ERROR ESTIMATORS FOR ADAPTIVE SCHEDULING ALGORITHM FOR SERIAL CO-SIMULATION

Emin Oguz Inci
Wim Desmet

DMMS Core Lab, Flanders Make
Mechanical Engineering, KU Leuven,
Celestijnenlaan 300, B-3001 Leuven, BELGIUM,
{eminoguz.inci, wim.desmet}@kuleuven.be

Cláudio Gomes

Dept. of Electrical and Computer Engineering,
Aarhus University, Åbogade 34
8200 Aarhus N, DENMARK,
claudio.gomes@ece.au.dk

Jan Croes

Forcebit NV,
Gezusters Grégoirestraat 5,
3010 Kessel-Lo BELGIUM
jan.croes@forcebit.eu

ABSTRACT

In serial co-simulation, the subsystems are solved and exchange their variables subsequently. For this reason, the solution sequence of the subsystems plays an important role. Prior work has demonstrated the effect of solution sequence on the accuracy of serial co-simulation and proposed an adaptive scheduling algorithm that predicts when to change the solution sequence using global input estimation error. This paper introduces another mechanism that considers physical causality to determine the computational causality in the co-simulation. We benchmark this method against input estimation error and static solution sequences in adaptive scheduling algorithm in serial co-simulations of a linear problem. The resulting method generally avoids the worst solution sequence and mitigates the risk of co-simulation being unstable if one of the sequences produces a greater error.

Keywords: co-simulation, solution sequence, causality, input estimation, state-space models.

1 INTRODUCTION

Co-simulation is a special kind of simulation where subsystems are decomposed (Kübler and Schiehlen 2000) based on their physical domain, modeling formalism or intellectual property concerns (Gomes et al. 2018). The main advantage of co-simulation is the partitioning of the solution of the subsystems as well as the decoupled the modeling it enables. By this way, the subsystems can be solved with dedicated solvers independently and exchange their input-output data when necessary at simulation run-time.

Serial co-simulations use the Gauss-Seidel algorithm to couple the interfacing subsystems in time (see Section 2). This is a weak coupling in time, where the solutions of subsystems exchange data and step sequentially. In (Inci et al. 2021), the authors demonstrate that the solution sequence of subsystems affects the accuracy of serial co-simulations. It is also demonstrated that an adaptive scheduler that predicts the best solution sequence at an interface can reduce co-simulation error. In that paper, the global input extrapolation errors was used to determine the computational causality.

Contribution. This paper proposes physical causality as a determining factor in the computational causality in co-simulation specialized for bond-graph modeling formalism. In bond-graph system models, the systems are connected with power bonds. Although these models are inherently acausal, we define computational causality as the system that transfers energy to other system at a certain time and interface. The physical causality is directed from the systems that does the work to the systems that receives it.

Structure. The following section introduces the running example and discusses prior related work. Then Section 3 describes our proposed extension. Section 4 discusses the results and Section 5 concludes.

2 BACKGROUND – ADAPTIVE CO-SIMULATION

2.1 Running Example

The system under study is the coupled mass spring damper system, which is a system typically used to evaluate novel co-simulation algorithms, as done in (González et al. 2019, Busch and Schweizer 2010, Li 2017). We omit the equations since these are not required to understand the contribution, but refer the reader to (Gomes et al. 2018, Section 4) for details on the example. The two mass-spring-dampers (MSD) are connected to rigid walls and coupled together with a spring-damper. Subsystem 1, MSD1, acts as an inert system by inputting the coupling force from Subsystem 2, MSD2, and outputting displacement and velocity to Subsystem 2. The parameters of the MSD1 are its mass m_1 (kg), stiffness of the spring connected to the rigid wall c_1 (N/m), and the damping constant of the damper connected to the wall d_1 (Ns/m). For MSD2 are the analogous parameters are m_2, c_2, and d_2. Furthermore, MSD2 also includes the stiffness of the spring coupling the two masses c_c (N/m), and the damping of the damper connecting the two masses d_c (Ns/m). As illustrates in the top left of Figure 1, the two systems are coupled by exchanging the time varying values of the displacement and speed of MSD1 (x_1, v_1) and the coupling force of MSD2 (F_k).

