The 5th Joint International Conference on Multibody System Dynamics
June 24 – 28, 2018, Lisboa, Portugal

A model-based corrector approach for explicit co-simulation using subspace identification

Timo Haid[1], Georg Stettinger[2], Daniel Watzenig[2,3] and Martin Benedikt[2]

[1] *EVD3 - Multi Domain Simulation, Dr. Ing. h.c. F. Porsche AG, timo.haid@porsche.de*
[2] *Area E - Electronics and Software, VIRTUAL VEHICLE Research Center, {georg.stettinger,martin.benedikt}@v2c2.at*
[3] *IRT - Institute of Automation and Control, TU Graz, daniel.watzenig@tugraz.at*

ABSTRACT — *In this paper a new model-based corrector approach for handling stiff coupling loops in explicit co-simulation problems is presented, where one or more subsystems may have direct feed-throughs, but do not form an algebraic loop. In a first step the algorithm is developed assuming exact model knowledge in form of directional derivatives as provided by the FMI 2.0 standard. In a second step the necessary model information is identified at runtime via a MOESP (Multivariable Output Error State sPace) subspace identification approach directly form the observed input and output data of the subsystems. The quality of the identified model is assessed using a Extended State Space Model (ESSM) approach, eliminating the need to explicitly estimate the state vector in order to predict a time-step. Both algorithm types, the one based on exact model knowledge as well as the identification based, are applied to a dual mass oscillator with linear and non-linear spring-damper characteristic. The gained results are compared to several other coupling algorithms for evaluation purposes. It is shown that the model-based corrector approach produces high quality results even for relatively large macro-steps, while having minimal requirements in terms of model interface and tool capabilities.*

1 Introduction

In recent years, the overall vehicle system has developed into an increasingly complex dynamic system in which a large number of components and functions are connected across several physical domains. This trend is mainly caused by the ongoing electrification of the powertrain, the use of active systems and intelligent control algorithms in almost all areas. On the other hand the development cycles are getting shorter while less real prototypes are available. For these reasons the use of simulation tools in the development of such complex technical systems has evolved from an auxiliary technology to an everyday practice. Although the majority of the development process is still strongly focused on hardware experiments and prototypes, an increasing trend towards pure or at least predominantly virtual development is emerging.

As a result of the trend mentioned above, engineering challenges are increasingly shifting from within the dedicated domains/departments to their interaction and integration. In order to effectively tackle these new cross-domain challenges they must be addressed by a holistic simulation approach, covering all necessary aspects of the vehicle. While some subsystems are modeled in multi-physical simulation tools, most are modeled in tools specifically designed for a particular area of application. As a result of a long-term application specific development process these simulation environments are very matured in terms of usability (through specialized user interfaces and component libraries) and quality of results (through specialized equation formulations and optimized solution methods), resulting in a very heterogeneous model landscape.

Keeping this situation in mind, so-called solver coupling or co-simulation offers a possible solution. Using this approach each involved subsystem is solved independently over a small time interval (the macro-step size ΔT) by

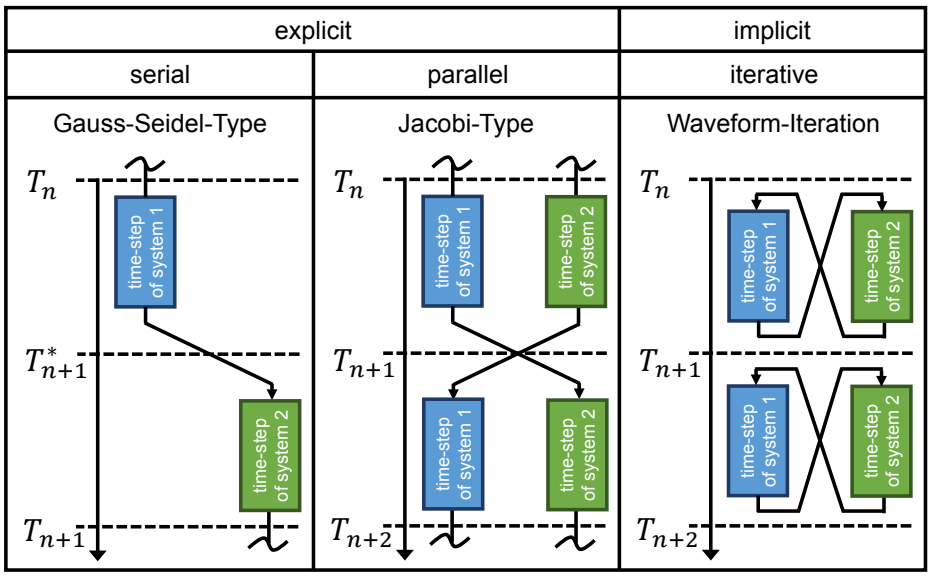


Fig. 1: Categorization of co-simulation methods, c.f. [1]

their respective solver using their respective solver step size (the micro-step size δT), which is why co-simulation is often referred to as a multi-rate, multi-method approach. By exchanging all necessary input and output quantities at the end of each macro-step, the interactions between the individual subsystems are established.The synchronization and the required interfacing of the individual subsystem solvers, as well as the data exchange are usually provided and coordinated by a so-called master algorithm. This master functionality can either be directly integrated in one of the involved simulation tools or provided by a dedicated software, a so called middleware. As systems grow in complexity (number of tools and inputs/outputs) using a dedicated middleware as neutral layer between the individual tools offers advantages in terms of usability, robustness and compatibility.

This paper will start by giving a short introduction to co-simulation in section 2 and address the problems involved with the different coupling schemes. The model-based correction algorithm will be derived from the coupled systems continuous state space form in section 3 . In section 4 a short introduction to subspace identification, especially the MOESP algorithm used in this work will be given. The combination of model-based correction and MOESP identification into an efficient coupling algorithm is descibed in section 5 . In section 6 the performance of the derived coupling algorithm will be assessed via simulation study. A final conclusion and outlook will be given in section 7 .

2 A Short Introduction to Co-Simulation

Because the involved subsystems are interconnected, meaning that inputs of one subsystem are usually outputs of another subsystem, these inputs are unknown at the beginning of the macro-step and are therefore not directly available. Depending on how this problem is solved, co-simulation methods can roughly be categorized by whether macro-steps are repeated and by the scheduling scheme of the subsystem solvers, depicted in figure 1.

2.1 Implicit and Explicit Co-Simulation

If an iteration scheme is used and macro-steps are repeated multiple times this leads to the so called implicit co-simulation [2], Waveform-Iteration [3] [4] or strong or consistent approach [5]. Starting from an estimated solution one macro-step is repeated several times, improving the solution in each step, until a certain termination criterion is reached. As a result, if the implicit co-simulation does converge, it gives very accurate results [6]. Because of the macro-step repetition the involved subsystem solvers have to be reset to a previous state after every iteration. This

steep requirement is not supported by most proprietary simulation tools used in the industry, rendering the iterative approach unfit for most applications [7]. In addition, the repeated calculation of the same macro-step produces an overhead which slows down the overall simulation.

If, on the other hand, a non-iterative scheme is used and every macro-step is only performed once this leads to the so called explicit co-simulation [2] or weak or inconsistent approach [8]. In the non-iterative approach, the necessary inputs are extrapolated from previous and therefore known outputs. Because the extrapolated and the true inputs usually differ, this method induces an error (the coupling error) and therefore gives biased results or may even lead to insability [9]. This basic problem in explicit co-simulation is often referred to as the coupling problem. On the upside explicit co-simulation has minimal requirements regarding the subsystem interfaces, which practically every tool that supports an external interface or the implementation of use-code fulfills, making the explicit approach applicable for most proprietary simulation tools used in the industry.

Depending on whether all subsystem solvers are executed at the same time or one after another, explicit co-simulation can further be divided into the parallel Jacobi type and the sequential Gauss-Seidel type [10] [11]. With regard to the coupling problem, the main difference between the two is that with the Jacobi type all inputs are extrapolated, while with the Gauss-Seidel type some inputs are directly available due to the subsystems already calculated in the sequence. Because the basic coupling problem is the same and to circumvent the scheduling problem, meaning the order of execution involved with sequential co-simulation, this paper refers on parallel co-simulation only.

2.2 Solution Approaches to the Coupling Problem

In order to minimize the coupling error very different approaches can be found in literature, which can basically be divided into three categories. First of all, there are extrapolation methods, which aim to minimize the induced coupling error by predicting the future inputs as accurately as possible before the macro step. Second there are extension methods which aim to adjust the coupling signals during the macro-step calculation for every micro-step by extending the model with an additional dynamic or functionality. Third there are correction methods, which aim to mend the coupling error made by accounting for the induced error in the outputs after the macro step. Because after one macro-step is before the next macro step, each correction method can theoretically be reformulated as an extrapolation method and vice versa. Never the less the presented categorization captures the intention behind the methods.

Typical extrapolation schemes use polynomials of various degrees in order to predict future values. In its simplest form polynomial extrapolation of order zero is used, meaning the previous value is kept constant for the duration of the macro step. This is known as zero order hold (ZOH) approach. For better approximation of the coupling signals first or second order polynomials can be used, resulting in a first order hold (FOH) or second order hold (SOH) approach. If available using micro steps or exact derivatives for extrapolation usually increases the accuracy. While the extrapolation quality might increase with the order of the polynomial, the robustness decreases due to Runge's phenomenon [12]. More sophisticated methods like model-based coupling (MBC) [13] use a local linear representation of the coupled subsystems in order to estimate the coupling signals.

