{J}(H) + \boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L} + \frac{5}{2}\boldsymbol{D}\boldsymbol{L}, \\
\boldsymbol{S}_{2,3} &= -((\boldsymbol{J}(H) + 2\boldsymbol{G}(H))\boldsymbol{B}\boldsymbol{L} + 2\boldsymbol{D}\boldsymbol{L})), \\
\boldsymbol{S}_{2,4} &= \boldsymbol{G}(H)\boldsymbol{B}\boldsymbol{L} + \frac{1}{2}\boldsymbol{D}\boldsymbol{L}, \\
\boldsymbol{O} &\in \mathbb{R}^{4\times 4}, \quad \boldsymbol{I} \in \mathbb{R}^{4\times 4}.
\end{aligned}
$$

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

Assuming there is *Forward Euler* method involved in combination with quadratic extrapolation, the discrete weak coupled system can be written as:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= \tilde{\boldsymbol{z}}_n + \dot{\tilde{\boldsymbol{z}}}_n H, \\
&= \tilde{\boldsymbol{z}}_n + (\boldsymbol{A}\tilde{\boldsymbol{z}}_n + \boldsymbol{B}\Phi(\boldsymbol{u}_n))H, \\
&= (\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \boldsymbol{B}H\left(\tilde{\boldsymbol{u}}_n + H\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H} + \frac{H^2}{2}\frac{(\tilde{\boldsymbol{u}}_n - 2\tilde{\boldsymbol{u}}_{n-1} + \tilde{\boldsymbol{u}}_{n-2})}{H^2}\right), \\
&= (\boldsymbol{I} + \boldsymbol{A}H)\boldsymbol{z}_n + \frac{5}{2}\boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_n - 2\boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_{n-1} + \frac{1}{2}\boldsymbol{B}\boldsymbol{L}H\tilde{\boldsymbol{y}}_{n-2}. \\
\tilde{\boldsymbol{y}}_{n+1} &= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}_n), \\
&= \boldsymbol{C}\tilde{\boldsymbol{z}}_{n+1} + D\left(\tilde{\boldsymbol{u}}_n + H\frac{\tilde{\boldsymbol{u}}_n - \tilde{\boldsymbol{u}}_{n-1}}{H} + \frac{H^2}{2}\frac{(\tilde{\boldsymbol{u}}_n - 2\tilde{\boldsymbol{u}}_{n-1} + \tilde{\boldsymbol{u}}_{n-2})}{H^2}\right), \\
&= \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \frac{5}{2}(\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L})\boldsymbol{y}_n - (2\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + 2\boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_{n-1} \\
&\quad + \frac{1}{2}(\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L})\tilde{\boldsymbol{y}}_{n-2}.
\end{aligned}
\tag{3.15}
$$

To combine above equations together:

$$
\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{M}_{1,1} & \boldsymbol{M}_{1,2} \\ \boldsymbol{M}_{2,1} & \boldsymbol{M}_{2,2} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_n \\ \tilde{\boldsymbol{y}}_n \\ \tilde{\boldsymbol{y}}_{n-1} \\ \tilde{\boldsymbol{y}}_{n-2} \end{bmatrix} \tag{3.16}
$$

where

$$
\begin{aligned}
\boldsymbol{M}_{1,1} &= \begin{bmatrix} \boldsymbol{I} + \boldsymbol{A}H & \frac{5}{2}\boldsymbol{B}\boldsymbol{L}H \\ \boldsymbol{C}(\boldsymbol{I} + \boldsymbol{A}H) & \frac{5}{2}(\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L}) \end{bmatrix}, \\
\boldsymbol{M}_{1,2} &= \begin{bmatrix} -2\boldsymbol{B}\boldsymbol{L}H & \frac{1}{2}\boldsymbol{B}\boldsymbol{L}H \\ -(2\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + 2\boldsymbol{D}\boldsymbol{L}) & \frac{1}{2}(\boldsymbol{C}\boldsymbol{B}\boldsymbol{L}H + \boldsymbol{D}\boldsymbol{L}) \end{bmatrix}, \\
\boldsymbol{M}_{2,1} &= \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix}, \\
\boldsymbol{M}_{2,2} &= \begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{I} & \boldsymbol{O} \end{bmatrix}, \\
\boldsymbol{O}, \boldsymbol{I} &\in \mathbb{R}^{4\times 4}
\end{aligned}
$$

The weakly coupled system is stable, if the spectral radius of the matrix above is smaller than 1.

3.3 Numerical Simulation

As indicated Chapter 2, our research model is *force/displacement* coupling with *Jacobi* communication pattern, which means it is a zero-stable model. Thus in this section, we will present the stability and numerical stability property of our research model.

We would like to observe the change of the spectral radius in our co-simulation model under Constant, Linear and quadratic extrapolation. Let us define the parameters as:

k_1	100.0	N/m
d_1	100.0	Ns/m
m_1	100.0	kg
k_2	100.0	N/m
d_2	100.0	Ns/m
m_2	100.0	kg
k_c	60.0	N/m
d_c	60.0	Ns/m
$H = h$	$[10^{-5}, 1]$	s

Table 3.1: Parameters set-up 1

We can obtain the matrix representation from Chapter 2, plug the parameters into the matrices (3.4), (3.9) and (3.14) and obtain the spectral radius. First let us have a look at the spectral radius without any numerical methods involved:

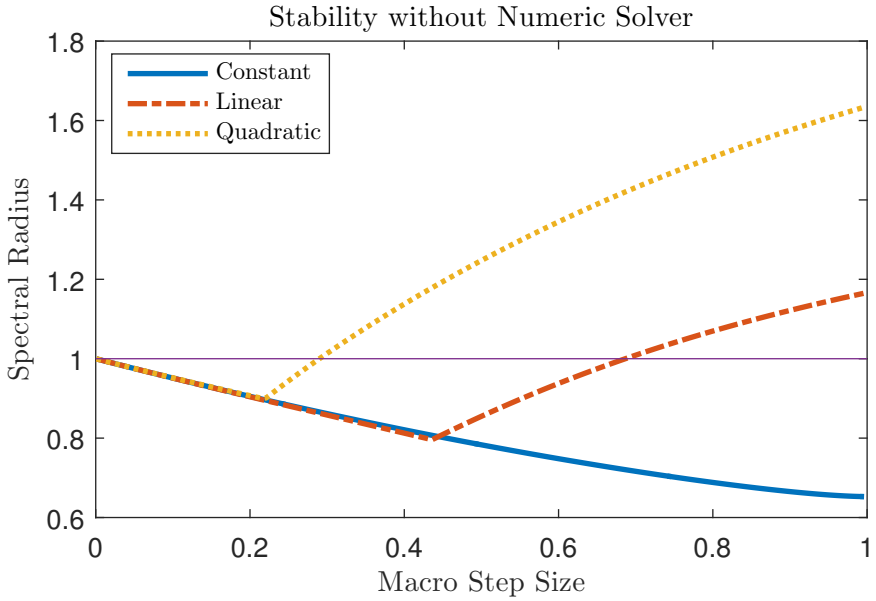

Figure 3.1: Spectral radius of weakly coupled system in combination of constant, linear and quadratic extrapolation

Assuming one uses *Forward Euler* in solving the differential equation, under same

parameter condition, plug the parameters into the matrices (3.6), (3.11), (3.16), the spectral radius looks like:

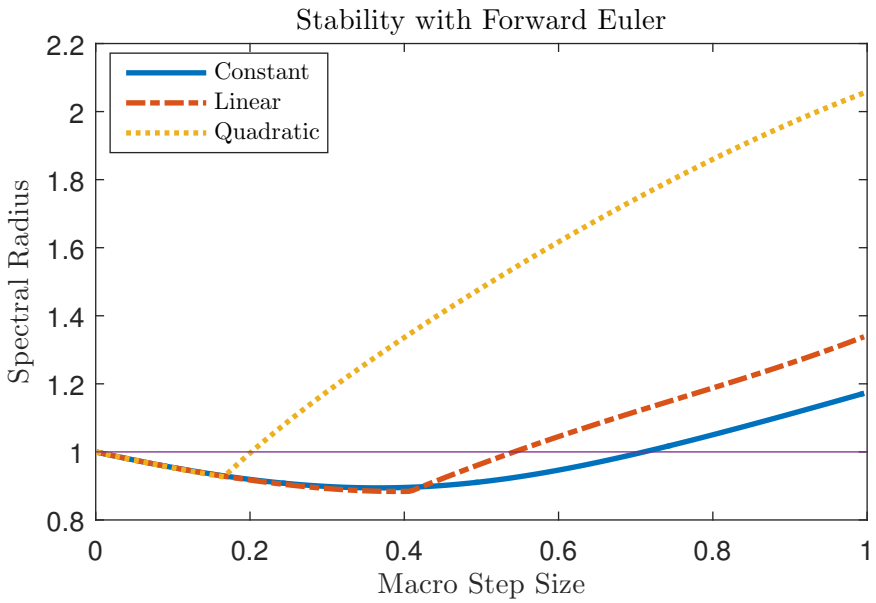

Figure 3.2: Spectral radius of weakly coupled system in combination of constant, linear and quadratic extrapolation 2

As expected, higher order extrapolation is more sensitive to step size changes, thus smaller stable region. *Forward Euler* is not a-stable method, therefore it works only in a small range of step size. In this case there are two unstable sources in weakly coupled system, extrapolation method and explicit numerical method. As we can see from Figure 3.1 and Figure 3.2, stability region further decreases correspondingly when there is explicit numerical method involved.

Remark. *The solution of the DE that is applied from an a-stable method with fixed step size,* $z_n \to 0$ *as* $n \to \infty$ *[8][7].*

3.4 Conclusions

The weakly coupled system presents different stability behaviour compare with the corresponding stable strongly coupled system. The stability of a weakly coupled dual mass spring damping system is strongly related to the input value extrapolation polynomial. Higher degree extrapolation polynomial is more sensitive to the change of step size. Stability region decreases as extrapolation degree increases, stability region further decreases if explicit numerical method is involved in co-simulation.

4

Weakly Coupled System Error Analysis

4.1 Introduction

Apart from stability analysis, accuracy is of high interest in numerical analysis as well. Generally speaking, there can be three kinds of error that affect the accuracy of the model:

- Round-off error
- Measurement error
- Approximation error

Round-off error occurs when the computer has limit digit on memory for calculation. All kinds of physical measurement for parameters lead to measurement error. Approximation error occurs in all kinds of approximation. In this chapter we neglect first two types of error, focus on the approximation error. For simplicity, we first analyse the global error without any numerical method. Then we compute the global error again with numerical method involved for comparison. Global error is usually of higher interest than local error, anyhow to reach global error one has to start from local error analysis.

4.2 Local Error

In classical local error analysis, it is assumed that in $[0, nH]$, the system is ideally exactly calculated. We focus only on the difference between the exact solution and the approximated solution in $(nH, (n+1)H]$ [9][27][28].

The exact solution can be obtained from the corresponding strongly coupled system, where the coupled variables are changed continuously:

$$\begin{aligned} \dot{\boldsymbol{z}} &= \boldsymbol{A}\boldsymbol{z} + \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{z} + \boldsymbol{D}\boldsymbol{u}, \\ \boldsymbol{u} &= \boldsymbol{L}\boldsymbol{y}. \end{aligned} \tag{4.1}$$

Strongly coupled state value at time $(n+1)H$ is denoted as $\boldsymbol{z}_{n+1}$. By solving the DE in (4.1) analytically we obtain:

$$\boldsymbol{z}_{n+1} = e^{\boldsymbol{A}H}\boldsymbol{z}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}u(\tau)d\tau. \tag{4.2}$$

The approximated solution is the solution that obtain from weakly coupled system with certain degrees of extrapolation polynomial:

$$\begin{aligned} \dot{\tilde{\boldsymbol{z}}} &= \boldsymbol{A}\tilde{\boldsymbol{z}} + \boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}), \\ \tilde{\boldsymbol{y}} &= \boldsymbol{C}\tilde{\boldsymbol{z}} + \boldsymbol{D}\Phi(\tilde{\boldsymbol{u}}), \\ \tilde{\boldsymbol{u}} &= \boldsymbol{L}\tilde{\boldsymbol{y}}. \end{aligned} \tag{4.3}$$

Weakly coupled state value with certain degree of polynomial approximation at time $(n+1)H$ is denoted as $\tilde{\boldsymbol{z}}_{n+1}$. By solving the DE in equation (4.3) exactly we obtain:

$$\tilde{\boldsymbol{z}}_{n+1} = e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}(\tau))d\tau. \tag{4.4}$$

Local state error at time $(n+1)H$ is denoted as $\varepsilon(\boldsymbol{z}_{n+1})$. Following the classical local error analysis in numerical analysis, we compute local error as:

$$\begin{aligned} \varepsilon(\boldsymbol{z}_{n+1}) &= \boldsymbol{z}_{n+1} - \tilde{\boldsymbol{z}}_{n+1}, \\ &= \boldsymbol{z}_{n+1} - (e^{\boldsymbol{A}H}\boldsymbol{z}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\boldsymbol{u}(\tau))d\tau), \\ &= e^{\boldsymbol{A}H}\boldsymbol{z}_n + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}u(\tau)d\tau \\ &\quad - e^{\boldsymbol{A}H}\boldsymbol{z}_n - \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}\Phi(\boldsymbol{u}(\tau))d\tau, \\ &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}\boldsymbol{B}(\boldsymbol{u}(\tau) - \Phi(\boldsymbol{u}(\tau)))d\tau, \\ &\leq \int_{nH}^{(n+1)H} e^{\|\boldsymbol{A}((n+1)H-\tau)\|}\boldsymbol{B}d\tau \max_{nH\leq\tau\leq(n+1)H}\|\boldsymbol{u}(\tau) - \Phi(\boldsymbol{u}(\tau))\|d\tau, \\ &\leq^{*} \mathcal{O}(H)\boldsymbol{B}\mathcal{O}(H^{k+1}), \\ &\leq \mathcal{O}(H^{k+2}), \end{aligned} \tag{4.5}$$

where, $\leq^{*}$ holds if $\|\boldsymbol{A}\| \leq c$, and $c \in \mathbb{N}$, and k is the degree of the extrapolation operator $\Phi(u)$.

Here we obtain the local error for state value $\varepsilon(\boldsymbol{z})$ is $\mathcal{O}(H^{k+2})$, in both *displacement/displacement* and *force/displacement* weakly coupled systems.

Next, we will have an insight into the local error in terms of output value. Use notation $\varepsilon(y_{n+1})$ for local output error at time $(n+1)H$. Output equations in state space systems (4.1) and (4.3) are linear equations, thus:

$$\varepsilon(\boldsymbol{y}_{n+1}) = \boldsymbol{C}\varepsilon(\boldsymbol{z}_{n+1}) + \boldsymbol{D}\varepsilon(\boldsymbol{u}_{n+1}), \tag{4.6}$$

where $\varepsilon(\boldsymbol{u}_{n+1})$ relates to the input extrapolation degree.

In *force/displacement* coupled system, we have:

$$\varepsilon(\boldsymbol{y}_{n+1}) = \boldsymbol{C}\mathcal{O}(H^{k+2}) + \boldsymbol{D}\mathcal{O}(H^{k+1}) = \mathcal{O}(H^{k+1}). \tag{4.7}$$

In this case, there is order reduction on local output error, due to the presence of direct feed-through in one of the subsystems:

$$\begin{aligned} \varepsilon(\boldsymbol{z}_{n+1}) &= \mathcal{O}(H^{k+2}), \\ \varepsilon(\boldsymbol{u}_{n+1}) &= \mathcal{O}(H^{k+1}), \\ \varepsilon(\boldsymbol{y}_{n+1}) &= \mathcal{O}(H^{k+1}), \end{aligned} \tag{4.8}$$

In *displacement/displacement* coupled system, we have:

$$\begin{aligned} \varepsilon(\boldsymbol{z}_{n+1}) &= \mathcal{O}(H^{k+2}), \\ \varepsilon(\boldsymbol{u}_{n+1}) &= \mathcal{O}(H^{k+1}), \\ \varepsilon(\boldsymbol{y}_{n+1}) &= \mathcal{O}(H^{k+2}), \end{aligned} \tag{4.9}$$

In this case, local error w.r.t the state and the output have the same order, due to nondirect feed-through is every subsystem.

Calculations and conclusions above are obtained by assuming that the previous time steps before the local error step are error-free. However, that is not true in practise. In the following section, we will have a look at the global error which has more practical meaning.

4.3 Global Error

Global error has more practical meaning than local error. However, it is usually very difficult to obtain global error directly. One way to do it is to derive local error recursively, and this is the approach we will do in this section.

4.3.1 Global Error without Numerical Method

Use notation $E(\boldsymbol{z}_n)$ for the global state error at time nH. Since the global error is the accumulation of all local error, we can not neglect the error from the previous time steps as we did in local error analysis:

$$
\begin{aligned}
E(\boldsymbol{z}_n) &= \boldsymbol{z}_n - \tilde{\boldsymbol{z}}_n \\
&= e^{\boldsymbol{A}H}\boldsymbol{z}_{n-1} + \int_{(n-1)H}^{nH} e^{\boldsymbol{A}(nH-\tau)}\boldsymbol{B}\boldsymbol{u}(\tau)d\tau \\
&\quad - e^{\boldsymbol{A}H}\tilde{\boldsymbol{z}}_{n-1} - \int_{(n-1)H}^{nH} e^{\boldsymbol{A}(nH-\tau)}\boldsymbol{B}\Phi(\tilde{\boldsymbol{u}}(\tau))d\tau \\
&= e^{\boldsymbol{A}H}E(\boldsymbol{z}_{n-1}) + \int_{(n-1)H}^{nH} e^{\boldsymbol{A}(nH-\tau)}\boldsymbol{B}(\boldsymbol{u}(\tau) - \Phi(\boldsymbol{u}(\tau)) + \Phi(\boldsymbol{u}(\tau)) - \Phi(\tilde{\boldsymbol{u}}(\tau)))d\tau \\
&\leq \|e^{\boldsymbol{A}H}E(\boldsymbol{z}_{n-1})\| + \int_{(n-1)H}^{nH} e^{\boldsymbol{A}(nH-\tau)}\boldsymbol{B} \max_{nH\leq\tau\leq(n+1)H} \|(\Phi(\boldsymbol{u}(\tau)) - \Phi(\tilde{\boldsymbol{u}}(\tau)))\| + \mathcal{O}(H^{k+2}) \\
&\quad \ldots\ldots \\
&\leq^* \mathcal{O}(H)\mathcal{O}(H^{k+1})\frac{e^{n\boldsymbol{A}H} - 1}{\mathcal{O}(H)} + \mathcal{O}(H^{k+2}) \\
&\leq \mathcal{O}(H^{k+1})
\end{aligned}
\tag{4.10}
$$

where $\leq^*$ holds if $\|\boldsymbol{A}\| \leq c$, and $c \in \mathbb{N}$

Following the similar calculation like equation (4.10), we can obtain global output error at time nH denoted as $E(\boldsymbol{y}_n)$ as follow:

$$E(\boldsymbol{y}_n) = E(\boldsymbol{z}_n) = \mathcal{O}(H^{k+1}),$$

which means there is no order reduction in global error. This is true for weakly coupled system without algebraic loop, (e.g. *displacement/displacement* coupled system and *force/displacement* weakly coupled system) [3].