Table 1 summarizes the two parameter sets that we will be using throughout this manuscript, to illustrate a case where our contribution works well, and a case where it does not.

Table 1: Parameter sets used for co-simulation sequence tests.

Parameters	Units	**A**	**B**	**C**	**D**
m_1	[kg]	1	100	10	100
d_1	[Ns/m]	1	1	1	300
c_1	[N/m]	1	1	100	2e4
m_2	[kg]	1	1	10	1700
d_2	[Ns/m]	1	1	1	700
c_2	[N/m]	1	1	100	2e4
d_c	[Ns/m]	1	1	1	1
c_c	[N/m]	1	100	100	100

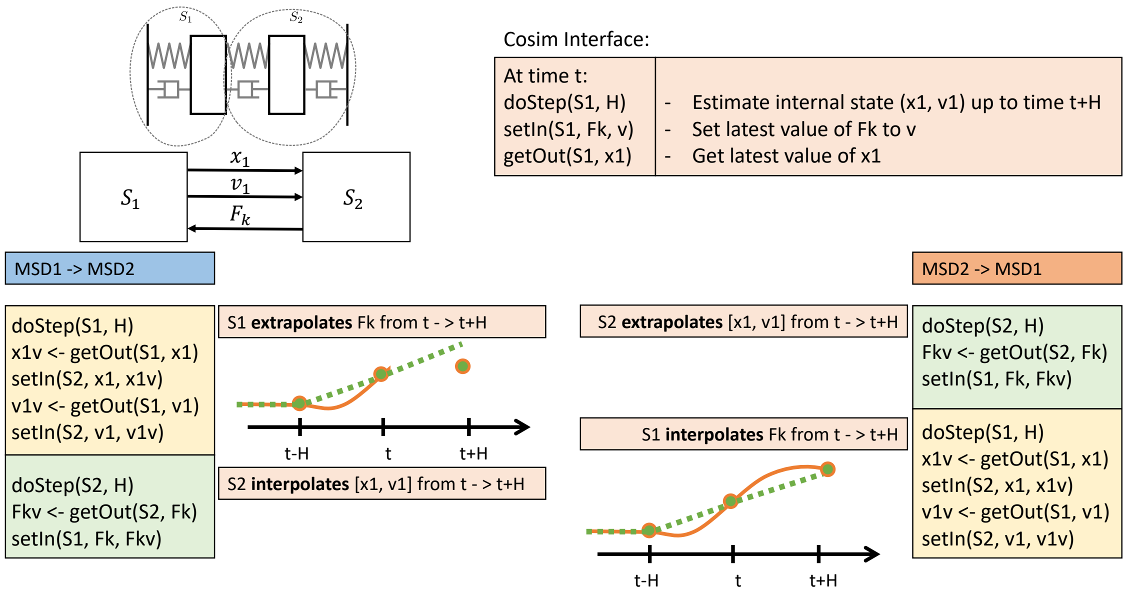


Figure 1: Example of the two orchestration algorithms studied in this manuscript.

2.2 Possible Orchestration Algorithms Under Study

Co-simulation consists of running simulations of each subsystem in an interleaved fashion, exchanging their partial results in each simulation step (Gomes et al. 2018). We refer the reader for a tutorial in (Gomes et al. 2018) for in-depth details. Each subsystem is responsible for its own simulation, exposing a simple API illustrated in the top right of Figure 1. A co-simulation orchestration represents the algorithm that realizes the co-simulation. This algorithm consists of calls to the functions exposed by each subsystem, in a certain order, to allow the simulation time to progress. The two possible algorithms studied in this manuscript are illustrated at the bottom of Figure 1, where only one co-simulation step is shown for each algorithm, and the co-simulation consists of the repeated application of the co-simulation steps. The communication interval, also known as the macro-step, is determined by the parameter H. Generally, the larger it is, the more simulation time is spent between data exchanges, which are achieved by the calls to the `getOut` and `setIn` function calls.