Because in explicit co-simulation coupling data typically can only be exchanged at the end of each macro-step ΔT and not in between, adjusting the input signals for every micro-step δT involves extension of the subsystem by an additional dynamic or functionality. The simplest extension approach is polynomial or spline interpolation of the input signals for each micro-step based on the previous and extrapolated values, which is why interpolation is often inherently included with the respective extrapolation method. Obviously such an interpolation approach only respects the changes in the input signals with respect to time. More advanced approaches also include the change of the inputs with respect to the outputs for every micro-step. Semi Implicit Coupling (SIC) [14] for example uses the corresponding directional derivatives via a Taylor series. Linear Implicit Stabilization (LIS) [15] on the

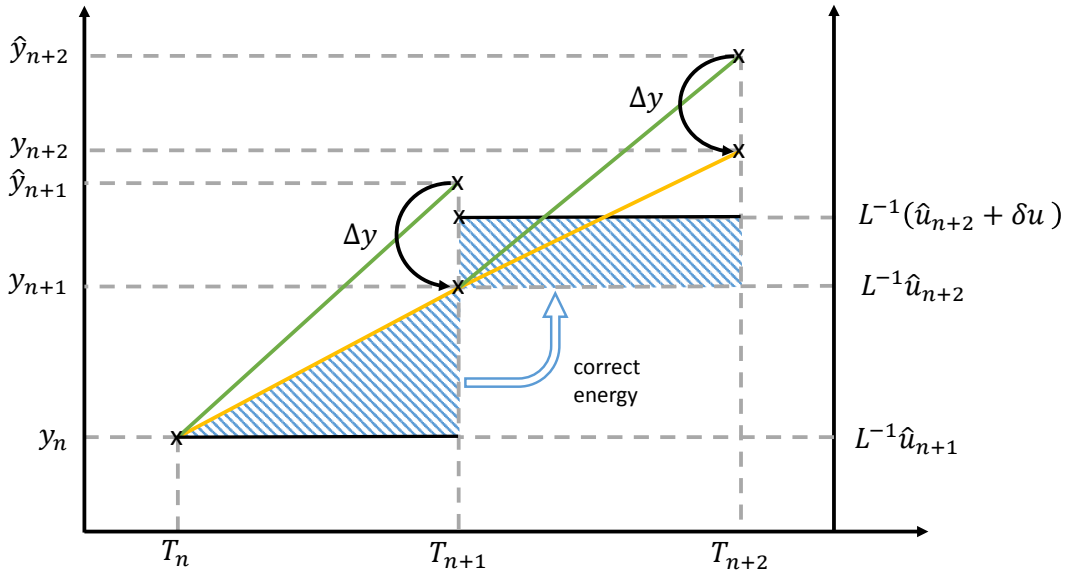


Fig. 2: schematic representation of the model based corrector approach

other hand uses a local linear representation of the residual system and solves it in conjunction with the subsystem itself. Depending on the capabilities of the tool or solver used, extension methods can be either implemented in the master algorithm, the input/output routine or the modeling itself, but apart from a simple interpolation approach this might be difficult to realize in practice.

Since after the macro-step the true values of the inputs are known this additional knowledge can be used in order to correct the error induced by the extrapolation necessary to calculate the macro-step in the first place. A possible approach to improve the overall solution is to correct the input error, which was transferred directly to the outputs due to direct feed-through. This method is known as Parallel Linear Correction (PLC) [16]. Since the approximation error in the input signals also adds a residual energy to the subsystems over the duraton of the macro-step another common approach is based on energy conservation. The Nearly Energy Preserving Coupling Element (NEPCE) [9] aims to correct this residual energy by either adding or removing energy during the next macro-step via an offset to the inputs. To further improve the solution NEPCE may also be combined with feed-though correction [17].

Several of the coupling algorithms presented above allow the solution of strongly coupled systems (and therefore difficult coupling problems) in explicit co-simulation (e.g. NEPCE [9] with feed-through correction [17], SIC [14], LIS [15] and PLC [16]). Almost all of them are based on model knowledge in form of directional derivatives and therefore require a linear representation of the coupled system in the current time step. In principle the FMI for co-simulation 2.0 standard allows the calculation of these directional derivatives [18], but only a few tools commonly used in the automotive industry currently support this feature.

3 A Model Based Corrector Approach

The basic idea of the model based corrector approach is the following two step procedure, which is also depicted in Fig. 2:

1. After every macro-step, estimate the local error $\Delta\mathbf{y}$ in the biased outputs $\hat{\mathbf{y}}$ caused by the error in the estimated inputs $\hat{\mathbf{u}}$ and calculate the corrected outputs $\mathbf{y} = \hat{\mathbf{y}} + \Delta\mathbf{y}$.

2. Account for the error in the subsystem states via an energy correction scheme (cf. NEPCE [9]) during the next macro-step by applying an offset to the estimated inputs $\hat{\mathbf{u}}_{corr} = \hat{\mathbf{u}} + \delta\mathbf{u}$.

3.1 Estimation and Correction of the Local Coupling Error

In order to derive an estimate for the local error we consider $r \geq 2$ coupled non-linear subsystems, where the *ith* subsystem is given in state-space form as

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{f}_i(t, \mathbf{x}_i, \mathbf{u}_i) \\ \mathbf{y}_i &= \mathbf{g}_i(t, \mathbf{x}_i, \mathbf{u}_i) \\ \mathbf{u}_i &= \mathbf{c}_i(\mathbf{y}_1 \dots \mathbf{y}_{i-1}, \mathbf{y}_{i+1}\mathbf{y}_r) \end{aligned} \tag{1}$$

where $i = (1, \dots, r)$ and $\mathbf{x}_i$ denotes the state vector, $\mathbf{u}_i$ and $\mathbf{y}_i$ denote the input and output vectors respectively and $\mathbf{c}(\dots)$ denotes the coupling equations, which are basically simple assignments of the output of other subsystems to the inputs of the considered subsystem. Combining all coupled systems in vector form we get

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(t, \mathbf{x}, \mathbf{u}) \\ \mathbf{y} &= \mathbf{g}(t, \mathbf{x}, \mathbf{u}) \\ \mathbf{u} &= \mathbf{L}\mathbf{y} \end{aligned} \tag{2}$$

where e.g. $\mathbf{x} = \left(\mathbf{x}_1^T, \dots, \mathbf{x}_r^T\right)^T$ denotes the combined state vector [19]. The coupling equation can be written in linear form, where $\mathbf{L}$ denotes the coupling matrix, which is a selection matrix with one non-zero entry per line. Splitting the simulation in time intervals $t \in (T_n, T_{n+1})$ of step size ΔT and replacing the input vector $\mathbf{u}$ for the time interval $T_n \to T_{n+1}$ by an estimated input vector $\hat{\mathbf{u}}$ we get

$$\begin{aligned} \hat{\mathbf{u}} &= e(t, \mathbf{u}_n, \mathbf{u}_{n-1}, \dots) \\ \dot{\hat{\mathbf{x}}} &= \mathbf{f}(t, \hat{\mathbf{x}}, \hat{\mathbf{u}}) \\ \hat{\mathbf{y}} &= \mathbf{g}(t, \hat{\mathbf{x}}, \hat{\mathbf{u}}) \\ \mathbf{u}_{n+1} &= \mathbf{L}\hat{\mathbf{y}}_{n+1} \end{aligned} \tag{3}$$

where the subscript denotes the time instance e.g. $\hat{\mathbf{y}}_{n+1} = \hat{\mathbf{y}}(T_{n+1})$. Because the estimated input vector $\hat{\mathbf{u}}$ is only based on previous values of $\mathbf{u}$, the original coupled system is decoupled into a set of r differential equations which can be solved independently for the duration of the macro-step $T_n \to T_{n+1}$. As stated before, each decoupled subsystem is typically solved with its own integration method and micro-step size (multi-rate, multi-method) with data being exchanged after each of the $n = 1, \dots, N-1$ macro-steps which make up the overall simulation time. Since, in general, the estimated inputs do not exactly correspond to the true inputs we get biased state and output vectors denoted by $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ respectively.

Substraction of eqn. 3 from eqn. 2 and linearization of the resulting error equations on the time interval $T_n \to T_{n+1}$ gives

$$\begin{aligned} \dot{\mathbf{x}} - \dot{\hat{\mathbf{x}}} &= \mathbf{A}_n(\mathbf{x} - \hat{\mathbf{x}}) + \mathbf{B}_n(\mathbf{u} - \hat{\mathbf{u}}) \\ \mathbf{y} - \hat{\mathbf{y}} &= \mathbf{C}_n(\mathbf{x} - \hat{\mathbf{x}}) + \mathbf{D}_n(\mathbf{u} - \hat{\mathbf{u}}) \end{aligned} \tag{4}$$

where $\mathbf{A}_n$, $\mathbf{B}_n$, $\mathbf{C}_n$, $\mathbf{D}_n$ denote the Jacobians $\frac{\partial \mathbf{f}}{\partial x}, \frac{\partial \mathbf{f}}{\partial \mathbf{u}}, \frac{\partial \mathbf{g}}{\partial x}, \frac{\partial \mathbf{g}}{\partial u}$ at time instance T_n respectively. The error in the outputs at the end of the macro-step can therefore be calculated as

$$\Delta \mathbf{y}_{n+1} = \mathbf{y}_{n+1} - \hat{\mathbf{y}}_{n+1} = \mathbf{C}_n(\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1}) + \mathbf{D}_n(\mathbf{u}_{n+1} - \hat{\mathbf{u}}_{n+1}) \tag{5}$$