Simulation results of the global error with different extrapolation methods have been shown:

k_1	100.0	N/m
d_1	100.0	Ns/m
m_1	100.0	kg
k_2	100.0	N/m
d_2	100.0	Ns/m
m_2	100.0	kg
k_c	60.0	N/m
d_c	60.0	Ns/m
$H = h$	$[10^{-3}, 10^{-2}]$	s

Table 4.1: Parameters set up-2

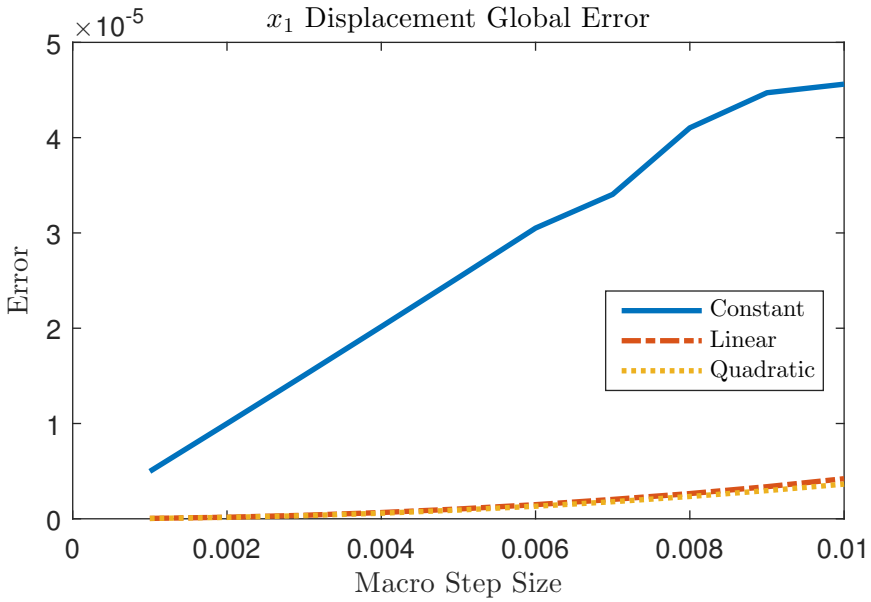

Figure 4.1: x_1 displacement global error order

Since the global error in constant extrapolation case is several order higher than the global error in linear and quadratic extrapolation cases. Therefor,we we neglect the constant extrapolation case to have a closer look in linear and quadratic extrapolation cases.

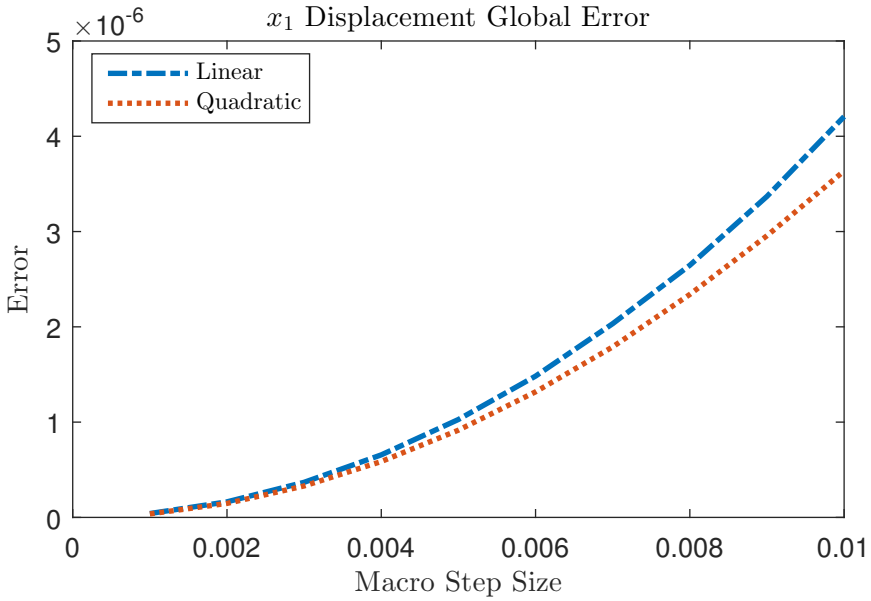

Figure 4.2: x_1 displacement global error order 2

From Figure 4.1 and Figure 4.2 it is clear that the global error value decreases as the extrapolation degree increases. And the global error increases if the step size increases. The simulation results coincide with our global error analysis.

4.3.2 Global Error with Numerical Method

By adding numerical methods in co-simulation, more errors are propagated than without numerical methods. To classify different error sources, we call error that is propagated by coupling (without the usage of numerical method) as coupling error, which we have shown in the last subsection. The error that is emerged by numerical method named as numerical error. For theoretical research we are interested in the error emerged by co-simulation. But from industrial point of view, how the numerical error involved in co-simulation has to be discussed as well.

We first conclude the global error with numerical method involved in co-simulation as the following lemma, then we will show the proof of it.

Lemma 4.3.1. *Given extrapolation polynomial of degree k, numerical method of order j, and micro step equal to macro step then one can have global error for co-simulation as:*

$$\begin{aligned} E(\boldsymbol{z}_n) &\leq \mathcal{O}(H^j) + \mathcal{O}(H^{k+1}) \\ E(\boldsymbol{y}_n) &\leq \mathcal{O}(H^j) + \mathcal{O}(H^{k+1}) \end{aligned} \tag{4.11}$$

Proof. First let us try to fix one numerical method but not the degree of extrapolation polynomial, then we will conclude to other numerical methods.

Suppose we choose *Forward Euler* as the local solver:

$$\begin{aligned} \tilde{\boldsymbol{z}}_{n+1} &= \tilde{\boldsymbol{z}}_n + \dot{\tilde{\boldsymbol{z}}}_n * H \\ &= (I + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \boldsymbol{B}H\Phi(\tilde{\boldsymbol{u}}_n). \end{aligned} \tag{4.12}$$

And the exact solution can be obtained by applying the Taylor series:

$$\begin{aligned} \boldsymbol{z}_{n+1} &= \boldsymbol{z}_n + \dot{\boldsymbol{z}}_n * H + \mathcal{O}(H^2) \\ &= (I + \boldsymbol{A}H)\boldsymbol{z}_n + \boldsymbol{B}H\boldsymbol{u}_n + \mathcal{O}(H^2). \end{aligned} \tag{4.13}$$

We have local error as:

$$\begin{aligned} \varepsilon(\boldsymbol{z}_{n+1}) &= \boldsymbol{z}_{n+1} - \tilde{\boldsymbol{z}}_{n+1} \\ &= (I + \boldsymbol{A}H)\boldsymbol{z}_n + \boldsymbol{B}H\boldsymbol{u}_n - ((I + \boldsymbol{A}H)\boldsymbol{z}_n + \boldsymbol{B}H\Phi(\boldsymbol{u})_n) + \mathcal{O}(H^{\cdot}) \\ &= \boldsymbol{B}H(u_n - \Phi(\boldsymbol{u})_n) + \mathcal{O}(H^2) \\ &\leq^* \mathcal{O}(H^{k+2}) + \mathcal{O}(H^2), \end{aligned} \tag{4.14}$$

where $\leq^*$ holds if $\|\boldsymbol{B}\| \leq c$, $c \in \mathbb{N}$.

We obtain global error by recursion:

$$
\begin{aligned}
E(\boldsymbol{z}_{n+1}) &= \boldsymbol{z}_{n+1} - \tilde{\boldsymbol{z}}_{n+1} \\
&= (I + \boldsymbol{A}H)\boldsymbol{z}_n + \boldsymbol{B}Hu_n - (I + \boldsymbol{A}H)\tilde{\boldsymbol{z}}_n + \boldsymbol{B}H\Phi(\tilde{\boldsymbol{u}})_n) + \mathcal{O}(H^2) \\
&= (I + \boldsymbol{A}H)E(\boldsymbol{z}_n) + \boldsymbol{B}H(\boldsymbol{u}_n - \Phi(\boldsymbol{u})_n + \Phi(\boldsymbol{u})_n - \Phi(\tilde{\boldsymbol{u}})_n) + \mathcal{O}(H^2) \\
&\leq \|(I + \boldsymbol{A}H)E(\boldsymbol{z}_n)\| + \boldsymbol{B}H\mathcal{O}(H^{k+1}) + \|\Phi(\boldsymbol{u})_n - \Phi(\tilde{\boldsymbol{u}})_n\| + \boldsymbol{B}H\mathcal{O}(H^{k+1}) + \mathcal{O}(H^2) \\
&\leq ..... \\
&\leq (I + \boldsymbol{A}H)E(\boldsymbol{z}_0) + [I + (I + \boldsymbol{A}H) + ... + (I + \boldsymbol{A}H)^n](\boldsymbol{B}H\mathcal{O}(H^{k+1}) + \mathcal{O}(H^2) \\
&\leq^* \frac{(I + cH)^n - I}{cH}(\boldsymbol{B}H\mathcal{O}(H^{k+1})) + \mathcal{O}(H^2) \\
&\leq \mathcal{O}(H^{k+1}) + \mathcal{O}(H^1),
\end{aligned}
\tag{4.15}
$$

where $\leq^*$ holds if $\|\boldsymbol{A}\| \leq c$, $c \in \mathbb{N}$.

Numerical methods typically proceed by truncating the Taylor series at some point, arguing that the error that is introduced by doing so is small enough to be unimportant [25]. Therefore we can conclude the global error order to other numerical methods as the following:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{n+1} &= \phi(\sum_{i=n-j}^{n} \tilde{\boldsymbol{z}}_i, j, H, \boldsymbol{A}) + \Phi(\sum_{l=n-k}^{n} \tilde{\boldsymbol{u}}_l, k, H, \boldsymbol{B}) \\
\boldsymbol{z}_{n+1} &= \phi(\sum_{i=n-j}^{n} z_i, j, H, \boldsymbol{A}) + \Phi(\sum_{l=n-k}^{n} u_l, k, H, \boldsymbol{B}) + \mathcal{O}(H^{k+2}) + \mathcal{O}(H^{j+1}) \\
\varepsilon(\boldsymbol{z}_{n+1}) &= \boldsymbol{z}_{n+1} - \tilde{\boldsymbol{z}}_{n+1} \\
&\leq \mathcal{O}(H^{j+1}) + \mathcal{O}(H^{k+2}) \\
E(\boldsymbol{z}_{n+1}) &= z_{n+1} - \tilde{\boldsymbol{z}}_{n+1} \\
&\leq \mathcal{O}(H^j) + \mathcal{O}(H^{k+1}).
\end{aligned}
\tag{4.16}
$$

where notation ϕ is the numerical method of order j, notation Φ is the extrapolation operator of degree k.

And we know there is no order reduction in global error [3]. Therefore, we can generate immediately or follow the same proof, and obtain:

$$
E(\boldsymbol{y}_n) \leq \mathcal{O}(H^j) + \mathcal{O}(H^{k+1}). \tag{4.17}
$$

□

Remark. *For global error in co-simulation where $H \neq h$, similar analysis processes can be done and obtain:*

$$
\begin{aligned}
E(\boldsymbol{z}_n) &\leq \mathcal{O}(h^j) + \mathcal{O}(H^{k+1}), \\
E(\boldsymbol{y}_n) &\leq \mathcal{O}(h^j) + \mathcal{O}(H^{k+1}),
\end{aligned}
\tag{4.18}
$$

where $h = max h_1, h_2, ...h_m$ *is the largest micro step size in all the subsystems. In usual case:* $H \geq h$, $k \in [0, 2]$ *and* $j \in [1, 6]$, *the error is dominated by the second term or we call it coupling error.*

4.4 Conclusions

In local error analysis, output error experiences error order reduction due to the presence of direct feed-through in *force/displacement* weakly coupled system. However, in global error case there is no error order reduction in our model. Global error is bounded by the summation of numerical error and coupling error. In usual case, global error is dominated by coupling error, which makes it effective to reduce the global error by an appropriate extrapolation method or by more robust interface and causality design.

5

Interface and Causality Design

5.1 Introduction

In *force/displacement* weakly coupled system there is a matter of causality, whether the force should be placed as input or output. The scope of this chapter is to give a preferable design for causality and interface preference, by comparing the weakly coupled solution and the strongly coupled solution.

5.2 Interface and Causality Design

In strongly coupled system the input and output variables exchange continuously. Therefore, it can be written in form of:

$$\begin{aligned} \dot{\boldsymbol{z}} &= \boldsymbol{A}\boldsymbol{z} + \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{z} + \boldsymbol{D}\boldsymbol{u}, \\ \boldsymbol{u} &= \boldsymbol{L}\boldsymbol{y}. \end{aligned} \tag{5.1}$$

Analytical solution for the strongly coupled system can be written as:

$$\begin{aligned} \boldsymbol{z}_{n+1} &= e^{(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{L}(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C})H}\boldsymbol{z}_n \\ \boldsymbol{y}_{n+1} &= (\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C}\boldsymbol{z}_{n+1} \end{aligned} \tag{5.2}$$

where matrix $(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}$ is assumed to be non-singular.

Rewrite equation (5.2) in matrix form:

$$\begin{bmatrix} \boldsymbol{z}_{n+1} \\ \boldsymbol{y}_{n+1} \end{bmatrix} = \begin{bmatrix} e^{(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{L}(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C})H} & \boldsymbol{O} \\ (\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C}e^{(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{L}(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C})H} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_n \\ \boldsymbol{y}_n \end{bmatrix} \tag{5.3}$$

Assuming that the input value is approximated by the linear extrapolation polynomial, we obtain the weakly coupled solution from equation (3.4):

$$\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \end{bmatrix} = \begin{bmatrix} e^{\boldsymbol{A}H} & \boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L} \\ Ce^{\boldsymbol{A}H} & \boldsymbol{C}\boldsymbol{K}(H)\boldsymbol{B}\boldsymbol{L}+\boldsymbol{D}\boldsymbol{L} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_n \\ \tilde{\boldsymbol{y}}_n \end{bmatrix} \tag{5.4}$$

where

$$\boldsymbol{K}(H) = \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau$$

Intuitively we want to have the weakly coupled model as close as possible to strongly coupled model , which means $\tilde{\boldsymbol{z}}(t) \approx \boldsymbol{z}(t)$, $\forall t \in \mathbb{T}$, where $\mathbb{T}$ represents total simulation time set. By comparing equation (5.4) and equation (5.3), the weakly coupled solution and strongly coupled solution will be roughly 'equal',

if

$$\begin{aligned} \boldsymbol{DL} &\approx \boldsymbol{O} \\ \boldsymbol{BL} &\approx \boldsymbol{O} \end{aligned} \tag{5.5}$$

or

$$\begin{aligned} \boldsymbol{K}(H) &\approx \boldsymbol{O} \\ \boldsymbol{DL} &\approx \boldsymbol{O} \end{aligned} \tag{5.6}$$

or

$$\begin{aligned} \boldsymbol{DL} &\approx \boldsymbol{O} \\ \boldsymbol{BL} &\approx \boldsymbol{O} \\ \boldsymbol{K}(H) &\approx \boldsymbol{O} \end{aligned} \tag{5.7}$$

The above possibilities can be further extended for the parameters study based on our research model:

$$\boldsymbol{BL} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{k_c}{m_1} & \frac{d_c}{m_1} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m_2} & 0 & 0 \end{bmatrix} \tag{5.8}$$

$$\boldsymbol{DL} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -k_c & -d_c \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.9}$$

If k_c and d_c is relative small, and m_2 is relatively large, $\boldsymbol{DL}$ and $\boldsymbol{BL}$ are nearly zero matrices.

Then we analyse the second possibility,

$$\begin{aligned} \boldsymbol{K}(H) &\approx 0, \\ \boldsymbol{DL} &\approx \boldsymbol{O} \end{aligned} \tag{5.10}$$

it is equivalent to:

$$\begin{aligned} \boldsymbol{K}(H) \approx \boldsymbol{O} &\implies H \approx 0, \\ \boldsymbol{DL} \approx \boldsymbol{O} &\implies k_c \approx d_c \approx \textbf{small value}. \end{aligned} \tag{5.11}$$

Therefore, we would like to have k_c and d_c as small value at the interface and small global time step size. The result is very trivial. If each subsystems communicates

in high frequency, the weakly coupled system behaves like a strongly coupled system.

For the third possibility, it is very straight forward. As it combines all the requirements from first and second possibilities.