Each algorithm has advantages and disadvantages, since it determines which subsystem needs to perform extrapolation of the inputs given (since the inputs are exchanged as single real values for a specific time, but each subsystem needs to approximate continuous dynamics forward in time), as illustrated in Figure 1. Prior work in (Inci et al. 2021) as shown that the answer to “which subsystem should extrapolate?” varies over simulated time, motivating the need for adaptive orchestration algorithms, capable of changing the co-simulation sequence from one co-simulation step to the next.

2.3 Adaptive Co-simulation Algorithms

Figure 2 illustrates the block diagram of an adaptive algorithm, for the coupled MSDs example. It consists of the two coupled subsystems, along extra signal connections that are used to estimate quantities (discussed later in Section 3) that helps the `SwitchingDecision` block determine the best co-simulation sequence for the next co-simulation step. The next section focuses on which signals can be used to and how can the best co-simulation sequence be determined.

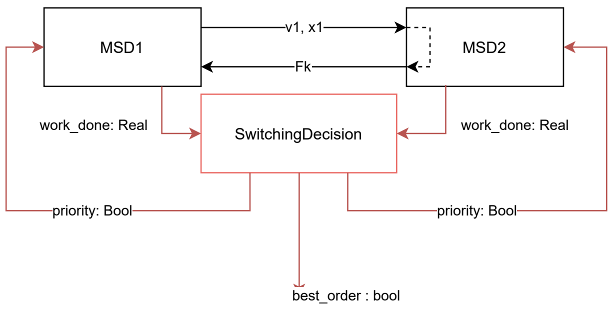


Figure 2: Adaptive sequence switching algorithm using power transfer.

3 POWER TRANSFER AS AN ERROR ESTIMATOR

In this section, we present error estimators in order to predict how the states are affected by the sequence selection. We propose two possible error estimators to determine the correct solution sequence of a co-simulation. The first hypothesis is using input estimation error as the error estimator. In Gauss-Seidel coupling, the system that is solved first extrapolates whereas the other connected system interpolates its inputs. In this hypothesis, we assume that the system to be solved first does the extrapolation error and the error done by the system that interpolates the inputs is negligible. By computing the extrapolation errors to be committed by each subsystem, the system with smaller extrapolation error must be run first. As the adaptive scheduling algorithm using this hypothesis was introduced in (Inci et al. 2021), it is not elaborated again in this paper.

The second hypothesis, and the contribution of this manuscript, is using causality as a co-simulation error estimator. The motivation here is to detect the subsystem that excites the other subsystem by transferring power at any time, t, and setting it as the first system to be solved. In this hypothesis, each macro-step is assumed to be a small simulation and the system that provides causality (effort or flow) must be solved first for each sub-simulation. The energy flow from one system to another is considered to determine the direction of the action and the reaction, therefore, the causality. The energy conservation at each subsystem must be considered:

$$\Delta(T+U) = \Delta E = W_{in} - W_{out} - L \tag{1}$$

where: T and U are the kinetic and potential energy of the system and their sum is the total energy of the system, E; ΔE is the change in total energy for a given time; W_{in} is the work done to the system and W_{out} is the work done by the system to the connected systems within this time frame; L is the dissipated energy to the environment in the form of heat, e.g. friction or damping. All the energy change, work done and dissipation are computed within communication intervals, $t \in [t_{i-1}, t_i]$, in this co-simulation study.

The power transferred to the connected system can be calculated by the difference between the work done by the system and work done to the system, $W = W_{out} - W_{in}$. For this reason, the energy loss of each subsystem is computed at every time interval to determine how much energy is transferred to the other subsystem, as is done in (Sadjina et al. 2017):

$$S_1 \rightarrow S_2: \qquad (W_1)_i = (E_1)_{i-1} - (E_1)_i - (L_1)_i, \tag{2}$$

$$S_2 \rightarrow S_1: \qquad (W_2)_i = (E_2)_{i-1} - (E_2)_i - (L_2)_i. \tag{3}$$

where $E_s = T_s + U_s$ is the total energy of the subsystem, s, at time t_i and i represents the $i-th$ co-simulation step. The energy of each system is calculated as