In order to calculate the necessary state error $(\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1})$ at the end of the macro-step we need to solve the state error differential equation. A solution on the interval $t \in (T_n, T_{n+1})$ can be given as

$$\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1} = e^{\mathbf{A}_n(T_{n+1} - T_n)}(\mathbf{x}_n - \hat{\mathbf{x}}_n) + \int_{T_n}^{T_{n+1}} e^{\mathbf{A}_n(T_{n+1} - \tau)} \mathbf{B}_n(\mathbf{u}(\tau) - \hat{\mathbf{u}}(\tau))\, d\tau \tag{6}$$

Since we are interested in the local error, that is the error induced in a single macro-step, we assume the exact and the biased state vector to be equal at the beginning of the macro step, which eliminates the first term in equation 6

since given this assumption $\mathbf{x}_n - \hat{\mathbf{x}}_n = 0$. In order to obtain a simple solution to the convolution integral in the second term, for now we assume that both the estimated and exact inputs are kept constant for the duration of the macro-step, which results in

$$\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1} = \left(e^{\mathbf{A}_n \Delta T} - \mathbf{I}\right) \mathbf{A}_n^{-1} \mathbf{B}_n \left(\mathbf{u}_{n+1} - \hat{\mathbf{u}}_{n+1}\right) = \mathbf{B}_{d_n} \left(\mathbf{u}_{n+1} - \hat{\mathbf{u}}_{n+1}\right) \tag{7}$$

which is the zero-order hold, discrete time equivalent. Inserting eqn. 7 in eqn. 5 finally yields an estimate for the local error in the outputs due to the coupling error induced in the inputs as

$$\Delta \mathbf{y}_{n+1} \approx \mathbf{y}_{n+1} - \hat{\mathbf{y}}_{n+1} = \left(\mathbf{C}_n \mathbf{B}_{d_n} + \mathbf{D}_n\right) \left(\mathbf{u}_{n+1} - \hat{\mathbf{u}}_{n+1}\right) \tag{8}$$

Together with the coupling equation

$$\mathbf{u}_{n+1} = \mathbf{L} \mathbf{y}_{n+1} \tag{9}$$

eqn. 8 can be solved directly in order to obtain the corrected outputs y_{n+1} from the biased inputs and outputs as

$$\mathbf{y}_{n+1} \approx \left(\mathbf{I} - \left(\mathbf{C}_n \mathbf{B}_{d_n} + \mathbf{D}_n\right) \mathbf{L}\right)^{-1} \left(\hat{\mathbf{y}}_{n+1} - \left(\mathbf{C}_n \mathbf{B}_{d_n} + \mathbf{D}_n\right) \hat{\mathbf{u}}_{n+1}\right) \tag{10}$$

3.2 Correction of the Error in the States

Applying eqn. 10 only corrects the error in the outputs after the macro-step but not the error induced in the states of the subsystems themselves. Since recalculation of the macro-step using the corrected outputs is not possible in explicit co-simulation, the next best thing is to correct the error in the states during the next macro-step by using an energy correction scheme (cf. NEPCE [9] e.g with feed-through correction [17]). The basic idea is to calculate the error integral between the estimated inputs and the exact inputs (based on the corrected outputs). This error integral, which can be interpreted as a sort of generalized energy, is compensated by a corresponding offset δu to the inputs during the next macro-step.

$$\hat{\mathbf{u}}_{c,n+2} = \hat{\mathbf{u}}_{n+2} + \delta \mathbf{u}_{n+2} \tag{11}$$

Assuming constant correction the therefore constant offset for the next macro step can be calculated as

$$\delta \mathbf{u}_{n+2} = \frac{\alpha}{\Delta T} \int_{T_n}^{T_{n+1}} \left(\mathbf{u}(\tau) - \hat{\mathbf{u}}(\tau)\right) d\tau \tag{12}$$

where α denotes a scaling factor. As stated before we assume the estimated inputs $\hat{\mathbf{u}}(\tau)$ to be constant, while in contrast to before, the exact inputs $\mathbf{u}(\tau)$ are estimated using linear interpolation. Using these assumptions together with the coupling equation $\mathbf{u} = L\mathbf{y}$, eqn. 12 can be written as

$$\begin{aligned} \delta \mathbf{u}_{n+2} &= \frac{\alpha}{\Delta T} \left(\Delta T \mathbf{L} \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2} - \Delta T \hat{\mathbf{u}}_{n+1}\right) \\ &= \alpha \left(\mathbf{L} \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2} - \hat{\mathbf{u}}_{n+1}\right) \end{aligned} \tag{13}$$

Applying additional feed-through correction (see [17]), eqn. 13 results in

$$\delta \mathbf{u}_{n+2} = \alpha \left(\mathbf{I} - \mathbf{L}\mathbf{D}_n\right)^{-1} \left(\mathbf{L} \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2} - \hat{\mathbf{u}}_{n+1}\right) \tag{14}$$

Finally the overall algorithm for one macro-step $T_n \rightarrow T_{n+1}$ looks as follows:

- calculate the estimated inputs $\hat{\mathbf{u}}_{n+1}$ using some extrapolation scheme, e.g. ZOH: $\hat{\mathbf{u}}_{n+1} = \mathbf{L}\mathbf{y}_n$
- apply the correction offset from the previous time-step as $\hat{\mathbf{u}}_{c,n+1} = \hat{\mathbf{u}}_{n+1} + \delta \mathbf{u}_{n+1}$
- let the individual subsystems calculate the macro-step using the estimated and corrected inputs $\hat{\mathbf{u}}_{c,n+1}$
- retrive the biased outputs $\hat{\mathbf{y}}_{n+1}$ from the subsystems
- correct the outputs using eqn. 10: $\mathbf{y}_{n+1} \approx \left(\mathbf{I} - \left(\mathbf{C}_n \mathbf{B}_{d_n} + \mathbf{D}_n\right) \mathbf{L}\right)^{-1} \left(\hat{\mathbf{y}}_{n+1} - \left(\mathbf{C}_n \mathbf{B}_{d_n} + \mathbf{D}_n\right) \hat{\mathbf{u}}_{c,n+1}\right)$
- calculate the correction offset for the next macro-step: $\delta \mathbf{u}_{n+2} = \alpha \left(\mathbf{I} - \mathbf{L}\mathbf{D}_n\right)^{-1} \left(\mathbf{L} \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2} - \hat{\mathbf{u}}_{c,n+1}\right)$

4 A Short Introduction to Sub-Space Identification

As stated in the previous section a linear representation of the system in form of the Jacobians $\frac{\partial \mathbf{f}}{\partial x}, \frac{\partial \mathbf{f}}{\partial \mathbf{u}}, \frac{\partial \mathbf{g}}{\partial x}, \frac{\partial \mathbf{g}}{\partial u}$ at time instance T_n or better yet in form of the discrete transition matrices B_{d_n}, C_n, D_n is necessary, in order to estimate and correct the local coupling error. In principle the FMI for co-simulation 2.0 standard allows the calculation of these directional derivatives [18], but only a few tools commonly used in the automotive industry currently support this feature.

Therefore if the Jacobians are not provided they can be estimated using a MOESP subspace identification approach [20] which works on a small batch of previously observed input and output data. In contrast to the more common prediction error methods, e.g. least squares estimation of an ARX model [21], subspace identification methods avoid a priori parameterization like number of poles, zeros and dead time, which gets especially tricky for MIMO systems. In addition subspace techniques are inherently MIMO capable, can handle unstable systems and non-zero initial states. Furthermore the used MOESP method is based on robust numerical operations like QR and singular value decomposition (SVD) [22], which makes them ideal for an automated and user friendly coupling algorithm. Apart from the model-based corrector approach presented in this paper, this technique can also be used to support all other model-based coupling algorithms like PLC or NEPCE with feed-through correction when no model knowledge is available.

4.1 Data and System Matrices

All subspace identification algorithms like N4SID [23], CVA or MOESP [20] are based on a discrete LTI model in state space form, which is repeated for multiple successive time-steps. Since in this case simulation data is used for identification, noise is not an issue and the noise terms can be omitted. The resulting state space equation can be given as

$$\begin{aligned}
\mathbf{y}_n &= \mathbf{C}_d\mathbf{x}_n + \mathbf{D}_d\mathbf{u}_n & \mathbf{x}_{n+1} &= \mathbf{A}_d\mathbf{x}_n + \mathbf{B}_d\mathbf{u}_n \\
\mathbf{y}_{n+1} &= \mathbf{C}_d\mathbf{x}_{n+1} + \mathbf{D}_d\mathbf{u}_{n+1} & \mathbf{x}_{n+2} &= \mathbf{A}_d\mathbf{x}_{n+1} + \mathbf{B}_d\mathbf{u}_{n+1} \\
&= \mathbf{C}_d\mathbf{A}_d\mathbf{x}_n + \mathbf{C}_d\mathbf{B}_d\mathbf{u}_n + \mathbf{D}_d\mathbf{u}_{n+1} & &= \mathbf{A}_d^2\mathbf{x}_n + \mathbf{A}_d\mathbf{B}_d\mathbf{u}_n + \mathbf{B}_d\mathbf{u}_{n+1} \\
\mathbf{y}_{n+2} &= \mathbf{C}_d\mathbf{x}_{n+2} + \mathbf{D}_d\mathbf{u}_{n+2} & \mathbf{x}_{n+2} &= \mathbf{A}_d\mathbf{x}_{n+2} + \mathbf{B}_d\mathbf{u}_{n+2} \\
&= \mathbf{C}_d\mathbf{A}_d^2\mathbf{x}_n + \mathbf{C}_d\mathbf{A}_d\mathbf{B}_d\mathbf{u}_n + \mathbf{C}_d\mathbf{B}_d\mathbf{u}_{n+1} + \mathbf{D}_d\mathbf{u}_{n+2} & &= \mathbf{A}_d^3\mathbf{x}_n + \mathbf{A}_d^2\mathbf{B}_d\mathbf{u}_n + \mathbf{A}_d\mathbf{B}_d\mathbf{u}_{n+1} + \mathbf{B}_d\mathbf{u}_{n+2} \\
\vdots & & \vdots &
\end{aligned} \tag{15}$$