What if one increase the degree of input value extrapolation? Assuming input value is approximated by linear extrapolation polynomial, we obtain weakly coupled solution from equation (3.9):

$$\begin{bmatrix} \tilde{\boldsymbol{z}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n+1} \\ \tilde{\boldsymbol{y}}_{n} \end{bmatrix} = \begin{bmatrix} e^{\boldsymbol{A}H} & (\boldsymbol{K}(H)+\boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L} & -\boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L} \\ \boldsymbol{C}e^{\boldsymbol{A}H} & \boldsymbol{C}(\boldsymbol{K}(H)+\boldsymbol{J}(H))\boldsymbol{B}\boldsymbol{L}+2\boldsymbol{D}\boldsymbol{L} & -C\boldsymbol{J}(H)\boldsymbol{B}\boldsymbol{L}-\boldsymbol{D}\boldsymbol{L} \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{z}}_{n} \\ \tilde{\boldsymbol{y}}_{n} \\ \tilde{\boldsymbol{y}}_{n-1} \end{bmatrix} \tag{5.12}$$

where

$$\begin{aligned} \boldsymbol{K}(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}((n+1)H-\tau)}d\tau \\ \boldsymbol{J}(H) &= \int_{nH}^{(n+1)H} \frac{e^{\boldsymbol{A}((n+1)H-\tau)}\tau}{H}d\tau. \end{aligned} \tag{5.13}$$

In order to compare with strongly coupled system, we rewrite equation (5.3) as:

$$\begin{bmatrix} \boldsymbol{z}_{n+1} \\ \boldsymbol{y}_{n+1} \\ \boldsymbol{y}_{n} \end{bmatrix} = \begin{bmatrix} e^{(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{L}(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C})H} & \boldsymbol{O} & \boldsymbol{O} \\ (I-DL)^{-1}\boldsymbol{C}e^{(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{L}(\boldsymbol{I}-\boldsymbol{D}\boldsymbol{L})^{-1}\boldsymbol{C})H} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{z}_{n} \\ \boldsymbol{y}_{n} \\ \boldsymbol{y}_{n-1} \end{bmatrix} \tag{5.14}$$

Again,the above two matrices will be roughly identical element-wise, if

$$\begin{aligned} \boldsymbol{D}\boldsymbol{L} &\approx \boldsymbol{O} \\ \boldsymbol{B}\boldsymbol{L} &\approx \boldsymbol{O} \end{aligned} \tag{5.15}$$

or

$$\begin{aligned} \boldsymbol{K}(H) &\approx \boldsymbol{O} \\ \boldsymbol{J}(H) &\approx \boldsymbol{O} \\ \boldsymbol{D}\boldsymbol{L} &\approx \boldsymbol{O} \end{aligned} \tag{5.16}$$

or

$$\begin{aligned} \boldsymbol{D}\boldsymbol{L} &\approx \boldsymbol{O} \\ \boldsymbol{B}\boldsymbol{L} &\approx \boldsymbol{O} \\ \boldsymbol{K}(H) &\approx \boldsymbol{O} \\ \boldsymbol{J}(H) &\approx \boldsymbol{O} \end{aligned} \tag{5.17}$$

As we can see that equation (5.8) coincides with equation (5.18), equation (5.9) coincides with equation (5.19), equation (5.10) coincides with Equation (5.20).

Therefore, we would like to have rather soft interface, and force applied to the subsystem which has larger mass value. This results can be extended to higher extrapolation degree and multi-subsystems co-simulation.

5.3 Simulation

In this section we will validate three different possibilities. Overall speaking, all the possibilities require that interface has to be chosen where the spring and damper has relative small value. Following the notation in our research model it would be k_c and d_c has relative small value. In the following simulation we process quasi-experiments, which means each time only one variable is changed. Beside we always compare the ratio of weakly coupled and strongly coupled simulation result correspondingly. Thus the closer the ratio to 1, the better the simulation result we can get.

To see the effect when the interface is getting more stiff, we choose step size $H = h = 0.01$ s and define the initial value for the parameters as Table 3.1 with varying parameters $k_c \in$[60.0, 120.0] N/m and $d_c \in$ [60.0, 120.0] Ns/m.

Insert the given parameters set-up into equation (4.2) and (4.4), and compute the ratio of weakly coupled solution and strongly coupled solution in each simulation time:

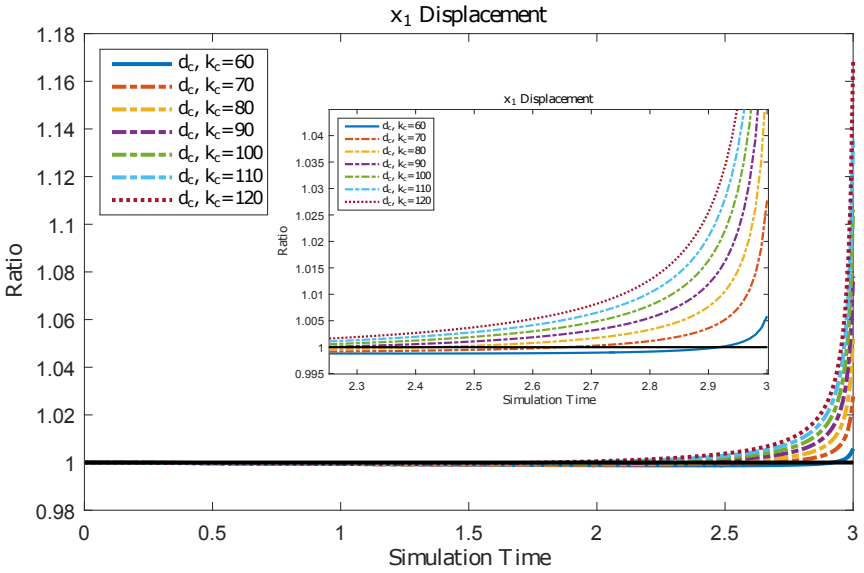

Figure 5.1: Increase the value of spring and damper at the interface

As we can see from Figure 5.1, the increase of spring and damper value at the interface has a obvious effect on the weakly coupled simulation results. As simulation time pass, the traces diverge from value 1. But the model with the softest interface delivers a rather robust trace among all other models.

Next we validate if force should go to the subsystem that is heavier. We choose step size $H = h = 0.01$ s and define the initial value for the parameters as Table 3.1 with varying parameter $m_2 \in$[100.0,400.0] kg:

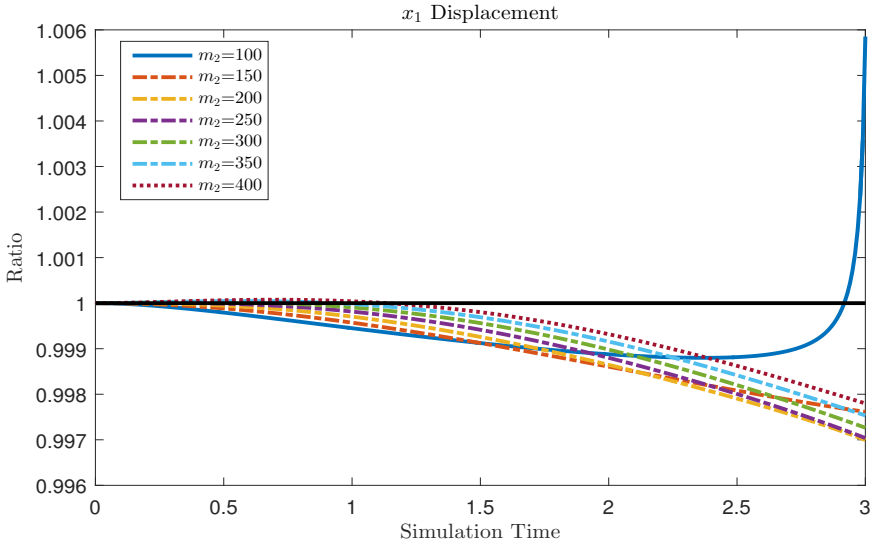

Figure 5.2: increase the mass value m_2

From Figure 5.2 we can see that, the increase value of m_2, makes the ratio getting closer to value 1, which means more accurate. The result coincides with our assumption.

Then we validate the effect when the step size increase. We choose step size $H = h = 0.01$ s and define the initial value for the parameters as Table 3.1 by increase the step size in each simulation:

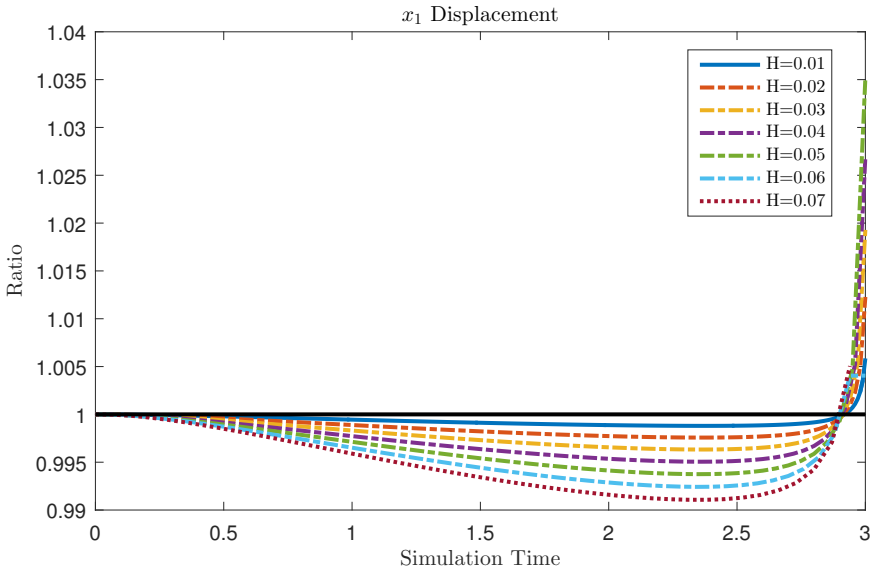

Figure 5.3: Increase macro step size

As we can see from the Figure 5.3, the smaller macro step is, the closer trace is to value 1. The simulation results coincide with our assumption. The result is trivial to see, as small macro step size means more frequent communication between subsystems, and this is the reason for accuracy. We can see this from Chapter 4 as well, the error order is strongly related to step size and extrapolation order. Since we fixed the simulation to constant extrapolation, then the only difference left is step size.

The third assumption is the combination of the first two assumptions. We define the initial value for the parameters as Table 3.1 but with the step size $H = h = 0.01$ s, the mass value $m_2 = 400$ kg and increase the step size in each simulation:

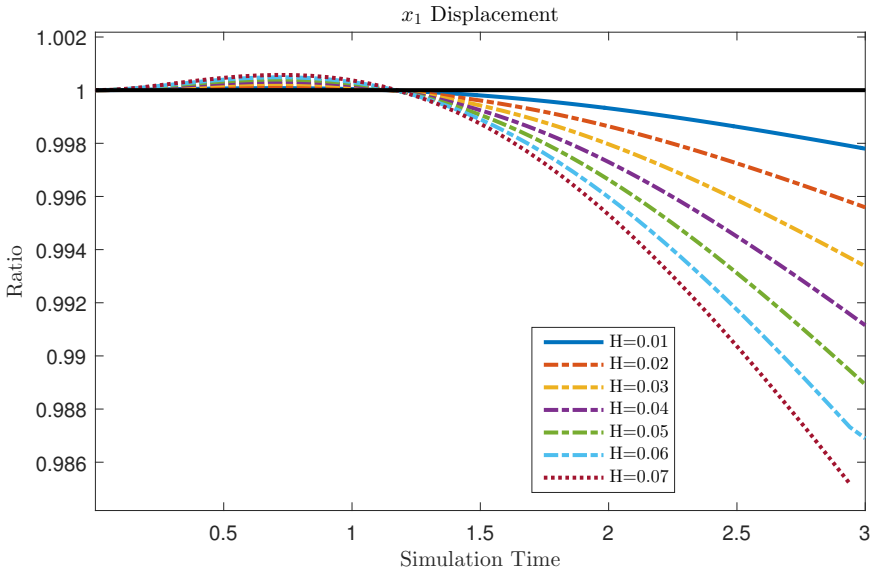

Figure 5.4: Increase macro step size

Obviously co-simulation with the smallest macro step has the best performance,the simulation results coincide with our previous assumptions. If we try to compare Figure 5.3 and Figure 5.4, under same macro step size, each trace in Figure 5.4 is closer to value 1 than the trace in Figure 5.3 respectively. Because the third assumption combine all the good parameter setting in the first and the second assumptions.

5.4 Conclusions

Interface should be always chosen in the part that it is soft, which means small value in stiffness and damping coefficients. Force is placed as input to the subsystem which weights more. Macro step size is always preferable to be small. But all these parameter can not be infinitely large or small, there is some kinds of physical

connection between the parameters. For instance, the coupled systems lost the sense of coupling, when the coefficients at the interface are set to be infinitely small.

6

Evaluation of Nearly Energy Preserving Coupling Element

6.1 Introduction

In weakly coupled co-simulation, it is very difficult or impossible to have an insight look into the state value or even the state space equations in different simulators. The reasons for this can be the models or the calculation algorithms are part of the intellectual property [13]. Therefore, we have a lack of a reference to compare with the weakly coupled co-simulation for error estimation. One way to find the reference is to take a relative small step size, compute the solution interactively till the error is lower than the predefined tolerance as the reference for weakly coupled system. This requires the rollback ability for the simulators, thus it is not applicable in practice. *Nearly Energy Preserving Coupling Element* (NEPCE) method [21] claimed that, it can be used as an error indicator without the knowledge about the dynamics inside the simulators and rollback ability. In this chapter, we first introduce the idea that NEPCE method came from, then explain the method, lastly evaluate the method based on our research model.

6.2 Energy Conservation and Power Bond

The usage of energy conservation and power bond in NEPCE came from the bond graph theory [5][20]. It states that the energy transactions between two systems as the general energy continuity equation:

$$\frac{\partial \epsilon(x,t)}{\partial t} + \Delta j_\epsilon(x,t) = \sigma_\epsilon(x,t) \tag{6.1}$$

where $\epsilon(x,t)$ is the local energy density at time t position x, j_ϵ is the energy flux through the surface, and σ_ϵ is the energy dissipation rate.

The general equation of energy continuity states that a net flux of energy through the surface is either stored or dissipated. It is designed to represent the behaviour of energy, power, entropy and other physical properties. This reflects the energy conservation. And the energy transactions between two systems are seen as power bonds [20][5].

6.3 Nearly Energy Preserving Coupling Element

NEPCE method claimed to describe the performance for non-interactive co-simulation base on the input value extrapolation violation of the (generalized) energy conservation law [21].

The idea of NEPCE is to apply the power bond theory and energy conservation law in weakly coupled mechanical systems. Assuming there are two weakly coupled subsystems S_1 and S_2, between each subsystem there are input u_i and output y_i for $i = 1, 2$. Two subsystems exchange energy through a power bond k, at a rate P_{ki}. Power bond P_{ki} and energy E_i for $i = 1, 2$ are defined as follow:

$$\begin{aligned} P_{ki}(t) &= \boldsymbol{u}_i(t)\boldsymbol{y}_i(t) \\ E_i(t) &= \int P(t)dt \end{aligned} \tag{6.2}$$

No energy should be dissipated or stored by general equation of energy continuity:

$$-\left(P_{k1} + P_{k2}\right) = 0. \tag{6.3}$$

However, input variables are generally unknown between each macro time step point and extrapolated input value $\Phi(\tilde{u}(t)) \approx u(t)$ has to be used. Denote power bond rate in this case as $\tilde{P}_k$:

$$\begin{aligned} \tilde{P}_{ki}(t) &= \Phi(\tilde{\boldsymbol{u}}_i(t))\tilde{\boldsymbol{y}}_i(t) \\ \tilde{P}_{ki}(t) &\neq P_{ki}(t) \end{aligned} \tag{6.4}$$

It is straight forward to see that the weakly coupled system violates energy conservation due the extrapolated input value [21][22]. Here we use the notation $\delta P(t)$ and $\delta E(t)$ to represent the residual power and residual energy respectively, and $\delta P(t) \neq 0$ as $\tilde{u} \neq u$.

$$\begin{aligned} \delta P(t) &= -\left(\tilde{P}_{k1}\left(t\right) + \tilde{P}_{k2}\left(t\right)\right) \\ \delta E(t) &= \int \delta\tilde{P}(t)dt \end{aligned} \tag{6.5}$$

S. Sadjina et.al.[21] believes that if co-simulation model follows energy conservation law, residual power will be zero [21] . But this does not suffice in general, energy either leaks from ($\delta P_k < 0$) or accumulates at ($\delta P_k > 0$) the power bond. Therefore, it is concluded that residual energy as error indicator [21].

Considering the time dependent states $\boldsymbol{z} = [z_1, z_2]$, the discrete weakly coupled systems can be expressed as:

$$\begin{aligned} \dot{\tilde{\boldsymbol{z}}}_n &= \boldsymbol{f}(\tilde{\boldsymbol{z}}_n, \Phi(\tilde{\boldsymbol{u}}_n)), \\ \tilde{\boldsymbol{y}}_n &= \boldsymbol{g}(\tilde{\boldsymbol{z}}_n, \Phi(\tilde{\boldsymbol{u}}_n)) \\ \tilde{\boldsymbol{u}}_n &= \boldsymbol{L}\tilde{\boldsymbol{y}}_n \end{aligned} \tag{6.6}$$

The concept behind NEPCE method is to find corrections δu for the input at each communication time step to reduces the residual energy. Thus the most ideal correction would be:

$$\delta \boldsymbol{u}(t) = -\varepsilon \boldsymbol{u}(t) \tag{6.7}$$

where $\varepsilon u(t)$ is the local error for input at time instant t. It means the corrections ideally cancel the local error in the inputs.After adding a correction δu, it will affect the output $\tilde{y}$ as well, we use the notation δy as the modification for output. This modification to output is of the same order as the input corrections, $\delta y = \mathcal{O}(H^{k+1})$ in *force/displacement* case.