$$T_1 = \frac{1}{2}m_1{v_1}^2, \quad U_1 = \frac{1}{2}c_1{x_1}^2 \tag{4}$$

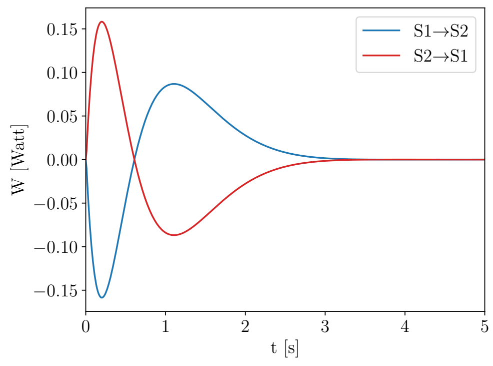
$$T_2 = \frac{1}{2}m_2{v_2}^2, \quad U_2 = \frac{1}{2}c_2{x_2}^2 + \frac{1}{2}c_c(x_2-x_1)^2 \tag{5}$$

and L_s is the energy dissipation at the subsystem due to damping between in the interval $[t_{i-1}, t_i]$. The energy dissipation at each subsystem is calculated as

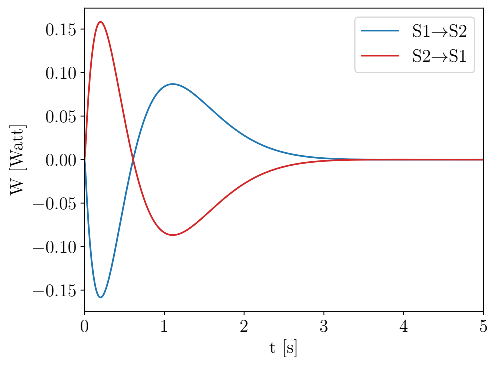
$$L_1 = \int |F_d|\,dx = \int |d_1 v_1|\,dx = \frac{1}{2}d_1({v_1}_i + {v_1}_{i-1})({x_1}_i - {x_1}_{i-1}) \tag{6}$$

$$L_2 = \frac{1}{2}d_2({v_2}_i + {v_2}_{i-1})({x_2}_i - {x_2}_{i-1}) + \frac{1}{2}d_c[(v_2-v_1)_i + (v_2-v_1)_{i-1}][(x_2-x_1)_i - (x_2-x_1)_{i-1}] \tag{7}$$

The validation of the power transfer is made by showing the energy inputted by a system is equal to the energy outputted by the other system at each time step, see power transfer for parameter set A in Figure 3. One can observe that the power transfer between systems is almost identical for both sequence algorithms. The sequence based on the power transfer between subsystems is represented in Figure 4. When it is compared with the actual sequence based on the filtered state error on v_1 in Figure 4a, the sequences do not match exactly. However, the impact of the change might be observed with a time delay based on the current dynamics of the system. Therefore, the error estimators are still used for developing an adaptive scheduling algorithm to mitigate the co-simulation error.

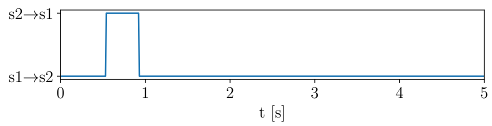


(a) MSD1 → MSD2.

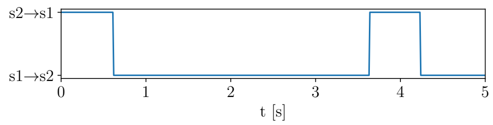


(b) MSD2 → MSD1.

Figure 3: Power transfer between systems for parameter set A.



(a) Actual sequence based on v_1 error.



(b) Co-simulation sequence based on power transfer estimator.

Figure 4: Error estimation using power transfer for parameter set A.

The adaptive scheduling algorithm with power transfer as an error estimator compares the work done by each subsystem to the connected subsystem to determine the solution sequence at run-time. With the power transfer criterion, the algorithm prioritizes the system with the positive work value to be solved earlier for the sake of causality. Therefore, the system that does the work is solved first and the system that receives energy is solved last:

$$\text{Solution Sequence at time } t_{i+1} = \begin{cases} S_1 \rightarrow S_2, & \text{if } (W_1)_i \geq (W_2)_i \\ S_2 \rightarrow S_1, & \text{otherwise} \end{cases} \tag{8}$$

where $(W_1)_i$ and $(W_2)_i$ are calculated in Equation (2) and Equation (3), respectively. The diagram of the adaptive sequence switching algorithm based on the power transfer is illustrated in Figure 2.