Writing the output equations of eqn. (15) in matrix form for k time-steps yields

$$\begin{bmatrix} \mathbf{y}_n \\ \mathbf{y}_{n+1} \\ \mathbf{y}_{n+2} \\ \vdots \\ \mathbf{y}_{n+k-1} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_d \\ \mathbf{C}_d\mathbf{A}_d \\ \mathbf{C}_d\mathbf{A}_d^2 \\ \vdots \\ \mathbf{C}_d\mathbf{A}_d^{k-1} \end{bmatrix} \mathbf{x}_n + \begin{bmatrix} \mathbf{D}_d & & & \\ \mathbf{C}_d\mathbf{B}_d & \mathbf{D}_d & & \\ \mathbf{C}_d\mathbf{A}_d\mathbf{B}_d & \mathbf{C}_d\mathbf{B}_d & \mathbf{D}_d & \\ \vdots & \ddots & \ddots & \\ \mathbf{C}_d\mathbf{A}_d^{k-2}\mathbf{B}_d & \dots & \mathbf{C}_d\mathbf{B}_d & \mathbf{D}_d \end{bmatrix} \begin{bmatrix} \mathbf{u}_n \\ \mathbf{u}_{n+1} \\ \mathbf{u}_{n+2} \\ \vdots \\ \mathbf{u}_{n+k-1} \end{bmatrix}$$
$$\mathbf{y}_{n|n+k-1} = \quad \boldsymbol{\Gamma}_k \quad \mathbf{x}_n + \quad \boldsymbol{\Psi}_k \quad \mathbf{u}_{n|n+k-1} \tag{16}$$

where $\boldsymbol{\Gamma}_k$ denotes the extended observability matrix, $\boldsymbol{\Psi}_k$ denotes the Toepliz matrix and $\mathbf{y}_{n|n+k-1}$ and $\mathbf{u}_{n|n+k-1}$ hold the k consecutive outputs and inputs respectively. Therefore k also denotes the number of block rows in $\boldsymbol{\Gamma}_k$, $\mathbf{y}_{n|n+k-1}$ and $\mathbf{u}_{n|n+k-1}$ and both the number of block rows and block columns in $\boldsymbol{\Psi}_k$. Note that the index indicates the first and the last time-sample in the data vector, separated by $|$. By repeating eqn. (16) for N consecutive samples, where N is sufficiently large, we get

$$\mathbf{Y}_{n|n+k-1} = \boldsymbol{\Gamma}_k\mathbf{X}_n + \boldsymbol{\Psi}_k\mathbf{U}_{n|n+k-1} \tag{17}$$

where X_n denotes the state sequence given as

$$\mathbf{X}_n = \begin{bmatrix} \mathbf{x}_n & \mathbf{x}_{n+1} & \dots & \mathbf{x}_{n+N-1} \end{bmatrix} \tag{18}$$

and $\mathbf{Y}_{n|n+k-1}$ and $\mathbf{U}_{n|n+k-1}$ denote the Block Hankel matrices of the outputs and inputs, given e.g for the outputs as

$$\begin{aligned} \mathbf{Y}_{n|n+k-1} &= \begin{bmatrix} \mathbf{y}_{n|n+k-1} & \mathbf{y}_{n+1|n+k} & \cdots & \mathbf{y}_{n+N-1|n+k+N-2} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{y}_n & \mathbf{y}_{n+1} & \cdots & \mathbf{y}_{n+N-1} \\ \mathbf{y}_{n+1} & \mathbf{y}_{n+2} & \cdots & \mathbf{y}_{n+N} \\ \mathbf{y}_{n+2} & \mathbf{y}_{n+3} & \cdots & \mathbf{y}_{n+N+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{y}_{n+k-1} & \mathbf{y}_{n+k} & \cdots & \mathbf{y}_{n+k+N-2} \end{bmatrix} \end{aligned} \tag{19}$$

where the index indicates the first and the last time sample in the first block column, since the number of block rows k is relevant, while the number of block columns e.g. samples N just have to be sufficiently large, but has no impact on the algorithm otherwise. The size of the necessary data window therefore results in $K = k + N - 1$.

The basic idea of subspace identification is, that in order to derive the state space model it is sufficient to know either the column space of $\mathbf{\Gamma}_k$ or the row space of $\mathbf{X}_n$. The exact numerical values are not necessary since the state space model is not unique but the subspaces spanned by the rows of $\mathbf{X}_n$ and the columns of $\mathbf{\Gamma}_k$ are invariant [24]. Therefore, only the pair $\mathbf{\Gamma}_k \mathbf{X}_n$ is needed form eqn. (17), which can be derived from observed input and output data by matrix projections.

4.2 The MOESP Algorithm

A quick access to the MOESP algorithm [25] (Multivariable Output Error State sPace) as proposed by Verhagen [20] starts from a matrix formulation of eqn. (17) and a corresponding LQ decomposition of the form

$$\begin{bmatrix} \mathbf{U}_{n|n+k-1} \\ \mathbf{Y}_{n|n+k-1} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{\Psi}_k & \mathbf{\Gamma}_k \end{bmatrix} \begin{bmatrix} \mathbf{U}_{n|n+k-1} \\ \mathbf{X}_n \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{11} & 0 \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_1^T \\ \mathbf{Q}_2^T \end{bmatrix} \tag{20}$$

where $\mathbf{Q}_1^T$ and $\mathbf{Q}_2^T$ are orthogonal and $\mathbf{L}_{11}$,$\mathbf{L}_{22}$ are lower triangular matrices. Inserting the upper block row into the lower block yields

$$\mathbf{Y}_{n|n+k-1} = \mathbf{\Psi}_k \mathbf{L}_{11} \mathbf{Q}_1^T + \mathbf{\Gamma}_k \mathbf{X}_n = \mathbf{L}_{21} \mathbf{Q}_1^T + \mathbf{L}_{22} \mathbf{Q}_2^T \tag{21}$$

Note that the terms of the sum in the right part are orthogonal whereas the terms in the middle part are not necessarily orthogonal, which implies that the summands themselves are not equal to one another. Furthermore it can be seen, that the output data matrix $\mathbf{Y}_{n|n+k-1}$ is split into an orthogonal sum of the orthogonal projection of the row space of $\mathbf{Y}_{n|n+k-1}$ onto the row space of the input data matrix $\mathbf{U}_{n|n+k-1}$ (first summand) and onto its complement $\mathbf{U}_{n|n+k-1}^{\perp}$ (second summand).

Multiplying by $\mathbf{Q}_2$ from the right yields

$$\mathbf{Y}_{n|n+k-1} \mathbf{Q}_2 = \mathbf{\Gamma}_k \mathbf{X}_n \mathbf{Q}_2 = \mathbf{L}_{22} \tag{22}$$

since $\mathbf{Q}_1^T \mathbf{Q}_2 = 0$ and $\mathbf{Q}_2^T \mathbf{Q}_2 = I$. In order to extract the row space of $\mathbf{\Gamma}_k$ a singular value decomposition is applied to eqn. (22), which is partitioned into a significant part and a non-significant or noise part.

$$\mathbf{L}_{22} = \begin{bmatrix} \mathbf{U}_s & \mathbf{U}_n \end{bmatrix} \begin{bmatrix} \mathbf{S}_s & 0 \\ 0 & \mathbf{S}_n \end{bmatrix} \begin{bmatrix} \mathbf{V}_s^T \\ \mathbf{V}_n^T \end{bmatrix} \tag{23}$$

If the input and output data used for identification is derived from a noise-free linear model the number of significant singular values corresponds with the order of the source model and the non-significant singular values are

zero. If the source model is non-linear the non-significant singular values are non-zero since they now contain the non-linearities. The desired order of the estimated state space model can then be chosen either directly by the user or by some heuristic e.g. based on the absolute values or the difference between the singular values. Either way, an estimate for the extended observability matrix of the desired order with the corresponding row space can be given as

$$\mathbf{\Gamma}_k = \mathbf{U}_s \mathbf{S}_s^{1/2} \tag{24}$$

From the definition of extended observability matrix the output matrix $\mathbf{C}_d$ can be directly seen from the first block row. The state transition matrix $\mathbf{A}_d$ can be extracted via least squares from the shift structure of $\mathbf{\Gamma}_k$ as

$$\underline{\mathbf{\Gamma}}_k \mathbf{A}_d = \overline{\mathbf{\Gamma}}_k \tag{25}$$

where $\underline{\mathbf{\Gamma}}_k$ denotes $\mathbf{\Gamma}_k$ without the last block row and $\overline{\mathbf{\Gamma}}_k$ denotes $\mathbf{\Gamma}_k$ without the first block row respectively. In order to derive the feed-through matrix $\mathbf{D}_d$ and the input matrix $\mathbf{B}_d$, eqn.(21) is multiplied from the left by $\mathbf{U}_n^T$ and from the right by $\mathbf{Q}_1$ which yields

$$\mathbf{U}_n^T \mathbf{\Psi}_k \mathbf{L}_{11} = \mathbf{U}_n^T \mathbf{L}_{21} \tag{26}$$

since $\mathbf{U}_n^T \mathbf{L}_{22} = 0$, $\mathbf{U}_n^T \mathbf{\Gamma}_k \mathbf{X}_n = 0$ and $\mathbf{Q}_1^T \mathbf{Q}_1 = I$. The result is an over-determined linear equation with the unknowns $\mathbf{B}_d$ and $\mathbf{D}_d$, which can be solved using least-squares. An efficient algorithm to solve this problem is given in [25].