As we did in local error analysis in Chapter 4:

$$\begin{aligned} \varepsilon \boldsymbol{u}(t) &= \Phi(\tilde{\boldsymbol{u}}(t)) - \boldsymbol{u}(t) \\ &= \Phi(\tilde{\boldsymbol{u}}(t)) - L(\tilde{\boldsymbol{y}}(t) - \varepsilon \boldsymbol{y}(t)) \\ \varepsilon \boldsymbol{y}(t) &= \tilde{\boldsymbol{y}}(t) - \boldsymbol{y}(t) \\ &= \boldsymbol{J_g}(\boldsymbol{z})\varepsilon \boldsymbol{z}(t) + \boldsymbol{J_g}(\boldsymbol{u})\varepsilon \boldsymbol{u}(t) + \mathcal{O}(H^{k+2}) \\ \varepsilon \boldsymbol{z}(t) &= \mathcal{O}(H^{k+2}) \end{aligned} \tag{6.8}$$

where k is the degrees of the extrapolation polynomial operator, $J_{\boldsymbol{g}ij}(\boldsymbol{u}) = \partial g_i / \partial u_j$ is the interface Jacobian and $J_{\boldsymbol{g}ij}(\boldsymbol{x}) = \partial g_i / \partial x_j$. Rewriting equation (6.8), we obtain:

$$\varepsilon \boldsymbol{u}(t) = (\boldsymbol{I} - \boldsymbol{LD})^{-1}(\tilde{\boldsymbol{u}}(t) - L\tilde{\boldsymbol{u}}(t)) + \mathcal{O}(H^{k+2}). \tag{6.9}$$

Therefore, NEPCE choose:

$$\delta \boldsymbol{u}(t) = -\varepsilon \boldsymbol{u}(t) \approx (\boldsymbol{I} - \boldsymbol{LD})^{-1}(L\tilde{\boldsymbol{y}}(t) - \tilde{\boldsymbol{u}}(t)) \tag{6.10}$$

in *force/displacement* case. It would make the residual energy vanish.

If the simulation is on time step $(nH, (n+1)H]$, then $\tilde{y}(t)$ is unknown a prior for $t \in (nH, (n+1)H]$ in equation (6.6). Therefore , NEPCE method realize the correction in terms of previous coupling data:

$$\delta \boldsymbol{u}(t) \approx \frac{\alpha}{H}(\boldsymbol{I} - \boldsymbol{LD})^{-1} \int_{(n-1)H}^{nH} (L\tilde{\boldsymbol{y}}(\tau) - \tilde{\boldsymbol{u}}(\tau)) d\tau, \quad \alpha \in [0, 1] \tag{6.11}$$

6.4 NEPCE method Evaluation

To validate NEPCE method, I would like to start my arguments by the following order:

- Is NEPCE method related to power bond and energy conservation law?
- Does residual power indicate the dynamic error in co-simulation system?

Let us start from the first question. It is not precise enough to use power bond and energy conservation in co-simulation system.

First, The idea of energy conservation in NEPCE[21] comes from general equation of energy continuity[20]. In general equation of energy continuity, energy and power has strict physical meaning. Furthermore, the continuity equation is valid only if the energy terms are properly evaluated [20]. The idea of power bond comes from bond graph theory [20][5], where it is designed to represent the behaviour of energy, power, entropy and other physical properties.

Power bond theory and energy conservation are applicable, if we only look into a single subsystem in co-simulation in our research model, that is a mass spring damping system.

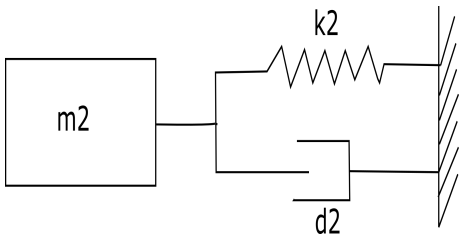

Figure 6.1: A mass spring damping system

where energy is stored in mass and spring, energy is dissipated in the damper. Assuming spring and damper are non-linear, we sum up the physical phenomenon as follow [20]

$$P_{in} - P_{out} = \frac{d}{dt}(E_{k2} + E_{m2}) + \rho_{d2}, \tag{6.12}$$

where E_{k2} and E_{m2} represents energy in spring and mass respectively, ρ_{d2} represents damping dissipation. This means the energy is stored in the mass and spring, while the energy is dissipated by the damper. There exists a power bond between the mechanical system and the outer environment.

However in general weakly coupled system there is no continuous energy transaction between two subsystems, instead it is manually defined macro step size for energy transactions.

Second the usage of power bond concept in co-simulation does not always have strict physical meaning. From Figure 2.1 we can see that, there exists three most common kinds of weakly coupled system in modular level.

- *force/force* coupling, input and output in each subsystem are force.
- *force/displacement* coupling, input or output in each subsystem is either force or velocity and displacement.
- *displacement/displacement* coupling, input and output in each subsystem are displacement.

Therefore only in *force/displacement* coupling, the product of force and velocity has physical meaning. However, as we can see from the explicit expression in *force/force* coupling and *displacement/displacement* coupling , input and output are a vector of displacement or velocity both. Therefore, input or output is not an extensive variable (velocity, flow, etc) strictly. There is no power bonds in the bond graph

and the energy continuity sense [20][5] connecting these subsystems.

For second question we evaluate the functionality of residual power as error indicator. An error indicator should be able to give the information about how good/bad the simulation is, which means difference between the dynamic in the weakly coupled model and strongly coupled system. Here we ignore the physical power and energy meaning, only focus on the residual power numeric value in NEPCE [21] method. A general co-simulation model with two subsystems [18] can be generated as follow:

$$\begin{aligned} \dot{\boldsymbol{z}}_i &= f_i(\boldsymbol{z}_i(t), \boldsymbol{u}_i(t)), \quad i = 1, 2 \\ \boldsymbol{y}_i(t) &= g_i(\boldsymbol{z}_i(t), \boldsymbol{u}_i(t)), \quad i = 1, 2 \end{aligned}$$

By residual power definition:

$$\begin{aligned} -\delta P(t_i) &= \Phi(\tilde{\boldsymbol{u}}_1(t_i))\tilde{\boldsymbol{y}}_1(t_i) + \Phi(\tilde{\boldsymbol{u}}_2(t_i))\tilde{\boldsymbol{y}}_2(t_i) \\ 0 &= \boldsymbol{u}_1(t_i)\boldsymbol{y}_1(t_i) - \boldsymbol{u}_2(t_i)\boldsymbol{y}_2(t_i) \end{aligned} \tag{6.13}$$

We subtract two equations above, see how much residual power speaks as an error indicator:

$$\begin{aligned} -\delta P(t_i) &= [\Phi(\tilde{\boldsymbol{u}}_1(t_i))\tilde{\boldsymbol{y}}_1(t_i) - \boldsymbol{u}_1(t_i)\boldsymbol{y}_1(t_i)] + [\Phi(\tilde{\boldsymbol{u}}_2(t_i))\tilde{\boldsymbol{y}}_2(t_i) - \boldsymbol{u}_2(t_i)\boldsymbol{y}_2(t_i)] \\ &= [\Phi(\tilde{\boldsymbol{u}}_1(t_i))\tilde{\boldsymbol{y}}_1(t_i) - \Phi(\tilde{\boldsymbol{u}}_1(t_i))\boldsymbol{y}_1(t_i) + \Phi(\tilde{\boldsymbol{u}}_1(t_i))\boldsymbol{y}_1(t_i) - \boldsymbol{u}_1(t_i)\boldsymbol{y}_1(t_i)] \\ &\quad + [\Phi(\tilde{\boldsymbol{u}}_2(t_i))\tilde{\boldsymbol{y}}_2(t_i) - \Phi(\tilde{\boldsymbol{u}}_2(t_i))\boldsymbol{y}_2(t_i) + \Phi(\tilde{\boldsymbol{u}}_2(t_i))\boldsymbol{y}_2(t_i) - \boldsymbol{u}_2(t_i)\boldsymbol{y}_2(t_i)] \\ &= [\Phi(\tilde{\boldsymbol{u}}_1(t_i))E(\boldsymbol{y}_1(t_i)) + E(\boldsymbol{u}_1(t_i))\boldsymbol{y}_1(t_i)] \\ &\qquad\qquad + [\Phi(\tilde{\boldsymbol{u}}_2(t_i))E(\boldsymbol{y}_2(t_i)) + E(\boldsymbol{u}_2(t_i))\boldsymbol{y}_2(t_i)] \\ &= [\Phi(\tilde{\boldsymbol{u}}_1(t_i))\frac{\partial g_1}{\partial z_1}(t_i)E(\boldsymbol{z}_1(t_i)) + \Phi(\tilde{\boldsymbol{u}}_1(t_i))\frac{\partial g_1}{\partial \boldsymbol{u}_1}(t_i)E(\boldsymbol{u}_1(t_i)) \\ &\quad + E(\boldsymbol{u}_1(t_i))\boldsymbol{y}_1(t_i)] + [\Phi(\tilde{\boldsymbol{u}}_2(t_i))\frac{\partial g_2}{\partial z_2}(t_i)E(\boldsymbol{z}_2(t_i)) \\ &\quad + \Phi(\tilde{\boldsymbol{u}}_2(t_i))\frac{\partial g_2}{\partial \boldsymbol{u}_2}(t_i)E(\boldsymbol{u}_2(t_i)) + E(\boldsymbol{u}_2(t_i))\boldsymbol{y}_2(t_i)] \end{aligned} \tag{6.14}$$

where $\frac{\partial g_i}{\partial z_i}$, $\frac{\partial g_i}{\partial u_1}$ represent the jacobian matrix.

We can see that not only the global error in dynamics but also the global error in input will contribute to the value of residual power. For residual power being a good error indicator, it should be able to tell the error (dynamic behaviour $E(z)$) of the model. Therefore, one would hope the irrelevant components (not relate to $E(z)$) contribution to residual power as less as possible. Only in *displacement/displacement* coupling, where there is nondirect feed-through, residual power has the least irrelevant components;

$$\begin{aligned} -\delta P(t_i) = &[\Phi(\tilde{\boldsymbol{u}}_1(t_i))\frac{\partial g_1}{\partial \boldsymbol{z}_1}(t_i)E(\boldsymbol{z}_1(t_i)) + E(\boldsymbol{u}_1(t_i))\boldsymbol{y}_1(t_i)] \\ &+ [\Phi(\tilde{\boldsymbol{u}}_2(t_i))\frac{\partial g_2}{\partial \boldsymbol{z}_2}(t_i)E(\boldsymbol{z}_2(t_i)) + E(\boldsymbol{u}_2(t_i))\boldsymbol{y}_2(t_i)] \end{aligned} \tag{6.15}$$

However this does not mean that NEPCE methods are quantitatively better always with *displacement/displacement* coupling than with *force/displacement* coupling.

Generally speaking:

$$\begin{aligned} -\delta P(t_i) \to 0 &\Rightarrow \Phi(\tilde{\boldsymbol{u}}_1(t_i))\tilde{\boldsymbol{y}}_1(t_i) - \Phi(\tilde{\boldsymbol{u}}_2(t_i))\tilde{\boldsymbol{y}}_2(t_i) \to 0 \\ &\not\Rightarrow \Phi(\tilde{\boldsymbol{u}}(t_i)) \to \boldsymbol{u}_(t_i) \quad \text{and} \quad \Phi(\tilde{\boldsymbol{y}}(t_i)) \to \boldsymbol{y}(t_i) \\ &\not\Rightarrow E(\boldsymbol{z}(t_i)) \to 0 \end{aligned}$$

6.5 Simulation

To see the connection between residual power and state error, here we proceed some numerical tests base on our research model *force/displacement* coupling. We get to know the proper design for *force/displacement* coupling in Chapter 5, hence we two different parameter set-ups are tested here. Namely one with proper designed parameters, the other with improper designed parameters. We compare the residual power with dynamic error in subsystem S_1 and S_2 and dynamic error summation respectively. To be clear that, in the numerical tests here only absolute value will be compared, input value will be approximated by constant, linear and quadratic extrapolation polynomial respectively.

6.5.1 Proper Designed Model

Adopting the conclusion from Chapter 5, it is favourable to have a soft interface and place force to the subsystem that weights more, therefore we define the parameters as follow:

k_1	100.0	N/m
d_1	100.0	Ns/m
m_1	100.0	kg
k_2	100.0	N/m
d_2	100.0	Ns/m
m_2	450.0	kg
k_c	60.0	N/m
d_c	60.0	Ns/m
$H = h$	0.01	s

Table 6.1: Parameters set-up 3

First, the input value is approximated by constant extrapolation.

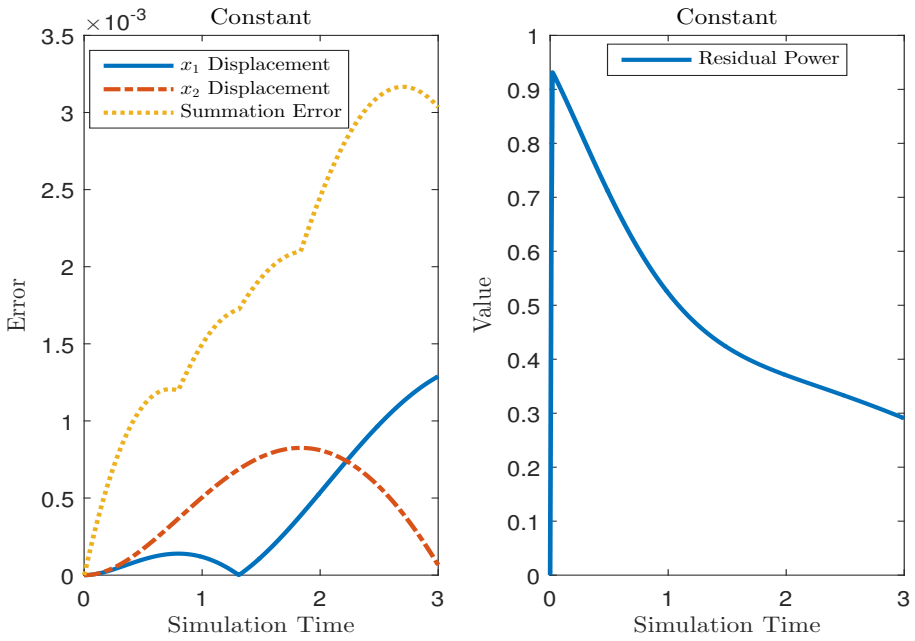

Figure 6.2: State error and residual power comparison 1

Considering there are several order difference in terms of numeric value between residual power and state error, we have separated them in two different graphs. But the trace is very obvious that they are at least not linear related. Residual power can indicates neither the summation state error nor the single subsystem state error.

To see the correlation better, here we introduce a method call Pearson correlation coefficient (PCC), which is a measure of the linear correlation between two variables. It has value from $[-1, 1]$, where 1 represents total positive linear correlation, 0 represents no linear correlation and -1 represents total negative linear correlation. Here we will compare the PCC between state error and residual power from the above numerical results.

Remark. *PCC is calculated through:*

$$r(\delta P, y) = \frac{\sum_{i=1}^{n}(\delta P(t_i) - \bar{\delta P})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(\delta P(t_i) - \bar{\delta P})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where r is PCC coefficient, $\bar{\delta P}$ and $\bar{y}$ represent the residual power expectation and y expectation. Set y can be chosen from subsystem S_1, S_2 and the summation state error set.
In the correlation matrix means:

$$\rho(\delta P, y) = \begin{bmatrix} r(\delta P, \delta P) & r(\delta P, y) \\ r(y, \delta P) & r(y, y) \end{bmatrix}$$

To obtain PCC, we just need to look at the anti-diagonal elements in correlation matrix.

Obtaining correlation matrices between the absolute state error on x_1 displacement, x_2 displacement, the state error summation and the residual power respectively as follow:

$$\begin{bmatrix} 1.0000 & -0.7012 \\ -0.7012 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.6646 \\ -0.6646 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.8968 \\ -0.8968 & 1.0000 \end{bmatrix} \tag{6.16}$$

PCC tells us that absolute state error summation and residual power has quite strong negative correlation. However, it is not enough to use it as an error indicator.

Second, the input value is approximated by linear extrapolation.

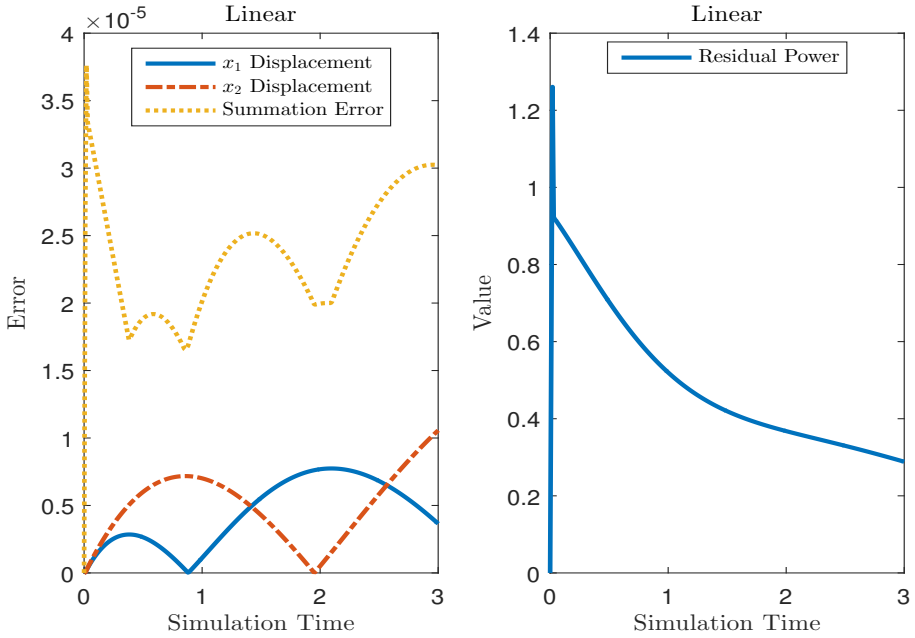

Figure 6.3: State error and residual power comparison 2

In this case, we can see there is no obvious connection between any state error and the residual power. For more information, we can obtain the correlation matrices between the absolute state error on x_1 displacement, x_2 displacement, state error summation and the residual power respectively as follow:

$$\begin{bmatrix} 1.0000 & -0.7035 \\ -0.7035 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.1576 \\ -0.1576 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.1679 \\ -0.1679 & 1.0000 \end{bmatrix} \tag{6.17}$$

Unsurprisingly, the correlation matrices tells us no much in correlation between state error and residual power as it is clear from Figure 6.3. The state error in subsystem S_1 has quite high negative correlation with residual power, but this is not good enough to indicate the state error as well.

Lastly, the input value is approximated by quadratic extrapolation.

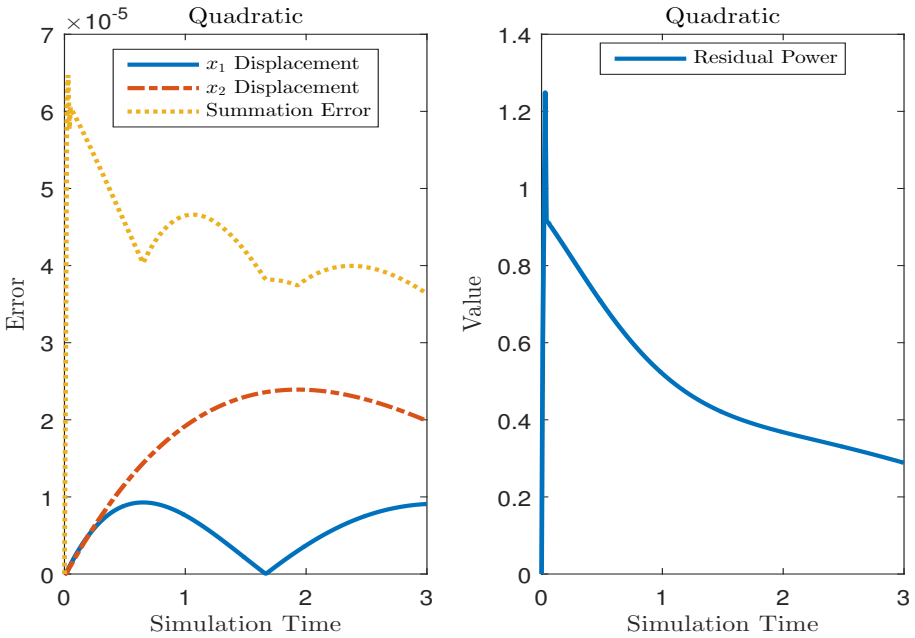

Figure 6.4: State error and residual power comparison 3

From the trace itself, we can see that residual power has a similar trend as the summation of state error. To see more effect on it, one can check the correlation matrices between the absolute state error on x_1 displacement, x_2 displacement, state error summation and the residual power respectively are as follow:

$$\begin{bmatrix} 1.0000 & 0.0320 \\ 0.0320 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.8882 \\ -0.8882 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.8663 \\ 0.8663 & 1.0000 \end{bmatrix} \tag{6.18}$$

Even there is quite negative correlation between residual power and sate error in subsystem S_2, this is not good enough to indicate the state dynamic. Again, it is very difficult to say residual power indicates the state error.

In this proper designed model, there is no sign that residual power has strong linear correlation with dynamic error in subsystems or overall error. Even if we increase extrapolation degree in co-simulation, no obvious signs of linear correlation are shown. Residual power does not have consistent good performance in this proper designed model.

6.5.2 Improper Designed Model

Adopting the conclusion from Chapter 5, it is not so favourable to have high value at the interface and force as input to the subsystem which has lighter mass value. Therefore we design parameters as follow:

k_1	100.0	N/m
d_1	100.0	Ns/m
m_1	100.0	kg
k_2	100.0	N/m
d_2	100.0	Ns/m
m_2	50.0	kg
k_c	130.0	N/m
d_c	130.0	Ns/m
$H = h$	0.01	s

Table 6.2: Parameters set-up 4

First, the input value is approximated by constant extrapolation.

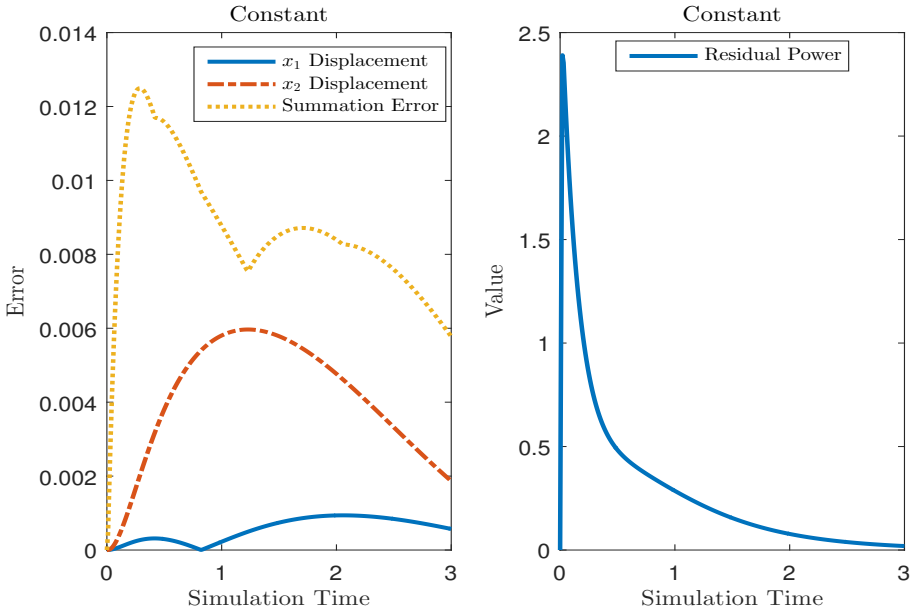

Figure 6.5: State error and residual power comparison 4

From the trace itself, there is not much connection between the dynamic error and the residual power like in proper design model. For more information we can compute the correlation matrices between the absolute state error on x_1 displacement, x_2 displacement, state error summation and the residual power respectively as follow:

$$\begin{bmatrix} 1.0000 & -0.6768 \\ -0.6768 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & -0.5436 \\ -0.5436 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.1602 \\ 0.1602 & 1.0000 \end{bmatrix} \tag{6.19}$$

Second, the input value is approximated by linear extrapolation.

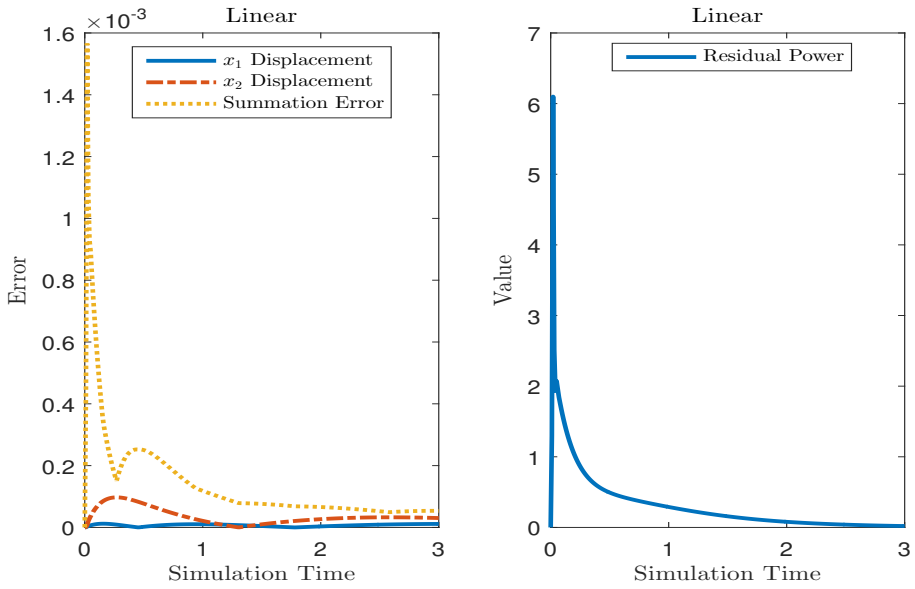

Figure 6.6: State error and residual power comparison 5

From Figure 6.6 we can see that there is very strong linear correlation between the residual power and the summation state error. For more information, PCC matrices between the absolute state error on x_1 displacement, x_2 displacement, the state error summation and the residual power respectively are as follow:

$$\begin{bmatrix} 1.0000 & 0.0408 \\ 0.0408 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.3713 \\ 0.3713 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.9541 \\ 0.9541 & 1.0000 \end{bmatrix} \tag{6.20}$$

There is positive linear correlation between residual power and summation state error as in Figure 6.6, but almost no linear correlation with any subsystem.
Last, the input value is approximated by quadratic extrapolation.

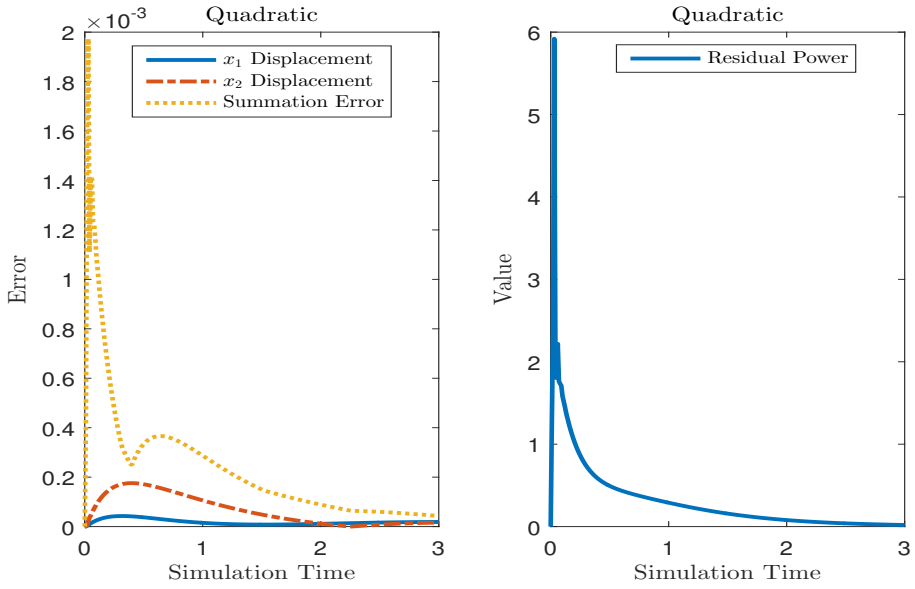

Figure 6.7: State error and residual power comparison 6

From the trace itself, at least one can see that residual power has similar trend like the summation of state error. To see more effect on it, we can obtain correlation matrices between the absolute state error on x_1 displacement, x_2 displacement, the state error summation and the residual power respectively as follow:

$$\begin{bmatrix} 1.0000 & 0.3215 \\ 0.3215 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.3687 \\ 0.3687 & 1.0000 \end{bmatrix}, \quad \begin{bmatrix} 1.0000 & 0.9339 \\ 0.9339 & 1.0000 \end{bmatrix} \tag{6.21}$$

From PCC matrices one can see that the absolute state error summation has very strong linear correlation with the residual power. But this is not the case in any single subsystem. Again, it is very difficult to say residual power as an error indicator for the weakly coupled model.

From the above numerical tests, we can see that as parameters change, residual power functionality changes drastically as well. This does not speak for the generality good performance of residual power as an error indicator.

6.6 Conclusions

In cases above, regardless of the design of the model, sometimes the residual power has positive/ negative correlation with the absolute summation state error, sometimes not. As we derived the residual power formula earlier, there are irrelevant contributions than just state error to the numeric value of residual power. That is the main reason why residual power does not have a consistent good performance to indicate state error. NEPCE method gives us more inside look into input/output connection, but it has no linear connection with co-simulation dynamic. Therefore, it is not appropriate to use residual power as an error indicator.

7

Causality-based Extrapolation

7.1 Introduction

In this chapter a so-called *causality-based* extrapolation method for co-simulation will be introduced. This method is easy to apply, compute and delivers a fairly good result.

As we only consider *force/displacement* model, the numerical error in the simulator will be larger if the simulator outputs force. To see how it arises, we can formulate it as follow:
In *force/displacement* coupling there will always exists a subsystem S_i with force output, i represents the index for the specific subsystem, which can be written as:

$$\begin{aligned} \dot{\boldsymbol{z}}_i &= \boldsymbol{A}_i \boldsymbol{z}_i + \boldsymbol{B}_i \boldsymbol{u}_i \\ \boldsymbol{y}_i &= \boldsymbol{C}_i \boldsymbol{z}_i + \boldsymbol{D}_i \boldsymbol{u}_i \end{aligned} \tag{7.1}$$

where $\boldsymbol{C}_i$, $\boldsymbol{D}_i$ are non-zero matrices.

From local error analysis we know that, output error order has order reduction if the subsystem is direct feed-through, because:

$$\varepsilon(\boldsymbol{y}_{n+1}) = \boldsymbol{C}\varepsilon(\boldsymbol{z}_{n+1}) + \boldsymbol{D}\varepsilon(\boldsymbol{u}_{n+1}) \tag{7.2}$$

Therefore, we can use extrapolation polynomial of higher degree when there is more numerical error (direct feed-through) in the subsystem but lower degree of extrapolation polynomial when there is less numerical error (nondirect feed-through).

The *causality-based* extrapolation has advantages over the mono-extrapolation methods in terms of:

- Enhance co-simulation accuracy compare to mono-low degree extrapolation.
- Decrease computation costs compare to mono-high degree extrapolation.
- Increase stability region compare to mono-high degree extrapolation

which will be demonstrated in the following sections.

7.2 Causality-based Extrapolation

Definition 7.2.1 (Causality Based Extrapolation)**.** Given a coupled system with subsystem $S_1, S_2 ... S_n$, where $n \in \mathbb{N}$, one use higher degree of input extrapolation

polynomial, if $\boldsymbol{D}_i \neq 0$; use lower degree of input extrapolation polynomial, if $\boldsymbol{D}_i = 0$.

It is not difficult to see why *causality-based* extrapolation has larger stability region than mono-high degree extrapolation co-simulation, since higher degree extrapolation is more sensitive to the step size. Regarding to the computation costs, *causality-based* extrapolation use cost less computation time as higher degree extrapolation costs more computation time than lower degree extrapolation. These will be shown when we proceed the numerical test. Therefore, we will show how it can enhance the co-simulation accuracy.

First let us look into the local error, adopting the procedure as we did in equation (4.5):

$$\begin{aligned}
\epsilon(\boldsymbol{z}_{n+1}) &= \begin{bmatrix} \boldsymbol{z}_{1,n+1} \\ \boldsymbol{z}_{2,n+1} \end{bmatrix} - \begin{bmatrix} \tilde{\boldsymbol{z}}_{1,n+1} \\ \tilde{\boldsymbol{z}}_{2,n+1} \end{bmatrix} \\
&\leq^* \begin{bmatrix} \int_{nH}^{(n+1)H} \| e^{\boldsymbol{A_1}(n+1)H} \| d\tau \boldsymbol{B_1} max \| u_1(\tau) - \Phi(u_1(\tau)) \| \\ \int_{nH}^{(n+1)H} \| e^{\boldsymbol{A_2}(n+1)H} \| d\tau \boldsymbol{B_2} max \| u_2(\tau) - \Phi(u_2(\tau)) \| \end{bmatrix} \\
&\leq \begin{bmatrix} \| (\boldsymbol{A_1})^{-1} \| \mathcal{O}(H) \| \boldsymbol{B_1} \| \mathcal{O}(H^{k+1}) \\ \| (\boldsymbol{A_2})^{-1} \| \mathcal{O}(H) \| \boldsymbol{B_2} \| \mathcal{O}(H^{j+1}) \end{bmatrix}
\end{aligned}$$

where $\leq^*$ suffices if $\| \boldsymbol{A_1} \| \leq c1$, and $\| \boldsymbol{A_2} \| \leq c2$, $c1, c2 \in \mathbb{N}$; and k, j is the extrapolation degree in subsystem S_1 and S_2 respectively.

Similar procedure can be done as in equation (4.10) to compute global error, we obtain:

$$E(z_{n+1}) \leq \begin{bmatrix} \mathcal{O}(H^{k+1}) \\ \mathcal{O}(H^{j+1}) \end{bmatrix} \tag{7.3}$$

In our research model, subsystem S_1 is direct feed-through and subsystem S_2 is nondirect feed-through. Applying *causality-based* extrapolation in our research model, we set $k > j$. Then it is straightforward to see that global error in *causality-based* extrapolation is lower than global error in mono-low degree j case. Numerical results will be performed in the next section.