Finally the overall MOESP algorithm looks as follows

- Compose the Block Hankel data matrices eqn.(19)
- Calculate the LQ decomposition eqn.(20)
- Calculate the SVD of $\mathbf{L}_{22}$ eqn.(23) and decide the order of the model
- Calculate the extended observability matrix $\mathbf{\Gamma}_k$ eqn.(24)
- Derive $\mathbf{C}_d$ and $\mathbf{A}_d$ from $\mathbf{\Gamma}_k$ eqn.(25)
- Derive $\mathbf{B}_d$ and $\mathbf{D}_d$ from eqn.(26)

5 Model Based Correction based on Sub-Space Identification

In order to derive a linear representation of the coupled system the presented MOESP algorithm is executed on a window of adequate size K after every time step. A flow chart of the overall algorithm combining MOESP identification and model based correction is depicted in figure 3. Although the Model Based Correction approach is quite robust against moderate errors in the identified model, it is still susceptible to very poor identification. This may happen due to insufficient excitation, wrong model order or non-linearities in the source model. Therefore the quality of the identified model has to be assessed in order to apply countermeasures, if necessary. Furthermore the identification process can be skipped if the identified model is still valid, reducing the overhead induced by the identification process.

5.1 Quality Assessment

The quality of the identified model is evaluated by checking how good the model predicts the current time step, by calculating the relative prediction error. In order to avoid explicit estimation of the state vector, a so called Extended State Space Model (ESSM) [26] is used for prediction. The state space model eqn. (15) or (16) can generally be written as

$$\hat{\mathbf{y}}_{n+1|n+k-1} = \tilde{\boldsymbol{A}}_k \mathbf{y}_{n|n+k-2} + \tilde{\boldsymbol{B}}_k \mathbf{u}_{n|n+k-1} \tag{27}$$

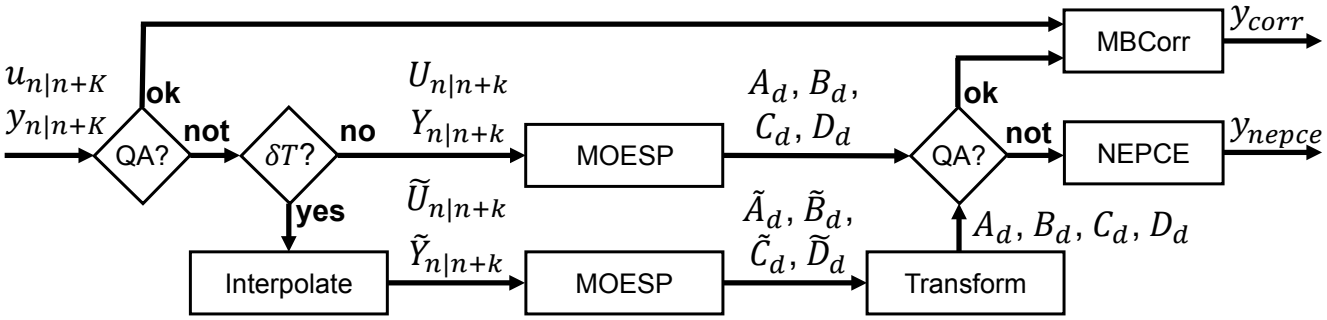


Fig. 3: Flow chart of the overall coupling algorithm, where "QA?" denotes the quality assessment of the previous or updated model and "δT?" denotes whether micro-steps are available or not. Data and system matrices which are not based on the macro-step size ΔT are marked with a tilde.

where $\mathbf{y}_{n|n+k-1}$ and $\mathbf{u}_{n|n+k-1}$ denote the known input and output data vectors respectively. Because the output data vector mimics the function of the state vector, it is sometimes referred to as extended state vector. Note that the equation on the right hand side shifts the output data matrix by one time-step, so that the predicted time step is located in the lowest block row. The transition matrices $\tilde{\boldsymbol{A}}_k$ and $\tilde{\boldsymbol{B}}_k$ can be derived from the underlying state space model eqn. (16) (see [27]):

$$\begin{aligned}\tilde{\boldsymbol{A}}_k &= \underline{\boldsymbol{\Gamma}}_k \mathbf{A}_d \left(\underline{\boldsymbol{\Gamma}}_k^T \underline{\boldsymbol{\Gamma}}_k\right)^{-1} \underline{\boldsymbol{\Gamma}}_k^T = \underline{\boldsymbol{\Gamma}}_k \mathbf{A}_d \underline{\boldsymbol{\Gamma}}_k^\dagger \\ \tilde{\boldsymbol{B}}_k &= \begin{bmatrix}\underline{\boldsymbol{\Gamma}}_k \mathbf{B}_d & \underline{\boldsymbol{\Psi}_k}\end{bmatrix} - \tilde{\boldsymbol{A}}_k \begin{bmatrix}\underline{\boldsymbol{\Psi}_k} & \mathbf{0}\end{bmatrix}\end{aligned} \tag{28}$$

The fact that the ESSM is invariant against transformation of the underlying SSM (and therefore unique for a given system which the SSM is not) and the reuse of data and system matrices used for identification make the ESSM formulation uniquely suited in combination with subspace identification methods. Note that again $\underline{\boldsymbol{\Gamma}}_k$ is reduced by the last block row and $\underline{\boldsymbol{\Psi}_k}$ is reduced by both the last block row and block column compared to $\boldsymbol{\Gamma}_k$ and $\boldsymbol{\Psi}_k$ respectively, in order to reuse the data and system matrices used for identification with block row count k.

In order to predict the current time step, which is located in the lower right corner of the data matrix $\mathbf{Y}_{n|n+k-1}$ (see eqn. (19)), eqn. (27) is applied to the last block column of the input and output data matrices, which yields

$$\hat{\mathbf{y}}_{n+N|n+k+N-2} = \tilde{\boldsymbol{A}}_k \mathbf{y}_{n+N-1|n+k+N-1} + \tilde{\boldsymbol{B}}_k \mathbf{u}_{n+N|n+k+N-2} \tag{29}$$

Extracting the current true and estimated outputs from the last block row of $\mathbf{y}_{n+N|n+k+N-2}$ and $\hat{\mathbf{y}}_{n+N|n+k+N-2}$ respectively, a relative error indicator can then be calculated according to

$$\begin{aligned}\boldsymbol{\varepsilon}_i &= \frac{1}{\alpha \left| \frac{\mathbf{y}_i}{\mathbf{y}_i - \hat{\mathbf{y}}_i} \right| + \frac{1}{\beta}} \\ e_i &= \frac{1}{\sqrt{n_y}} \left|\left| \boldsymbol{\varepsilon}_i \right|\right|_2\end{aligned} \tag{30}$$

where $\mathbf{y}_i$, $\hat{\mathbf{y}}_i$ denote the true and estimated output vector at sample $i = n+k+N-2$ which represents the current macro-step. The parameter α denotes a scaling factor which controls the point where modified and classical relative error are equal and β denotes an upper bound for the relative error. This slightly modified version of the relative error converges towards the classical relative error if the error is small (for $\alpha = 1$), but is bound by β if the relative error gets ill defined. This relative prediction error is then compared to both the prediction error of the previous model and the zero-order-hold prediction error. Both factors α,β are tuned to fit the classical relative error around the chosen threshold.

As can be seen in figure 3 the relative prediction error is calculated once using the previous model at the beginning of the algorithm and then compared to a constant threshold defined by the user in order to decide whether the model can be kept or has to be updated. After the identification step the relative prediction error is calculated

again using the newly identified model. If the newly identified model performs worse than the previous model, the previous model is kept for the correction step, although it violates the given error threshold. If both the previous and new model perform worse than ZOH, the correction step is skipped and the algorithm therefore defaults to energy correction alone, based on the biased outputs, which resembles the NEPCE algorithm in its simplest form.

5.2 Usage of Micro-Steps or Interpolated Data

As stated before the necessary minimal sample size for a given number of block rows k results in $K = k + N - 1$. In order to improve the identification result an being able to work with a smaller amount of makro-steps as usually needed for the identification process, micro-steps can be included, if provided by the subsystem. Since micro-steps are not necessarily equidistant across all subsystems or several macro-steps, e.g. if a variable step solver is used, all available samples (macro- and micro-steps) must be interpolated to a common step-size. Since choosing this interpolation step-size too small is equally bad as choosing the step-size to large, finding a good compromise can be tricky. A simple practical approach is to take the interpolation step-size δT_{int} as a fraction $\delta T_{int} = \frac{1}{r}\Delta T$ of the macro-step size ΔT somewhere in the range of $r = 2..10$ [28].

Interpolation can also be used to create artificial data-points in order to fulfill the formal requirement towards the minimal sample size, which can be helpful at the beginning of the simulation or with non-linear subsystems. Of course neither micro-steps nor interpolation help with insufficient excitation in the input and output signals.

Constructing the input and output data matrices $\tilde{\mathbf{U}}_{n|n+k-1}$, $\tilde{\mathbf{Y}}_{n|n+k-1}$ based on the interpolation step-size δT_{int} and running the MOESP algorithm as usual consequently gives the discrete system matrices $\mathbf{A}_{d_{\delta T}}$,$\mathbf{B}_{d_{\delta T}}$ in the interpolation step-size δT_{int}. Since both the quality assessment and correction algorithms need the discrete system matrices in the macro-step size ΔT, the result has to be transformed [28] using the relation

$$\begin{bmatrix} \mathbf{A}_{d_{\Delta T}} & \mathbf{B}_{d_{\Delta T}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{d_{\delta T}} & \mathbf{B}_{d_{\delta T}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^{\frac{\Delta T}{\delta T}} = \begin{bmatrix} \mathbf{A}_{d_{\delta T}} & \mathbf{B}_{d_{\delta T}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}^{r} \tag{31}$$

Because raising the power of the matrix is necessary it is good practice to choose the factor r as an integer value in order to make computations more efficient.