Furthermore *causality-based extrapolation* can have very good performance like mono-high degree extrapolation co-simulation, if the following are fulfilled:

$$\| \boldsymbol{A_2}^{-1} \| \leq \| \boldsymbol{A_1}^{-1} \| \quad \text{and} \quad \| \boldsymbol{B_2} \| \leq \| \boldsymbol{B_1} \| \tag{7.4}$$

In our research model, we use constant input extrapolation polynomial in subsystem S_2, and linear input extrapolation polynomial in subsystem S_1. Denote $\boldsymbol{z}_{i,j}$ as state value in subsystem i, at time j. And the idea can be written as follow:

$$
\begin{aligned}
\tilde{\boldsymbol{z}}_{1,n+1} &= e^{\boldsymbol{A}_1 H}\boldsymbol{z}_{1,n} + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}_1((n+1)H-\tau)}\boldsymbol{B}_1\left(\tilde{\boldsymbol{u}}_{1,n} + H\frac{(\tilde{\boldsymbol{u}}_{1,n}-\tilde{\boldsymbol{u}}_{1,n-1})}{H}\right)d\tau \\
&= e^{\boldsymbol{A}_1 H}\tilde{\boldsymbol{z}}_{1,n} + (\boldsymbol{K}_1+\boldsymbol{J}_1)\boldsymbol{B}_1\tilde{\boldsymbol{y}}_{2,n} - \boldsymbol{j}_1\boldsymbol{B}_1\tilde{\boldsymbol{y}}_{2,n-1}, \\
\tilde{\boldsymbol{y}}_{1,n+1} &= \boldsymbol{C}_1\boldsymbol{z}_{1,n+1} + \boldsymbol{D}_1\left(\tilde{\boldsymbol{u}}_{1,n} + H\frac{(\tilde{\boldsymbol{u}}_{1,n}-\tilde{\boldsymbol{u}}_{1,n-1})}{H}\right) \\
&= (\boldsymbol{C}_1(\boldsymbol{K}_1(H)+\boldsymbol{J}_1(H))\boldsymbol{B}_1 + 2\boldsymbol{D}_1)\tilde{\boldsymbol{u}}_{1,n} - (\boldsymbol{C}_1\boldsymbol{J}_1(H)\boldsymbol{B}_1+\boldsymbol{D}_1)\tilde{\boldsymbol{u}}_{1,n-1} + \boldsymbol{C}_1 e^{\boldsymbol{A}_1 H}\tilde{\boldsymbol{z}}_{1,n} \\
&= (\boldsymbol{C}_1(\boldsymbol{K}_1(H)+\boldsymbol{J}_1(H))\boldsymbol{B}_1 + 2\boldsymbol{D}_1)\tilde{\boldsymbol{y}}_{2,n} - (\boldsymbol{C}_1\tilde{\boldsymbol{J}}_1(H)\boldsymbol{B}_1+\boldsymbol{D}_1)\tilde{\boldsymbol{y}}_{2,n-1} + \boldsymbol{C}_1 e^{\boldsymbol{A}_1 H}\tilde{\boldsymbol{z}}_{1,n} \\
\tilde{\boldsymbol{z}}_{2,n+1} &= e^{\boldsymbol{A}_2 H}\tilde{\boldsymbol{z}}_{2,n} + \int_{nH}^{(n+1)H} e^{\boldsymbol{A}_2((n+1)H-\tau)}d\tau\boldsymbol{B}_2\tilde{\boldsymbol{u}}_{2,n}, \\
&= e^{\boldsymbol{A}_1 H}\tilde{\boldsymbol{z}}_{2,n} + \boldsymbol{K}_2\boldsymbol{B}_2\tilde{\boldsymbol{y}}_{1,n}, \\
\tilde{\boldsymbol{y}}_{2,n+1} &= \boldsymbol{C}_2\tilde{\boldsymbol{z}}_{2,n+1} + \boldsymbol{D}_1\tilde{\boldsymbol{u}}_{2,n} \\
&= (\boldsymbol{C}_2\boldsymbol{K}_2(H)\boldsymbol{B}_2 + \boldsymbol{D}_2)\tilde{\boldsymbol{y}}_{1,n} + \boldsymbol{C}_1 e^{\boldsymbol{A}_2 H}\tilde{\boldsymbol{z}}_{2,n},
\end{aligned}
\tag{7.5}
$$

where

$$
\begin{aligned}
\boldsymbol{K}_1(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}_1((n+1)H-\tau)}d\tau \\
\boldsymbol{J}_1(H) &= \frac{\int_{nH}^{(n+1)H} e^{\boldsymbol{A}_1((n+1)H-\tau)}\tau}{H}d\tau \\
\boldsymbol{K}_2(H) &= \int_{nH}^{(n+1)H} e^{\boldsymbol{A}_2((n+1)H-\tau)}d\tau
\end{aligned}
$$

To combine equations (7.5) together, we obtain:

$$
\begin{bmatrix}
\boldsymbol{z}_{1,n+1} \\ \boldsymbol{z}_{2,n+1} \\ \boldsymbol{y}_{1,n+1} \\ \boldsymbol{y}_{2,n+1} \\ \boldsymbol{y}_{1,n} \\ \boldsymbol{y}_{2,n}
\end{bmatrix}
=
\begin{bmatrix}
\boldsymbol{S}_{1,1} & \boldsymbol{S}_{1,2} \\ \boldsymbol{S}_{2,1} & \boldsymbol{S}_{2,2}
\end{bmatrix}
\begin{bmatrix}
\boldsymbol{z}_{1,n} \\ \boldsymbol{z}_{2,n} \\ \boldsymbol{y}_{1,n} \\ \boldsymbol{y}_{2,n} \\ \boldsymbol{y}_{1,n-1} \\ \boldsymbol{y}_{2,n-1}
\end{bmatrix}
\tag{7.6}
$$

where

$$
\begin{aligned}
\boldsymbol{S}_{1,1} &= \begin{bmatrix} e^{\boldsymbol{A}_1 H} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & e^{\boldsymbol{A}_2 H} & \boldsymbol{K}_2\boldsymbol{B}_2 \\ \boldsymbol{C}_1 e^{\boldsymbol{A}_1 H} & \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \\
\boldsymbol{S}_{1,2} &= \begin{bmatrix} (\boldsymbol{K}_1(H)+\boldsymbol{J}_1(H))\boldsymbol{B}_1 & \boldsymbol{O} & -\boldsymbol{J}_1(H)\boldsymbol{B}_1 \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{C}_1(\boldsymbol{K}_1(H)+\boldsymbol{J}_1(H))\boldsymbol{B}_1 + 2\boldsymbol{D}_1 & \boldsymbol{O} & -\boldsymbol{C}_1\boldsymbol{J}_1(H)\boldsymbol{B}_1 - \boldsymbol{D}_1 \end{bmatrix} \\
\boldsymbol{S}_{2,1} &= \begin{bmatrix} \boldsymbol{O} & \boldsymbol{C}_2 e^{\boldsymbol{A}_2 H} & \boldsymbol{C}_2\boldsymbol{K}_2(H)\boldsymbol{B}_2 + \boldsymbol{D}_2 \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \\
\boldsymbol{S}_{2,2} &= \begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{I} & \boldsymbol{O} & \boldsymbol{O} \end{bmatrix}
\end{aligned}
$$

Based on the assumption above, *causality-based* extrapolation has almost as good as linear extrapolation co-simulation results, if:

$$\begin{aligned} &\|\boldsymbol{A_2}^{-1}\| \leq \|\boldsymbol{A_1}^{-1}\| \\ \text{and} \quad &m_1 \leq m_2 \quad \text{or} \quad k_c, d_c \geq 1 \end{aligned} \tag{7.7}$$

7.3 Simulation

Next we will try to test it on the same model where we used in the previous Chapters, which happens to fulfill our assumption above.

k_1	100.0	N/m
d_1	100.0	Ns/m
m_1	100.0	kg
k_2	100.0	N/m
d_2	100.0	Ns/m
m_2	100.0	kg
k_c	60.0	N/m
d_c	60.0	Ns/m
$H = h$	0.01	s

Table 7.1: Parameters set-up 5

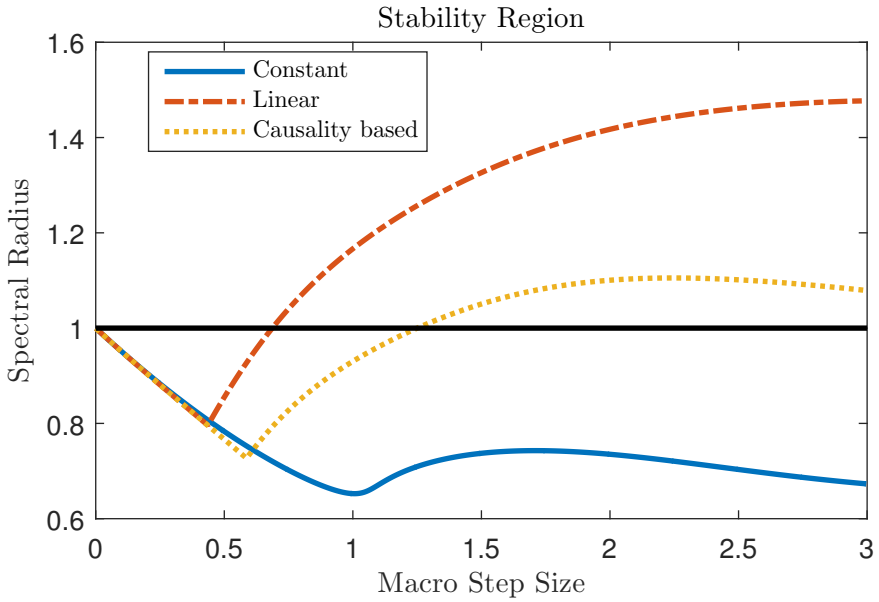

Figure 7.1: Constant, linear and *causality-based* extrapolation spectrum radius

From Figure 7.1, we can see that *causality-based* extrapolation has wider stability region than linear extrapolation but smaller stability region than constant extrapolation.

Apart from stability analysis, we are curious about the *causality-based* extrapolation effect in co-simulation as well. Let us choose step size $H = 0.05$, where it is a stable step size for constant and linear extrapolation.The error performance for three different extrapolation are shown in Figure 7.1.

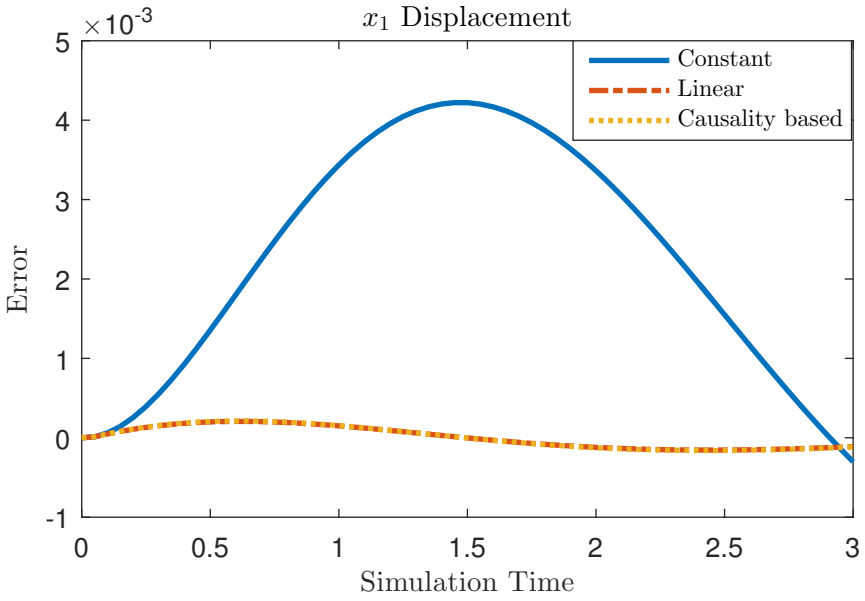

Figure 7.2: Constant, linear and *causality-based* extrapolation error analysis

Unsurprisingly that constant extrapolation has several magnitude large error than the other two, so we try to neglect constant extrapolation and see more details about it.

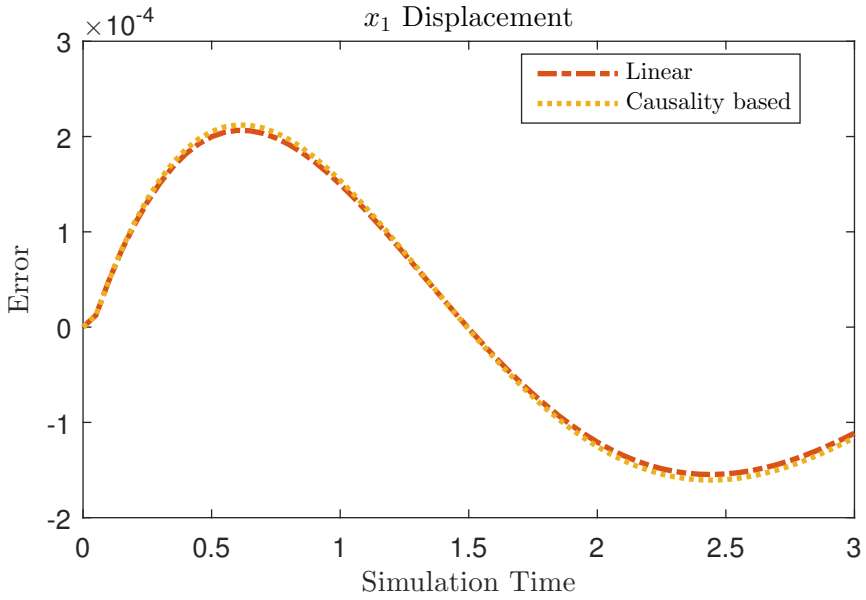

Figure 7.3: Linear and *causality-based* extrapolation error analysis

As expected *causality-based extrapolation* delivers result almost as good as linear extrapolation.

The Matlab computation time for three different extrapolations are listed in the following table:

constant	1.575269	s
causality-based	2.098703	s
linear	2.738973	s

Table 7.2: Constant, linear and *causality-based* extrapolation run time

Therefore,
In terms of error performance: constant < *causality base*d < linear
In terms of computation time: constant < *causality based* < linear
In terms of stability region: linear < *causality based* < constant

7.4 Conclusions

In *causality-based* extrapolation method, higher degree extrapolation polynomial is used only when it is necessary. For instance in output force subsystem, which means it has higher numeric error. We have seen that *causality-based* extrapolation has wider stability region and less computation cost than mono-high degree extrapolation co-simulation. *Causality-based* extrapolation is always more accurate than mono-low degree extrapolation, but less accurate than mono-high degree extrapolation. However, this method can be almost as good as mono-high degree extrapolation in certain models. Hence, *causality-based* extrapolation can be a good alternative in terms of stability and performance.

8

Case Study: EPAS and Chassis System Co-simulation

8.1 Introduction

Steering system is a basic subsystem of the vehicle chassis system. It can transmit the steering input from the human driver to the translation of the steering rack and further into the rotation of the front wheels so that the vehicle motion can be controlled. The steering system is also an important part that the driver has strong interaction with. The driver can perceive how the driving condition is and how the car feels like from the haptic feedback and road disturbance transmitted through the steering mechanisms.

A basic steering system normally involves with several parts, e.g. a steering wheel, a steering column, a steering rack and tie-rods. The upstream steering wheel and column can rotate freely around the rotation axis. The lower end of the steering column is linked to the rack through a rack-pinion which transforms the rotation into translation, as well as the torque into force. The downstream tied-rods are the mechanical linkages between the ends of the rack and the wheel uprights. They work as pull or push rods to make the wheel rotate around the spindle axis.

In modern steering systems some auxiliary parts have been added to the fundamental mechanical parts, for example the hydraulic power assisted steering (HPAS) system or electrically power assisted steering (EPAS) system. Basically an actuator driven by hydraulic or electric systems is added on the rack or column to provide an assisted force. With the help of such system the steering effort require from the driver can be reduced and the steering feeling can be further designed by control algorithms.

8.2 System Modeling

8.2.1 EPAS Modeling

The EPAS mechanical model is constituted by the dynamic equations on the steering column, rack and the electric motor. The equations are briefly given by following equations.

- The dynamic equations on the steering column:

$$J_{column}\ddot{\delta}_s = T_s - T_{pinion} - T_{c_f} \tag{8.1}$$

where J_{column} is the overall inertia of the steering wheel and column, δ_s and T_s are the steering angle and torque, T_{c_f} is the friction torque on the column, T_{pinion} is the torque on the torsion spring of pinion, which is further given by:

$$T_{pinion} = k_t(\delta_s - \delta_p) \tag{8.2}$$

where k_t is the torsion spring stiffness, δ_p is the output shaft angle which is related to the rack displacement x_R by constant ratio η_1.

- The dynamic equations on the steering rack:

$$m_{rack}\ddot{x}_R = F_{pinion} + F_{assist} - F_{rod} - F_{r_f} \tag{8.3}$$

where m_{rack} is the mass of the rack, x_R is the rack displacement, F_{pinion} is the force from pinion to the rack and related to the torsion bar torque T_{pinion} by constant ratio η_1 as well, F_{r_f} is the friction force on the rack.

- The dynamic equations on the electric motor:

$$J_{motor}\ddot{\delta}_m = T_e - T_{belt} - T_{m_f} \tag{8.4}$$

where J_{motor} is the motor inertia and δ_m is the motor rotation angle, T_e is the electric torque by the electric-magnetic field and T_{m_f} is the friction torque of the motor, T_{belt} is the load torque on the belt driven by the motor, T_{belt} can be calculated by:

$$\begin{aligned} T_{belt} &= k_{belt}(\delta_m - \delta_{r1}) + d_{belt}(\dot{\delta}_m - \dot{\delta}_{r1}) \\ \delta_{r1} &= \eta_2\delta_{r2} \end{aligned} \tag{8.5}$$

where k_{belt} and d_{belt} are the equivalent belt stiffness and damping on the input shaft, δ_{r2} is the rotation angle of the output shaft, δ_{r1} is the input shaft rotation angle without compliance and their ratio is η_2. The output shaft rotation δ_{r2} is further transmitted to the rack displacement x_R through a ball-nut gear system by a ratio of η_3. Similar to other mechanical systems the ball-nut gear has masses and friction losses, which the detailed equations are not shown in this thesis.

The friction element T_{c_f}, F_{r_f}, T_{m_f} on each part is modeled by the hyperbolic tangent equations so it facilitates the computation comparing with the discontinuous Coulomb friction model. In addition a dynamic effect has been added so it can capture the hysteresis effect.

8.2.2 Chassis Modeling

The vehicle chassis system consists of the vehicle body, suspensions in the front, rear and four wheels. A detailed chassis model can be a complex assembly of rigid parts which moves according to their constraints. This complex system are usually

simulated by specific multi-body software such as Adams/Car with its own methods. For the simplicity of our research work, a simplified chassis model called bicycle model as shown in Figure 8.1 has been used. The dynamics of the bicycle model is given by dynamic equilibrium in directions of X, Y and Z (equation (8.6)).

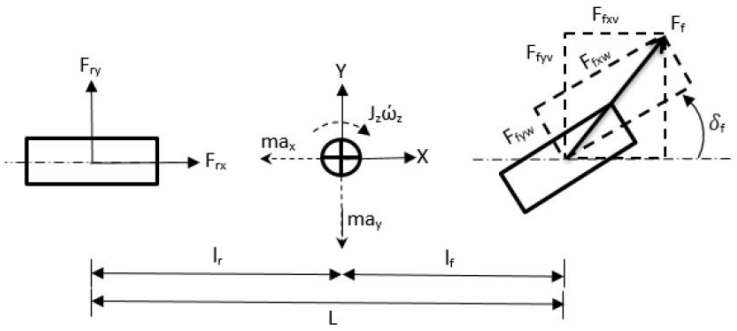

Fig. 8.1: The simplified bicycle model of chassis system

$$\begin{aligned} ma_x &= m(\dot{v}_x - \omega_z v_y) = F_{fxv} + F_{rx} \\ ma_y &= m(\dot{v}_y + \omega_z v_y) = F_{fyv} + F_{ry} \\ J_z\dot{\omega}_z &= F_{fyv} l_f - F_{ry} l_r \end{aligned} \tag{8.6}$$

where m is vehicle mass, J_z is the moment of inertia in Z direction, l_f and l_r are the distance front mass center to the front axle and rear axle.