6 Simulation Study

In a first step the algorithm developed in the previous chapters is now applied to a linear dual mass oscillator, which is probably the most common test-model in literature. In a second step the dual mass oscillator will be modified by adding a non-linear damper characteristic to the coupling spring-damper element.

In order to compare the performance of the different coupling schemes the following error norm is applied:

$$\begin{aligned} \overline{\mathbf{y}}^* &= \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i^* \\ \boldsymbol{\varepsilon}_n &= \frac{(\mathbf{y}_n - \mathbf{y}_n^*)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{y}_i^* - \overline{\mathbf{y}}^*\right)^2}} \\ e_n &= \sqrt{\frac{1}{n_y}}\,||\boldsymbol{\varepsilon}_n||_2 \\ e &= \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_n^2} \end{aligned} \tag{32}$$

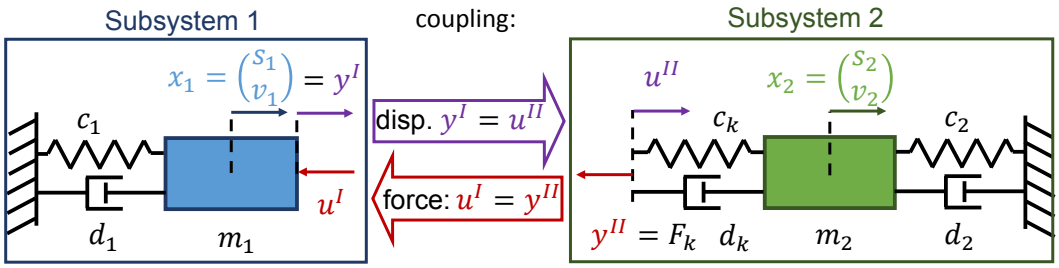


Fig. 4: Dual mass oscillator with force-displacement coupling

Here N denotes the number of macro-steps (number of samples) the simulation has run, n_y denotes the number of outputs and $\mathbf{y}_n$ denotes the outputs calculated by the co-simulation method at macro-step n whereas $\bar{\mathbf{y}}_n^*$ denotes the true outputs calculated by the closed form solution. The error norm therefore gives the global error. As can be seen in eqn. 32 the error of all outputs is normalized by their respective empirical standard deviation. Taking the root mean square of the normed error of the individual outputs yields a scalar error indicator for every macro-step n. Taking the root mean square of all macro-steps finally yields the normalized root mean square error (NRMSE) of the co-simulation method.

6.1 Linear Dual Mass Oscillator

For the linear test-model the two masses are separated into different subsystems using a force-displacement coupling approach, as depicted in figure 4. In this example the coupling spring damper element is located in subsystem II. Therefore subsystem I outputs its displacement and velocity whereas subsystem II returns the reaction force of the spring damper element.

For subsystem I we get:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1^I \\ \dot{x}_2^I \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\frac{c_1}{m_1} & -\frac{d_1}{m_1} \end{bmatrix} \begin{bmatrix} x_1^I \\ x_2^I \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \end{bmatrix} \begin{bmatrix} u_1^I \end{bmatrix} \\
\dot{\mathbf{x}}^I &= \mathbf{A}^I \mathbf{x}^I + \mathbf{B}^I \mathbf{u}^I \\
\begin{bmatrix} y_1^I \\ y_2^I \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1^I \\ x_2^I \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} u_1^I \end{bmatrix} \\
\mathbf{y}^I &= \mathbf{C}^I \mathbf{x}^I + \mathbf{D}^I \mathbf{u}^I
\end{aligned} \tag{33}$$

For subsystem II we get:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1^{II} \\ \dot{x}_2^{II} \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\frac{c_k+c_2}{m_2} & -\frac{d_k+d_2}{m_2} \end{bmatrix} \begin{bmatrix} x_1^{II} \\ x_2^{II} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{c_k}{m_2} & \frac{d_k}{m_2} \end{bmatrix} \begin{bmatrix} u_1^{II} \\ u_2^{II} \end{bmatrix} \\
\dot{\mathbf{x}}^{II} &= \mathbf{A}^{II} \mathbf{x}^{II} + \mathbf{B}^{II} \mathbf{u}^{II} \\
\begin{bmatrix} y_1^{II} \end{bmatrix} &= \begin{bmatrix} c_k & d_k \end{bmatrix} \begin{bmatrix} x_1^{II} \\ x_2^{II} \end{bmatrix} + \begin{bmatrix} -c_k & -d_k \end{bmatrix} \begin{bmatrix} u_1^{II} \\ u_2^{II} \end{bmatrix} \\
\mathbf{y}^{II} &= \mathbf{C}^{II} \mathbf{x}^{II} + \mathbf{D}^{II} \mathbf{u}^{II}
\end{aligned} \tag{34}$$

For the coupling equation we get:

$$\begin{aligned}
\begin{bmatrix} u_1^I \\ u_1^{II} \\ u_2^{II} \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1^I \\ y_2^I \\ y_1^{II} \end{bmatrix} \\
\mathbf{u} &= \mathbf{L}\mathbf{y}
\end{aligned} \tag{35}$$

Parameter	mass m_i	stiffness c_i	damping d_i	init disp. $s_{0,i}$	init vel. $v_{0,i}$
mass 1	$10kg$	$1e6\frac{N}{m}$	$1\frac{Ns}{m}$	$0.1m$	$0\frac{m}{s}$
mass 2	$10kg$	$1e7\frac{N}{m}$	$2\frac{Ns}{m}$	$0.1m$	$0\frac{m}{s}$
coupling	-	$2e5\frac{N}{m}$	$2e2\frac{Ns}{m}$	-	-

Tab. 1: Parameters for the dual-mass oscillator test model

Obviously the force-response of subsystem II has direct feed-through, but because subsystem I has no direct-feed-through in the output equation no algebraic loop exists. Both subsystems can be summarized together with the coupling equation which yields the coupled systems in state space form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \\ \mathbf{u} &= \mathbf{L}\mathbf{y} \end{aligned} \tag{36}$$

The input, output and state vectors are summarized e.g. as $\mathbf{x} = \begin{bmatrix} x_1^I & x_2^I & x_1^{II} & x_2^{II} \end{bmatrix}^T$. The system matrices **A**,**B**,**C** and **D** of the coupled system can then be written in block diagonal structure as

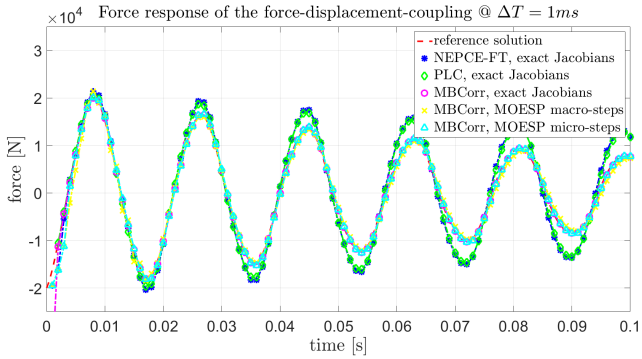
$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} \mathbf{A}^I & 0 \\ 0 & \mathbf{A}^{II} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \mathbf{B}^I & 0 \\ 0 & \mathbf{B}^{II} \end{bmatrix} \\ \mathbf{C} &= \begin{bmatrix} \mathbf{C}^I & 0 \\ 0 & \mathbf{C}^{II} \end{bmatrix}, \mathbf{D} = \begin{bmatrix} \mathbf{D}^I & 0 \\ 0 & \mathbf{D}^{II} \end{bmatrix} \end{aligned} \tag{37}$$

For the simulation study the parameters given in table 1 are used for the dual-mass-oscillator, which is co-simulated using macro-step sizes of $\Delta T = 1ms$, $1.5ms$, $2ms$ and $3ms$. The model-based correction approach presented in this paper is used with micro-steps included, macro-steps only and exact knowledge of the system jacobians. In addition NEPCE with feed-though correction [17] (and constant offset) and Parallel Linear Correction [16] are applied for reference.

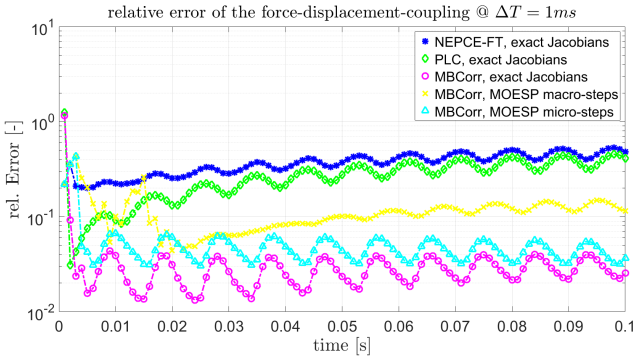
As can be seen in figure 5 the model-based corrector approach outperforms both NEPCE with feed-through correction and PLC across the range and produces only moderate errors even for relatively large step sizes until a certain macro-step threshold is reached. The overall NRMSE of several macro-step sizes is given in table 2 . For moderate macro-step sizes it can be seen that the use of system identification instead of the exact jacobians approximately doubles the error. Still, even with the use the MOESP algorithm, the error produced by the model-based correction approach is about five times less than the error produced by alternative coupling approaches. Furthermore it can be observed, that the results for the case with makro-steps only and the case with micro-steps included converge as the makro-step size is reduced. On the other hand if the macro-step size is increased, including the micro-

ΔT	$0.5ms$	$1.0ms$	$1.5ms$	$2ms$	$2.5ms$	$3ms$	$4ms$
NEPCE-FT	0.161	0.392	0.764	1.433	2.754	5.381	17.893
PLC	0.131	0.299	0.518	0.797	1.141	1.513	2.456
MBCorr exact	0.009	0.029	0.052	0.080	0.129	0.235	0.945
MBCorr macro-steps	0.031	0.114	0.116	0.467	0.484	0.734	4.026
MBCorr micro-steps	0.025	0.047	0.102	0.185	0.241	0.270	1.255

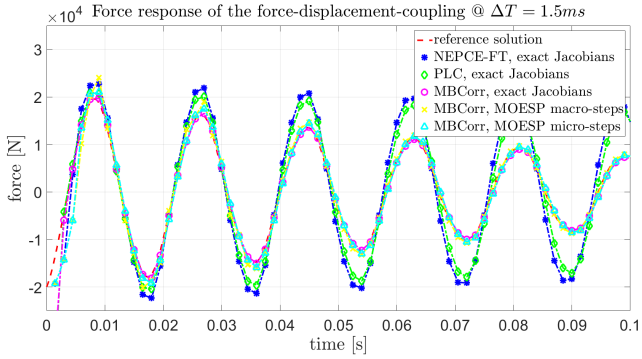
Tab. 2: NRMSE of the linear dual mass oscillator for different macro-step sizes



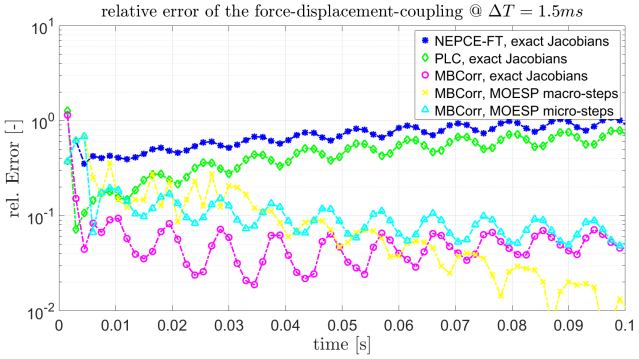
(a) Force response for $\Delta T = 1ms$



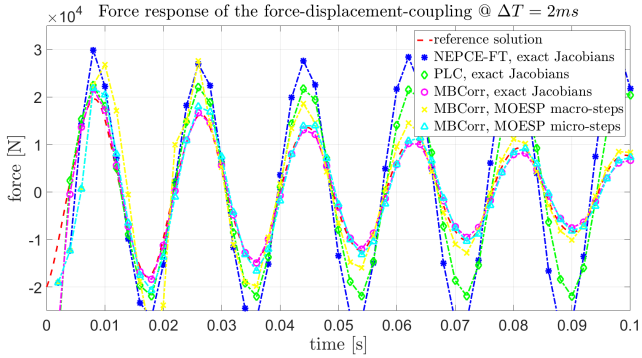
(b) Normalized error for $\Delta T = 1ms$



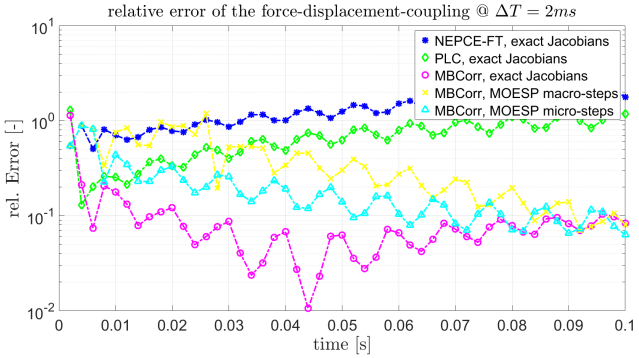
(c) Force response for $\Delta T = 1.5ms$



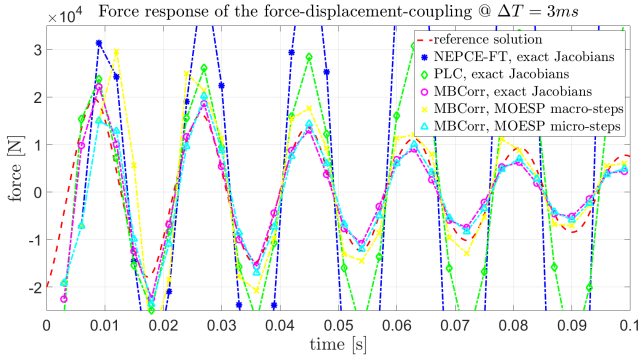
(d) Normalized error for $\Delta T = 1.5ms$



(e) Force response for $\Delta T = 2.0ms$



(f) Normalized error for $\Delta T = 2.0ms$



(g) Force response for $\Delta T = 3.0ms$

(h) Normalized error for $\Delta T = 3.0ms$

Fig. 5: force response and error for different coupling schemes and step sizes for the linear dual mass oscillator.

ΔT	$0.5ms$	$1.0ms$	$1.5ms$	$2ms$	$2.5ms$	$3ms$	$4ms$
NEPCE-FT	0.183	0.449	0.886	1.679	3.224	5.766	14.78
PLC	0.150	0.336	0.573	0.850	1.157	1.439	2.293
MBCorr exact	0.018	0.047	0.087	0.114	0.163	0.260	0.963
MBCorr macro-steps	0.032	0.206	0.105	2.886	0.442	0.826	4.221
MBCorr micro-steps	0.027	0.050	0.082	0.138	0.199	0.244	1.119

Tab. 3: NRMSE of the non-linear dual mass oscillator for different macro-step sizes

steps greatly improves the result and keeps the performance of the model-based corrector approach close to the maximum possible performance when the jacobians are known.

6.2 DMO with Non-Linear Damper Characteristic

In order to examine the effects of non-linearity, the damping constant d_k of the spring-damper element in subsystem II is now replaced by a variable damping factor $d_{k_{nl}}(t)$ given by

$$d_{k_{nl}}(t) = min\left(d_k\left(\left|x_2^{II} - u_2^{II}\right|\right)^{\frac{2}{(1+2n_d)-1}}, d_{k_{max}}\right) \tag{38}$$

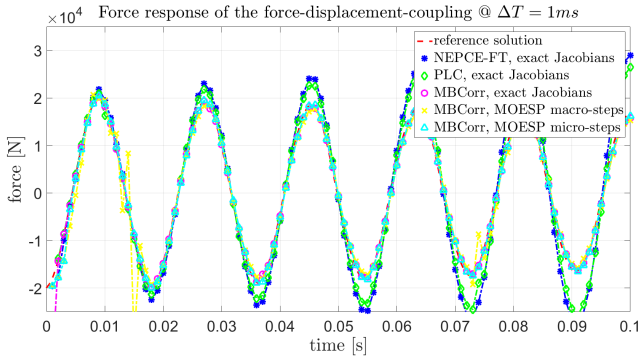
where the non-linearity factor n_d is set to 1.5. To avoid numerical problems if the velocities of the two masses are very similar, the damping factor is limited by $d_{k_{max}} = 1e5$. The remaining dual mass oscillator and the used parameters are left unchanged. The same macro-step sizes and co-simulation methods are used as with the linear dual mass oscillator. The force response and NRMSE is displayed in figure 6 .

Same as with the linear dual mass oscillator the model-based corrector outperforms NEPCE with feed-through correction and PLC, where the availability of micro-steps is especially valuable for larger macro-step sizes. The overall NRMSE of several macro-step sizes is given in table 3 . It can be seen, that for the non-linear test case the difference between the usage of the exact jacobians and the usage of system identification is not as evident as with the linear DMO. The convergence for the cases with and without usage of the micro-steps is still given for the non-linear test case. Likewise, the use of micro-steps is vital in order to get good results if the macro-step size increases.

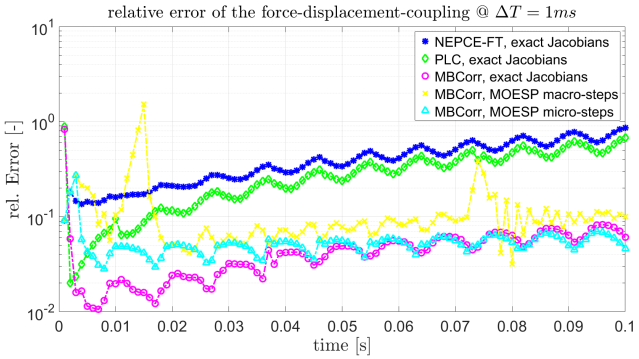
7 Conclusions and Outlook

In this paper a model-based correction algorithm for non-iterative co-simulation was derived and applied to a dual-mass oscillator with linear and non-linear spring-damper characteristic. Because the necessary jacobians of the subsystems are often not available, an algorithm based on MOESP subspace identification was introduced in order to estimate these jacobians from observed input/output data. As a result the overall coupling algorithm has minimal requirements in terms of tool interfaces, although the availability of micro-steps helps a lot with the system identification part, especially with non-linearity in the subsystems or relatively large macro-steps. It was shown that the model-based coupling algorithm performs well for all three modes of operation (exact knowledge of the jacobians, identification including micro-steps and identification using macro-steps only), for both linear and non-linear subsystems.