The tire forces on the right-hand side are calculated by linear tire equations:

$$\begin{aligned} F_{fyv} &= C_f(-\frac{v_y + l_f\omega_z}{v_x} + \delta_f) \\ F_{ry} &= C_r(-\frac{v_y - l_r\omega_z}{v_x}) \end{aligned} \tag{8.7}$$

where C_f and C_r are the cornering stiffness of front and rear axle, δ_f is the steering angle on the front wheel which is proportional to the rack displacement x_R in EPAS model equation (8.3). Similarly, the front tire force F_{fyv} is transmitted to rack as tie-rods force F_{rod} equation (8.3) following the similar kinematic relation.

The dynamics equations can be further formulated in state space form which shows a second-order system behavior. The bicycle model cannot capture the nonlinear and transient behaviors as the complex multibody model. However, it is fairly good to represent the main dynamics of the real chassis at high-speed. Since our purpose is only to verify the co-simulation coupling method, the simplified model has been adopted to mimic the behavior of the multibody one.

8.3 System Co-simulation Results

8.3.1 System Coupling

The layout of system coupling has been shown in Figure 8.2. The chassis model is coupled with steering mechanism by force/displacement or velocity, the steering mechanism is coupled with the electric motor by torque/ angle or rotation velocity. The system can be seen as a complicated version multi mass-spring-damper system. Differently it has nonlinear behavior and gear ratio.

Fig. 8.2: The EPAS and chassis system coupling layout

In practice the complex EPAS model might be provided and validated by the supplier. The vehicle companies have limited access to the detailed models and solvers of the subsystem due to intellectual property. Co-simulation is necessary to integrate subsystems for a holistic system development.

In theory interdisciplinary problem occurs in the detailed EPAS model and chassis model. The electric model model usually has a much faster dynamics: it has components with small masses, friction elements, electric system and the control algorithms which require implicit method or small integration step. The chassis model, which is a multibody system, has much slower dynamics and the Jacobian matrix calculation at every integration step is expensive. Calculating the model in a single simulator at the same step size would be inefficient.

As a result the EPAS model and chassis model are integrated in a multi-rate manner. The chassis model is simulated at fixed time step of 5 ms and the electric motor is simulated at fixed time step of 1 ms. The intermediate steering mechanism can be either modeled together with the chassis or the motor. Thus 4 different possible cases of interface and causality designs are created as shown in Table8.1. Subsystem S_1 and subsystem S_2 are implemented on different simulators. The simulation results of the different cases are given in next section.

Table 8.1: Interface and causality

cases	subsystem 1	subsystem 2	force/torque direction
I	chassis	rack-pinion + EPAS motor	to subsystem 2
II	chassis	rack-pinion + EPAS motor	to subsystem 1
III	chassis + rack-pinion	EPAS motor	to subsystem 2
IV	chassis + rack-pinion	EPAS motor	to subsystem 1

8.3.2 Co-simulation Results

A mono-simulation reference is prepared as well for comparison. A slow sine swept steer with vehicle speed of 50 km/h has been simulated, divergent results occur in case II and case IV. The reason of instability is that a less stable interface has been used according to our previous analysis. Due to the ratio from the belt transmission and the ball-nut gear transmission, the equivalent masses in subsystem 1 from the rack mass m_{rack} and chassis lateral dynamics are hugely scaled down and smaller than the motor inertia J_{motor}. As a result case II and case IV become unstable as expected when the gear ratio values η_2 and η_3 have been reduced.

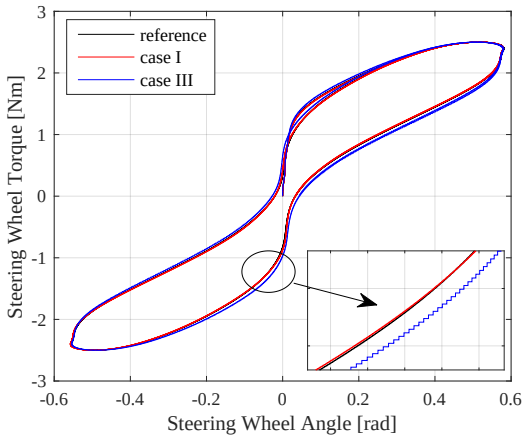

Fig. 8.3: Characteristic curve of steering torque to steering angle

On the contrary the force or torque is applied to the subsystem 2 in case I and case III and the simulations are stable. The resulting steering torque to steering wheel angle characteristic is plotted in Figure 8.3. The normalized root-mean-square error (RMSE) of several state variables are given in Table 8.2. Case I shows slightly less errors than case III. To be clear that, RMSE results indicate the offset errors (due to modular integration in weakly coupled system) and noise results.

Table 8.2: RMSE of co-simulation results in 5ms

variables	case I	case III
steering wheel angle	0.0178	0.1042
assist force	0.0431	0.1313
motor torque	0.2412	0.2963
motor speed	1.4101	3.9603
rack velocity	5.9446	6.1630

Remark. *Root-mean-square error (RMSE) represents the sample standard deviation of the differences date set, generally it can be calculated through:*

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - y_i)^2}{n}}, \quad x, y \text{ represent two different data sets.}$$

To further understand the robustness of case I and case III. Different integration step for subsystem 1 has been tested. The state of rack speed has shown different behaviors as shown in the Figure 8.4, Figure 8.5 and Figure 8.6. At 1 ms both case I and case III have shown a consistent results with the reference because the macro-time step is small. At 5 ms and 10 ms, case I shows more noise results and case III is more robust even with a small offset. Combining RMSE analysis and noise results, case I can have slightly smaller maximum error value than case III but the noisy behavior make the simulation result difficult to use.

From this comparison we can see that case III is more preferred than case I in our research model. The reason might be that the compliance of the transmission belt is a softer interface than the tie-rods. But it should be noticed that the mass difference on both sides also influences the robustness comprehensively. It is intuitive to consider that case I has large step-wise force input to the rack directly which is avoided in case III by different interface selection.

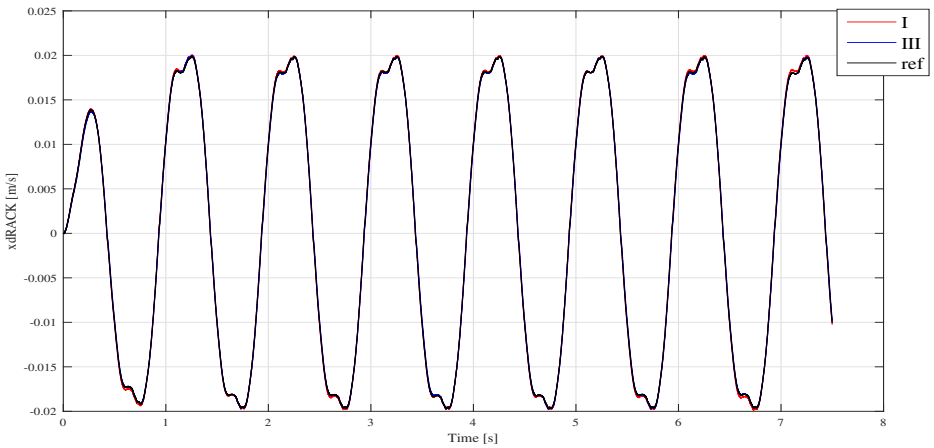

Fig. 8.4: Rack speed in co-simulation with subsystem 1 simulated at 1 ms

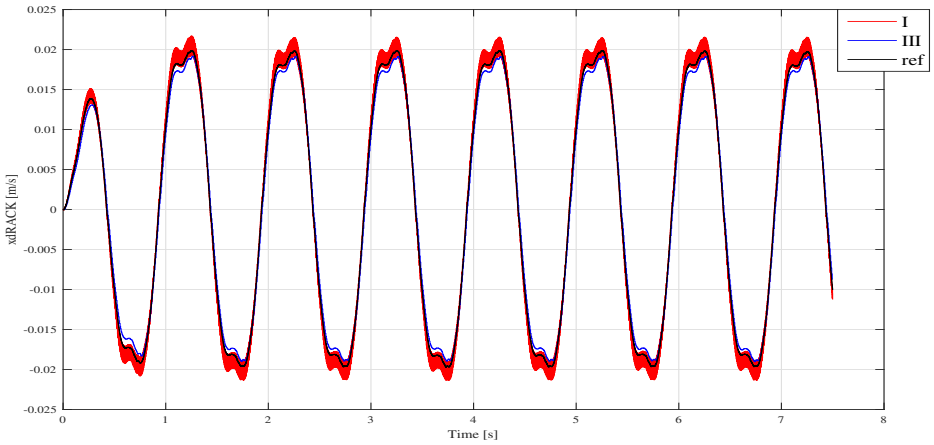

Fig. 8.5: Rack speed in co-simulation with subsystem 1 simulated at 5 ms

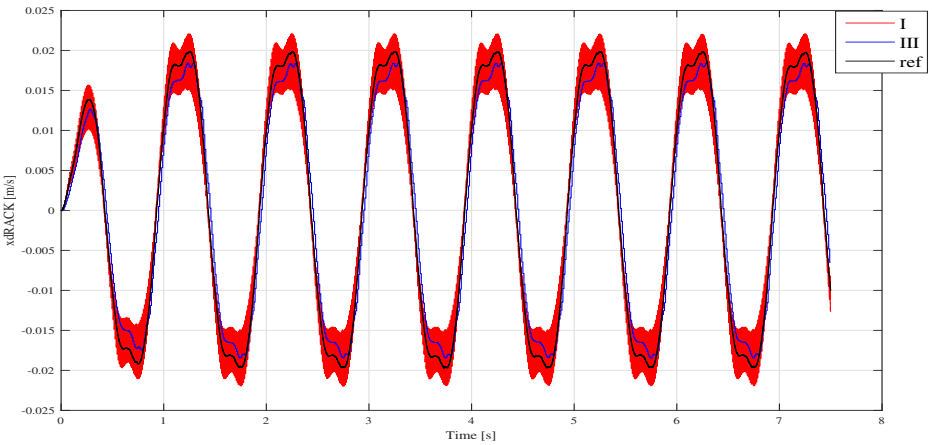

Fig. 8.6: Rack speed in co-simulation with subsystem 1 simulated at 10 ms

8.4 Conclusions

From the case study of chassis and EPAS co-simulation, we can see that the previous guideline from our research model can also apply to complex cases. It provided very useful and intuitive information when simulators are prepared in complex system development.

9
Conclusions

9.1 Summary

Co-simulation is an effective way to solve the multidisciplinary problems. Although the model and calculation algorithm in each simulator are inaccessible due to the intellectual property, we have proven that the coupling methods have more effect in co-simulation accuracy then choices of numerical methods inside the simulators in usual cases. The numerical drawback of modular integration is shown not only in error dynamics but also in the change of stability. The weakly coupled system has smaller stability region than the corresponding strongly coupled system. In addition, stability region in the weakly coupled system decreases when the extrapolation polynomial degree increases. It further decreases, when there is explicit numerical method involved. Based on a clear background of the weakly coupled system, we have given a preference on causality and interface design for our research model with numerical tests validation. Besides, we have presented that NEPCE (energy preserving coupling element) method has a limited usage in our research model by presenting theoretical results and simulation results. Furthermore, without violating the intellectual property of co-simulation we have presented a relative economical, fast and accurate method called *causality-based* extrapolation for *force/displacement* coupling model. In certain cases, it is shown that *causality-based* extrapolation is almost as accurate as mono-high degree extrapolation in co-simulation. Started from theoretical research we came back to a real industrial problem, which is a weakly coupled model of EPAS and chassis systems. It has shown a proper design of causality and interface has strong connection with the stability and robustness of the research model.

9.2 Limitation and Future Work

Due to the time limitation, only the *Lagrange* based extrapolation technique and explicit numerical methods were investigated in this thesis. *Hermite* based extrapolation technique can be discussed and compared with the *Lagrange* based extrapolation technique. We can further investigate the change of stability in weakly coupled system when there is implicit method (or A-stable method) involved. As A-stable methods have no step size restriction in mono-simulation, it is expected that A-stable methods do not effect the stability of weakly coupled system. *Causality-based* extrapolation can be further improved to be an optimal extrapolation method. For example, in continuous time-variant model the choice of extrapolation degree should

be not only based on the causality design but also matrices representation numeric value.

Bibliography

[1] URL: http://fmi-standard.org/.

[2] M. Arnold. "Stability of sequential modular time integration methods for coupled multibody system models". In: *Journal of Computational and Nonlinear Dynamics* 5 (2010).

[3] M. Arnold, C. Clauss, and T. Schierz. "Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation V2.0". In: *The Archive of Mechanical Engineering* 60.1 (2013), pp. 75–94.

[4] J. Bastian et al. "Master for co-Simulation using FMI". In: *8th International Modelica Conference. Dresden*. 2011.

[5] P. C. Breedveld. "Physical systems theory in terms of bond graphs". PhD thesis. Twente University, 1984.

[6] M. Busch. "Zur effizienten Kopplung von Simulationsprogrammen". PhD thesis. Kassel University, 2012.

[7] G. Dahlquist. "A special stability problem for linear multistep methods". In: *BIT* 3 (1963), 27–43.

[8] B.L. Ehle. "High order a-stable methods for the numerical solution of systems of D.E.'s". In: *BIT* 8 (1968), 276–278.

[9] A. C. Faul. *A Concise introduction to numerical analysis*. Chapman and Hall/CRC, 2016, p. 175.

[10] A. Ben Khaled-El Feki. "Distributed real-time simulation of numerical models:application to power-train". PhD thesis. Université de Grenoble, 2014.

[11] A. Ben Khaled-El Feki et al. "CHOPtrey: contextual online polynomial extrapolation for enhanced multi-core co-simulation of complex systems". In: *SIMULATION* 93.3 (2017), pp. 185–200.

[12] J. Fitzgerald, P.G. Larsen, and M. Verhoef(Eds.) *Collaborative design for embedded systems*. 2014.

[13] C. Gomes et al. *Co-simulation: state of the art*. 2017.

[14] B. Goodwine. *Engineering differential equations: theory and applications*. 2008.

[15] T. Groß, S. Trenna, and A. Wirsen. "Solvability and stability of a power system DAE model". In: *Systems Control Letters* 97 (2016), pp. 12–17.

[16] N. H. Ibragimov. *A practical course in differential equations and mathematical modeling*. 2010.

[17] A. Ben Khaled et al. "Context-based polynomial extrapolation and slackened synchronization for fast multi-core simulation using FMI". In: *10th International ModelicaConference, Lund Sweden*. 2014.

[18] R. Kübler and W. Schiehlen. “Two methods of simulator coupling”. In: *Mathematical and Computer Modelling of Dynamical Systems* 6.2 (2000), pp. 93–113.
[19] P. Li. “On the numerical stability of co-simulation methods”. PhD thesis. Technische Universität Darmstadt, 2017.
[20] H. M. Paynter. *Analysis and Design of Engineering Systems: Class Note fir M.I.T. Course 2.751*. Cambridge, Mass.: M.I.T. Press, 1961.
[21] S. Sadjina and E. Pedersen. *Energy conservation and coupling error reduction in non-iterative Co-Simulation*. 2016.
[22] S. Sadjina, S. Skjong, and E. Pedersen. *Energy conservation and power bonds in co-simulations: non-iterative adaptive step size control and error estimation*. 2016.
[23] T. Schierz, M. Arnold, and C. Clauß. “Co-simulation with communication step size control in an FMI compatible master algorithm”. In: *9th International Modelica Conference. Munich*. 2012.
[24] B. Simeon. *Computational Flexible Multibody Dynamics*. 2013.
[25] K. Soetaert and P.M.J. Herman. *A Practical Guide to Ecological Modelling*. 2009.
[26] T. Tomiyama et al. “Complexity of multi-disciplinary design”. In: *CIRP Annals* 56 (2007), pp. 185–188.
[27] L. Turyn. *Advanced engineering mathematics*. 2013, pp. 701–702.
[28] D. G. Zill and M. R. Cullen. *Advanced engineering mathematics*. 2006, p. 277.