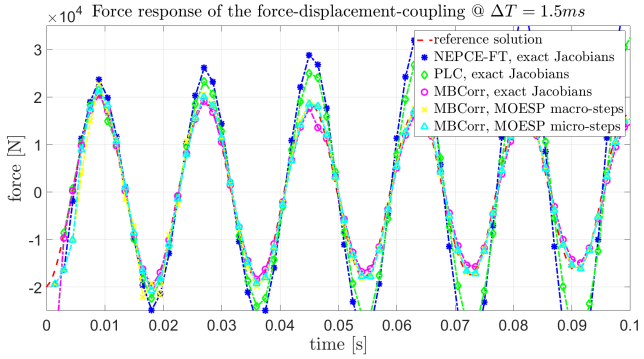
In the further process, the influence of non-linearity on the performance of the coupling algorithm as well as the influence of the identification window size will be investigated further. Finally, the coupling algorithm will be applied to real-life co-simulation problems from the industry.



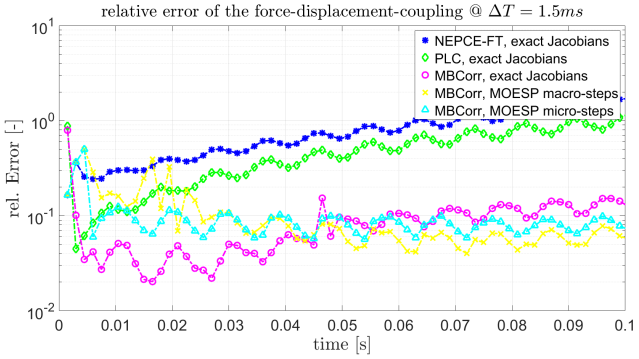
(a) Force response for $\Delta T = 1ms$



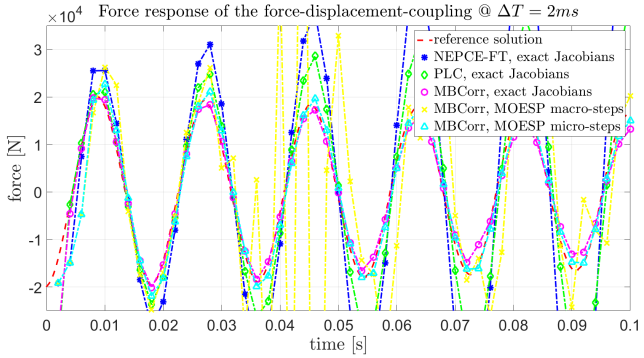
(b) Normalized error for $\Delta T = 1ms$



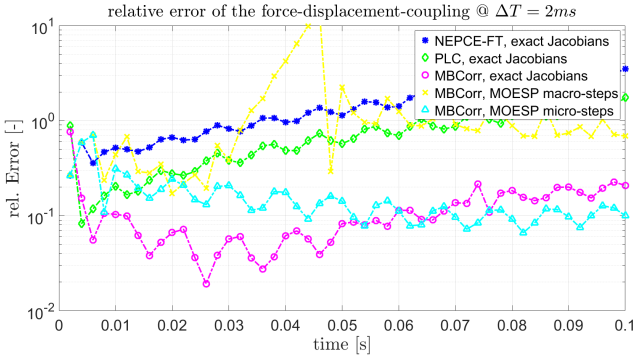
(c) Force response for $\Delta T = 1.5ms$



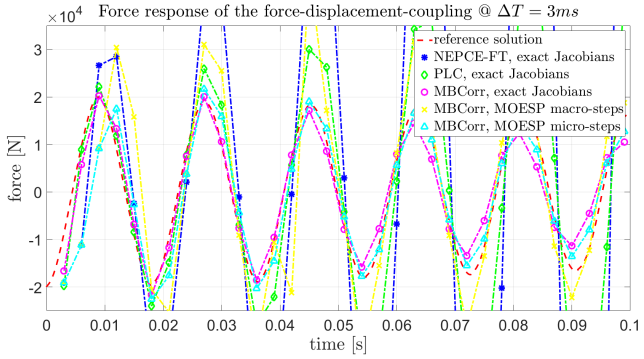
(d) Normalized error for $\Delta T = 1.5ms$



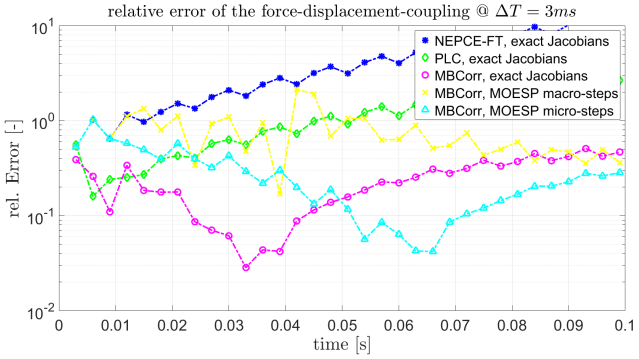
(e) Force response for $\Delta T = 2.0ms$



(f) Normalized error for $\Delta T = 2.0ms$



(g) Force response for $\Delta T = 3.0ms$



(h) Normalized error for $\Delta T = 3.0ms$

Fig. 6: Force response and normalized error for different coupling schemes and step sizes for the non-linear dual mass oscillator.

References

[1] M. Valasek, *Modeling, simulation and control of mechatronical systems*. 2008.

[2] R. Kübler and W. Schiehlen, "Modular simulation in multibody system dynamics," *Multibody System Dynamics*, vol. 4, no. 2-3, pp. 107–127, 2000.

[3] E. Lelarasmee, A. Ruehli, A., and L. Sangiovanni-Vincentelli, "The waveform relaxation method for time-domain analysis of large scale integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 1, pp. 131–145, 1982.

[4] M. Arnold and M. Gnther, "Preconditioned dynamic iteration for coupled differential-algebraic systems," *BIT Numerical Mathematics*, vol. 41, pp. 1–25, 2001.

[5] R. Kübler and W. Schiehlen, "Two methods of simulator coupling," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 6, no. 2, pp. 93–113, 2000.

[6] B. Schweizer, "Explicit and implicit co-simulation methods: Stability and convergence analysis for different solver coupling approaches," *Journal of Computational and Nonlinear Dynamics*, 01 2014.

[7] M. Benedikt, J. Zehetner, D. Watzenig, and J. Bernasch, "Moderne kopplungsmethoden - ist co-simulation beherrschbar?," *NAFEMS Online-Magazin*, vol. 22, pp. 63–74, 2012.

[8] S. Liebig, S. Helduser, M. Stwing, and S. Dronka, "Die modellierung und simulation gekoppelter mechanischer und hydraulischer systeme," 2001.

[9] M. Benedikt, D. Watzenig, and A. Hofer, "Modelling and analysis of the non-iterative coupling process for co-simulation," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 19, no. 5, pp. 451–470, 2013.

[10] M. Arnold, *Multi-Rate Time Integration for Large Scale Multibody System Models*, pp. 1–10. Dordrecht: Springer Netherlands, 2007.

[11] M. Busch, *Zur effizienten Kopplung von Simulationsprogrammen*. Kassel University Press, 2012.

[12] M. Friedrich, *Parallel Co-simulation for Mechatronic Systems*. PhD thesis, 2011.

[13] G. Stettinger, M. Horn, M. Benedikt, and J. Zehetner, "A model-based approach for prediction-based interconnection of dynamic systems," in *53rd IEEE Conference on Decision and Control*, 2014.

[14] B. Schweizer and D. Lu, "Semi-implicit co-simulation approach for solver coupling," *Archive of Applied Mechanics*, vol. 84, pp. 1739–1769, 12 2014.

[15] M. Arnold, "Numerical stabilization of co-simulation techniques the ode case," in *Working document MODELISAR, sWP 200 - 203*, 2011.

[16] C. Andersson, *Methods and Tools for Co-Simulation of Dynamic Systems with the Functional Mock-up Interface*. PhD thesis, 2016.

[17] S. Sadjina, "Energy conservation and coupling error reduction in non-iterative co-simulations," *arXiv.org*, 2016.

[18] Modelica Association Project "FMI", *Functional Mock-up Interface for Model Exchange and Co-Simulation, Ver. 2.0, July 25, 2014*, 2014.

[19] T. Schierz, M. Arnold, and C. Clauss, "Co-simulation with communication step size control in an fmi compatible master algorithm," in *Proceedings of the 9th International MODELICA Conference*, 2012.

[20] M. Verhaegen, "Identification of the deterministic part of mimo state space models given in innovations form from input-output data," *Automatica*, vol. 30, pp. 61–74, 1994.

[21] L. Ljung, *System Identification (2Nd Ed.): Theory for the User*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[22] W. Favoreel, B. DeMoor, and P. VanOverschee, "Subspace state space system identication for industrial processes," *Journal of Process Control*, vol. 10, pp. 149–155, 2000.

[23] P. V. Overschee and B. D. Moor, "N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75 – 93, 1994. Special issue on statistical signal processing and control.

[24] P. Trnka, "Subspace identification methods," 2005.

[25] T. Katayama, *Subspace Methods for System Identification*. Springer-Verlag London eBook ISBN 978-1-84628-158-7 DOI 10.1007/1-84628-158-X Softcover ISBN 978-1-84996-988-8 Softcover ISBN 978-1-85233-981-4 Buchreihen ISSN 0178-5354 Auflage 1 Seitenzahl XVI, 392 Anzahl der Bilder und Tabellen 66 schwarz-wei Abbildungen Themen Kommunikationstechnik und -netze, 1 ed., 2005.

[26] D. D. D. Ruscio, "Model based predictive control: an extended state space approach," in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 4, pp. 3210–3217 vol.4, Dec 1997.

[27] D. DiRuscio, "Telemark Institute of Technology, Porsgrunn, Norway, Lecture Notes: SUBSPACE SYSTEM IDENTIFICATION Theory and applications," 1995.

[28] K. J. Astrom and B. Wittenmark, *Computer Controlled Systems: Theory and Design*. Prentice Hall Professional Technical Reference, 1984.