A

Appendix 1

The first function is used to obtain the strongly coupled solution of our research model.
The second function is used to obtain the weakly coupled solution (without the usage of numerical method) of our research, where the input value is approximated by constant, linear, quadratic and causality-based extrapolation. In addition, the residual power is computed at every time step.
The third function is used to obtain the spectral radius of our weakly coupled model with and without the usage of numerical method, where the input value is approximated by constant, linear and quadratic extrapolation.

```
%% analytical solution function
function [z] = exactsol(k1,k2,k_c,d1,d2,d_c,m1,m2,t_end,H)
% H is macro step size
% t_end total simulation time

    % define matrix representations for our research model
    %***** define materix A_1 and A_2
    A_1 = [0,1,;-(k1+k_c)/m1, -(d1+d_c)/m1];
    A_2 = [0,1;-(k2/m2), -(d2/m2)];
    %***** define materix B_1 and B_2
    B_1 = [0,0;(k_c/m1), (d_c/m1)];
    B_2 = [0  0;0  (1/m2)];
    % ***** define materix C_1 and C_2
    C_1 = [0  0;k_c, d_c];
    C_2 = [1,0;0, 1];
    % ***** define materix D_1 and D_2
    D_1 = [0  0;-k_c, -d_c];
    D_2 = [0  0;0  0];

    O = zeros(2,2);
    A1 = [A_1 O;O A_2];
    B1 = [B_1 O;O B_2];
    C1 = [C_1 O;O C_2];
    D1 = [D_1 O;O D_2];

    I = eye(4);
    L=[0 0 1 0;0 0 0 1;1 0 0 0;0 1 0 0]; % L is coupling
```

```
        matrix
    time = 0;
    z(:,1) = [1;1;1;1];% initialize state value
        for it = 1:t_end/H
            S = A1+B1*L*inv(I - D1*L)*C1;
            z(:,it+1)= expm(time*S)*z(:,1);
            time=time+H;
        end
end

%% co-simulation performance function
function [z_con,z_lin,z_qua,p_res_con,p_res_lin,p_res_qua] =
    ErrorPerformance(H,t_end,k1,k2,k_c,d1,d2,d_c,m1,m2)
%micro step size=macro step size

%H is macro step size
%t_end total simulation time

    % define matrix representations for our research model
    %***** define materix A_1 and A_2
    A_1 = [0,1,;-(k1+k_c)/m1, -(d1+d_c)/m1];
    A_2 = [0,1;-(k2/m2), -(d2/m2)];
    %***** define materix B_1 and B_2
    B_1 = [0,0;(k_c/m1), (d_c/m1)];
    B_2 = [0  0;0  (1/m2)];
    % ***** define materix C_1 and C_2
    C_1 = [0  0;k_c, d_c];
    C_2 = [1,0;0,  1];
    % ***** define materix D_1 and D_2
    D_1 = [0  0;-k_c, -d_c];
    D_2 = [0  0;0  0];

    z(:,1) = [1;1;1;1]% initialize state value

    %initialize u_con_1 = [x2;x2']
    u_con_1(:,1) = [z(3,1);z(4,1)]; %input value in
        subsystem 1 under constant extrapolation
    u_lin_1 = u_con_1; %input value in subsystem 1 under
        linear extrapolation
    u_qua_1 = u_con_1; %input value in subsystem 1 under
        quadratic extrapolation

    %u_con_2 = [0;f], where f = k1(x1-x2)+d1(x1'-x2')
    u_con_2(:,1) = [0;k_c*(z(1,1)-z(3,1))+d_c*(z(2,1)-z(4,1)
        )];
    u_lin_2 = u_con_2;
    u_qua_2 = u_con_2;
```

```
%initialize z_con_1= [x1;x1']
z_con_1(:,1) = [z(1,1);z(2,1)];
z_lin_1 = z_con_1;
z_qua_1 = z_con_1;

%initialize z_con_2 = [x2;x2']
z_con_2(:,1) = [z(3,1);z(4,1)];
z_lin_2 = z_con_2;
z_qua_2 = z_con_2;

%initialize y_con_1 = [0;f]
y_con_1(:,1) = [0;k_c*(z(1,1)-z(3,1))+d_c*(z(2,1)-z(4,1)
    )];
y_lin_1 = y_con_1;
y_qua_1 = y_con_1;

%initialize y_con_2 = [x2;x2']
y_con_2(:,1) = [z(3,1);z(4,1)];
y_lin_2 = y_con_2;
y_qua_2 = y_con_2;

% initialize residual power
p_res_con = zeros(1,nH_total);
p_res_con(1,1) = u_con_1(2,1)*y_con_1(2,1)-u_con_2(2,1)
    '*y_con_2(2,1);
p_res_lin = p_res_con1;
p_res_qua = p_res_con1;

it = 1;

while it <=t_end/H

        %Exactly solve DE + Constant input approximation
        % ***subsystem1
        k_con_1=@(tau)expm(A_1*((it+1)*H-tau));
        K_con_1=integral(k_con_1,it*H,(it+1)*H,'
            ArrayValued',true);
        z_con_1(:,it+1) = expm(A_1*H)*z_con_1(:,it) +
            K_con_1*B_1*u_con_1(:,it);
        y_con_1(:,it+1) = C_1*z_con_1(:,it+1) + D_1*
            u_con1_1(:,it);
        % ***subsystem2
        k_con_2=@(tau)expm(A_2*((it+1)*H-tau));
        K_con_2=integral(k_con_2,it*H,(it+1)*H,'
            ArrayValued',true);
```

```
z_con_2(:,it+1) = expm(A_2*H)*z_con_2(:,it) +
    K_con_2*B_2*u_con_2(:,it);
y_con_2(:,it+1) = C_2*z_con_2(:,it+1) + D_2*
    u_con_2(:,it);

% residual power
p_res_con(:,it+1) = u_con_1(:,it)'*y_con_1(:,it
    +1)-y_con_2(:,it+1)'*u_con_2(:,it);

% update input value
u_con_1(:,it+1) = y_con_2(:,it+1);
u_con_2(:,it+1) = y_con_1(:,it+1);

if it>2

    %Exactly solve DE + Linear input
        approximation
    % ***subsystem1
    k_lin_1=@(tau)expm(A_1*((it+1)*H-tau))*(tau-
        it*H);
    K_lin_1=integral(k_lin_1,it*H,(it+1)*H,'
        ArrayValued',true);
    z_lin_1(:,it+1) = expm(A_1*H)*z_lin_1(:,it)
        + K_con_1*B_1*u_lin_1(:,it)+K_lin_1*B_1*(
        u_lin_1(:,it)-u_lin_1(:,it-1))/H;
    y_lin_1(:,it+1) = C_1*z_lin_1(:,it+1) + D_1
        *(u_lin_1(:,it)+u_lin_1(:,it)-u_lin_1(:,
        it-1));
    % ***subsystem2
    k_lin_2=@(tau)expm(A_2*((it+1)*H-tau))*(tau-
        it*H);
    K_lin_2=integral(k_lin_2,it*H,(it+1)*H,'
        ArrayValued',true);
    z_lin_2(:,it+1) = expm(A_2*H)*z_lin_2(:,it)
        + K_con_2*B_2*u_lin_2(:,it)+K_lin_2*B_2*(
        u_lin_2(:,it)-u_lin_2(:,it-1))/H;
    y_lin_2(:,it+1) = C_2*z_lin_2(:,it+1) + D_2
        *(u_lin_2(:,it)+u_lin_2(:,it)-u_lin_2(:,
        it-1));

    % residual power
    p_res_lin(:,it+1) = u_lin_1(:,it)'*y_lin_1
        (:,it+1)-y_lin_2(:,it+1)'*u_lin_2(:,it);

    % update input value
    u_lin_1(:,it+1) = y_lin_2(:,it+1);
```

```
        u_lin_2(:,it+1) = y_lin_1(:,it+1);

        %Exactly solve DE + Quadratic input
            approximation
        % ***subsystem1
        k_qua_1=@(tau)expm(A_1*((it+1)*H-tau))*(tau-
            it*H)*(tau-(it-1)*H);
        K_qua_1=integral(k_qua_1,it*H,(it+1)*H,'
            ArrayValued',true);
        z_qua_1(:,it+1) = expm(A_1*H)*z_qua_1(:,it)
            + K_con_1*B_1*u_qua_1(:,it)+K_lin_1*B_1*(
            u_qua_1(:,it)-u_qua_1(:,it-1))/H +
            K_qua_1*B_1*(.5*(u_qua_1(:,it)-2*u_qua_1
            (:,it-1)+u_qua_1(:,it-2)))/(H^2);
        y_qua_1(:,it+1) = C_1*z_qua_1(:,it+1) + D_1
            *(u_qua_1(:,it)+u_qua_1(:,it)-u_qua_1(:,
            it-1)+.5*(u_qua_1(:,it)-2*u_qua_1(:,it-1)
            +u_qua_1(:,it-2)));
        % ***subsystem2
        k_qua_2=@(tau)expm(A_2*((it+1)*H-tau))*(tau-
            it*H)*(tau-(it-1)*H);
        K_qua_2=integral(k_qua_2,it*H,(it+1)*H,'
            ArrayValued',true);
        z_qua_2(:,it+1) = expm(A_2*H)*z_qua_2(:,it)
            + K_con_2*B_2*u_qua_2(:,it)+K_lin_2*B_2*(
            u_qua_2(:,it)-u_qua_2(:,it-1))/H +
            K_qua_2*B_2*(.5*(u_qua_2(:,it)-2*u_qua_2
            (:,it-1)+u_qua_2(:,it-2)))/(H^2);
        y_qua_2(:,it+1) = C_2*z_qua_2(:,it+1) + D_2
            *(u_qua_2(:,it)+u_qua_2(:,it)-u_qua_2(:,
            it-1)+.5*(u_qua_2(:,it)-2*u_qua_2(:,it-1)
            +u_qua_2(:,it-2)));

        % residual power
        p_res_qua(:,it+1) = u_qua_1(:,it)'*y_qua_1
            (:,it+1)-y_qua_2(:,it+1)'*u_qua_2(:,it);

        % update input value
        u_qua_1(:,it+1) = y_qua_2(:,it+1);
        u_qua_2(:,it+1) = y_qua_1(:,it+1);

    elseif it==2

        %Exactly solve DE + Linear input
            approximation
        % ***subsystem1
```

```
k_lin_1=@(tau)expm(A_1*((it+1)*H-tau))*(tau-
    it*H);
K_lin_1=integral(k_lin_1,it*H,(it+1)*H,'
    ArrayValued',true);
z_lin_1(:,it+1) = expm(A_1*H)*z_lin_1(:,it)
    + K_con_1*B_1*u_lin_1(:,it)+K_lin_1*B_1*(
    u_lin_1(:,it)-u_lin_1(:,it-1))/H;
y_lin_1(:,it+1) = C_1*z_lin_1(:,it+1) + D_1
    *(u_lin_1(:,it)+u_lin_1(:,it)-u_lin_1(:,
    it-1));
% ***subsystem2
k_lin_2=@(tau)expm(A_2*((it+1)*H-tau))*(tau-
    it*H);
K_lin_2=integral(k_lin_2,it*H,(it+1)*H,'
    ArrayValued',true);
z_lin_2(:,it+1) = expm(A_2*H)*z_lin_2(:,it)
    + K_con_2*B_2*u_lin_2(:,it)+K_lin_2*B_2*(
    u_lin_2(:,it)-u_lin_2(:,it-1))/H;
y_lin_2(:,it+1) = C_2*z_lin_2(:,it+1) + D_2
    *(u_lin_2(:,it)+u_lin_2(:,it)-u_lin_2(:,
    it-1));

% residual power
p_res_lin(:,it+1) = u_lin_1(:,it)'*y_lin_1
    (:,it+1)-y_lin_2(:,it+1)'*u_lin_2(:,it);

% update input value
u_lin_1(:,it+1) = y_lin_2(:,it+1);
u_lin_2(:,it+1) = y_lin_1(:,it+1);

%Exactly solve DE + Quadratic input
    approximation
% ***subsystem1
z_qua_1(:,it+1) = expm(A_1*H)*z_qua_1(:,it)
    + K_con_1*B_1*u_qua_1(:,it);
y_qua_1(:,it+1) = C_1*z_qua_1(:,it+1) + D_1
    *(u_qua_1(:,it));
% ***subsystem2
z_qua_2(:,it+1) = expm(A_2*H)*z_qua_2(:,it)
    + K_con_2*B_2*u_qua_2(:,it);
y_qua_2(:,it+1) = C_2*z_qua_2(:,it+1) + D_2
    *(u_qua_2(:,it));

% update input value
u_qua_1(:,it+1) = y_qua_2(:,it+1);
u_qua_2(:,it+1) = y_qua_1(:,it+1);
```

```
elseif it==1

    %Exactly solve DE + Linear input
       approximation
    % ***subsystem1
    z_lin_1(:,it+1) = expm(A_1*H)*z_lin_1(:,it)
       + K_con_1*B_1*u_lin_1(:,it);
    y_lin_1(:,it+1) = C_1*z_lin_1(:,it+1) + D_1
       *(u_lin_1(:,it));
    % ***subsystem2
    z_lin_2(:,it+1) = expm(A_2*H)*z_lin_2(:,it)
       + K_con_2*B_2*u_lin_2(:,it);
    y_lin_2(:,it+1) = C_2*z_lin_2(:,it+1) + D_2
       *(u_lin_2(:,it));

    % residual power
    p_res_lin(:,it+1) = u_lin_1(:,it)'*y_lin_1
       (:,it+1)-y_lin_2(:,it+1)'*u_lin_2(:,it);

    % update input value
    u_lin_1(:,it+1) = y_lin_2(:,it+1);
    u_lin_2(:,it+1) = y_lin_1(:,it+1);

    %Exactly solve DE + Quadratic input
       approximation
    % ***subsystem1
    z_qua_1(:,it+1) = expm(A_1*H)*z_qua_1(:,it)
       + K_con_1*B_1*u_qua_1(:,it);
    y_qua_1(:,it+1) = C_1*z_qua_1(:,it+1) + D_1
       *(u_qua_1(:,it));
    % ***subsystem2
    z_qua_2(:,it+1) = expm(A_2*H)*z_qua_2(:,it)
       + K_con_2*B_2*u_qua_2(:,it);
    y_qua_2(:,it+1) = C_2*z_qua_2(:,it+1) + D_2
       *(u_qua_2(:,it));

    % residual power
    p_res_qua(:,it+1) = u_qua_1(:,it)'*y_qua_1
       (:,it+1)-y_qua_2(:,it+1)'*u_qua_2(:,it);

    % update input value
    u_qua_1(:,it+1) = y_qua_2(:,it+1);
    u_qua_2(:,it+1) = y_qua_1(:,it+1);

end
```

```
            it = it + 1;

    end

    z_con(1,:) = z_con_1(1,:);%x1 displacement
    z_con(2,:) = z_con_1(2,:);%x1 speed
    z_con(3,:) = z_con_2(1,:);%x2 displacement
    z_con(4,:) = z_con_2(2,:);%x2 speed

    z_lin(1,:) = z_lin_1(1,:);%x1 displacement
    z_lin(2,:) = z_lin_1(2,:);%x1 speed
    z_lin(3,:) = z_lin_2(1,:);%x2 displacement
    z_lin(4,:) = z_lin_2(2,:);%x2 speed

    z_qua(1,:) = z_qua_1(1,:);%x1 displacement
    z_qua(2,:) = z_qua_1(2,:);%x1 speed
    z_qua(3,:) = z_qua_2(1,:);%x2 displacement
    z_qua(4,:) = z_qua_2(2,:);%x2 speed

end

%% Spectral radius function
function [spectral_con, spectral_lin, spectral_qua,
   spectral_mix] = Spectral(k1,k2,d1,d2,k_c,d_c,m1,m2,H)
    % define matrix representations for our research model
    %***** define materix A_1 and A_2
    A_1 = [0,1,;-(k1+k_c)/m1, -(d1+d_c)/m1];
    A_2 = [0,1;-(k2/m2), -(d2/m2)];
    %***** define materix B_1 and B_2
    B_1 = [0,0;(k_c/m1), (d_c/m1)];
    B_2 = [0 0;0 (1/m2)];
    % ***** define materix C_1 and C_2
    C_1 = [0 0;k_c, d_c];
    C_2 = [1,0;0, 1];
    % ***** define materix D_1 and D_2
    D_1 = [0 0;-k_c, -d_c];
    D_2 = [0 0;0 0];

    O = zeros(2,2);
    A1 = [A_1 O;O A_2];
    B1 = [B_1 O;O B_2];
    C1 = [C_1 O;O C_2];
    D1 = [D_1 O;O D_2];

    % Constant extrapolation + DE solved exactly
    k=@(tau)expm(A1*(H-tau));
    K=integral(k,0,H,'ArrayValued',true);
```

```
A_Con=[expm(A1*H) K*B1*L;
    C1*expm(A1*H) C1*K*B1*L+D1*L];

% Constant extrapolation + DE solved by FE
AA_Con=[I4+H*A1 H*B1*L;C1*(I4+H*A1) C1*H*B1*L+D1*L];

% Linear extrapolation + DE solved exactly
j=@(tau)expm(A1*(H-tau))*tau/H;
J=integral(j,0,H,'ArrayValued',true);
A_Lin=[expm(A1*H) K*B1*L+J*B1*L -J*B1*L;
    C1*expm(A1*H) C1*(K+J)*B1*L+2*D1*L -C1*J*B1*L-D1*L;
    O4 I4 O4];

% Linear extrapolation + DE solved by FE
AA_Lin=[I4+H*A1 2*H*B1*L -H*B1*L;
    C1*(I4+H*A1) 2*C1*H*B1*L+2*D1*L -H*C1*B1*L-D1*L
    O4 I4 O4];

% Quadratic extrapolation + DE solved exactly
g=@(tau)expm(A1*(H-tau))*tau*(tau+H)/(2*H^2);
G=integral(g,0,H,'ArrayValued',true);
A_Qua=[expm(A1*H) (K+J+G)*B1*L -(J+2*G)*B1*L G*B1*L;
    C1*expm(A1*H) C1*(K+J+G)*B1*L + 2.5*D1*L -(J+2*G)*B1
        *L-2*D1*L G*B1*L+0.5*D1*L
    O4 I4 O4 O4
    O4 O4 I4 O4];

% Quadratic extrapolation + DE solved by FE
AA_Qua=[I4+A1*H 2.5*B1*H*L -2*B1*H*L 0.5*B1*H*L;
    C1*(I4+H*A1) 2.5*(B1*H*L+D1*L) -2*(B1*H*L+D1*L)
        0.5*(B1*H*L+D1*L);
    O4 I4 O4 O4
    O4 O4 I4 O4];

% Causality-based extrapolation + DE solved exactly
p_1=@(tau)expm(A_1*(H-tau));
P_1=integral(p_1,0,H,'ArrayValued',true);
pp1_1=@(tau)expm(A_1*(H-tau))*tau/H;
PP1_1=integral(pp1_1,0,H,'ArrayValued',true);
p1_2=@(tau)expm(A_2*(H-tau));
P1_2=integral(p1_2,0,H,'ArrayValued',true);
A1_Mix=[expm(A_1*H) O O (P_1+PP1_1)*B_1 O -PP1_1*B_1;
    O expm(A_2*H) P1_2*B_2 O O O;
    C_1*expm(A_1*H) O O C_1*(P_1+PP1_1)*B_1+2*D_1 O -C_1
        *PP1_1*B_1-D_1;
    O C_2*expm(A_2*H) C_2*P1_2*B_2+D_2 O O O;
```

```
        O O I2 O O O;
        O O O I2 O O];

    spectral_con(1)=max(abs(eig(A_Con)));
    spectral_con(2)=max(abs(eig(AA_Con)));

    spectral_lin(1)=max(abs(eig(A_Lin)));
    spectral_lin(2)=max(abs(eig(AA_Lin)));

    spectral_qua(1)=max(abs(eig(A_Qua)));
    spectral_qua(2)=max(abs(eig(AA_Qua)));

    spectral_mix=max(abs(eig(A1_Mix)));
end
```