4 ADAPTIVE ALGORITHM RESULTS

The reader can find the code to reproduce the results presented in this paper in the online repository[1].

4.1 Selected Parameter Studies

Both adaptive scheduling algorithms are tested with the linear example introduced in Section 2.1 with the parameter sets in Table 1. The comparison of the errors obtained by the adaptive scheduling algorithms with static simulations is illustrated in Figure 5. We selected the final simulation time of 5 because that is the best trade of because the time it takes to run the simulation, and the richness of the behavior obtained. For positive damper constants, the system slows down sufficiently after 5 seconds as to not display interesting error behavior. Benchmark of the state errors can be found in Figure 6 for any time, t, of the simulation.

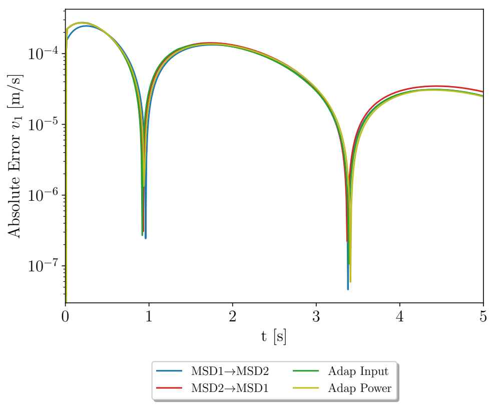
The adaptive simulations in these tests start with MSD1→MSD2 solution sequence. The algorithm does not start changing the initial sequence for 2 macro-steps to store enough inputs to perform linear extrapolation. And after the first 2 macro-steps, a cooldown period is allowed for the selected sequence whenever an algorithm switch is made in order to prevent zigzagging between two co-simulation sequences. As we use linear extrapolation for this example, the cooldown period is selected as a single macro-step.

For the given examples, both adaptive strategies seem to avoid the worst static sequence. Moreover, the adaptive scheduling algorithm generally performs better using the input estimation error estimator, than using the power transfer error estimator. However, this is not always the case, as illustrated in Figure 6b. In this figure, the adaptive scheduling with input estimation performs worse than the causal adaptive scheduling at the beginning of the simulation, and also performs worse than both static co-simulation algorithms for the entire simulation. The input estimation performs so poorly because the transient dynamics are too dominant for this example. Therefore, the switch that is done at the beginning of the simulation amplifies the error due to the change in input estimation algorithm (extrapolation → interpolation or vice versa) when the co-simulation sequence is changed from MSD1→MSD2 to MSD2→MSD1. Then, it accumulates throughout the simulation. The initial sequence for the power transfer is, on the other hand, not crucial, as the sequence selection criteria does not depend on the current sequence of the simulation (recall Figure 3).

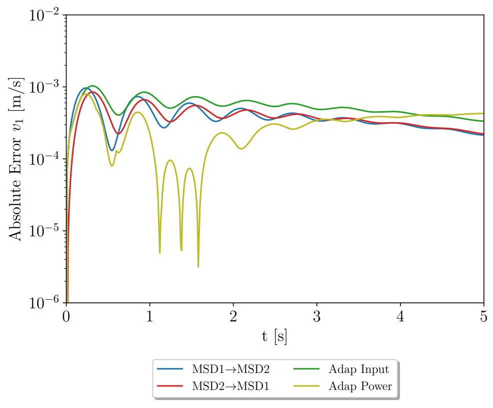
The effect of the initial sequence selection for adaptive scheduling algorithms is shown in Figure 7. The figure shows that the adaptive scheduling algorithm can obtain smaller errors compared to the static co-simulations provided that it starts from the correct initial sequence, which is typically not guaranteed.

We abstain from commenting on the impact of the co-simulation error, from the point of view of the modeler, because that depends on the requirements of the system.

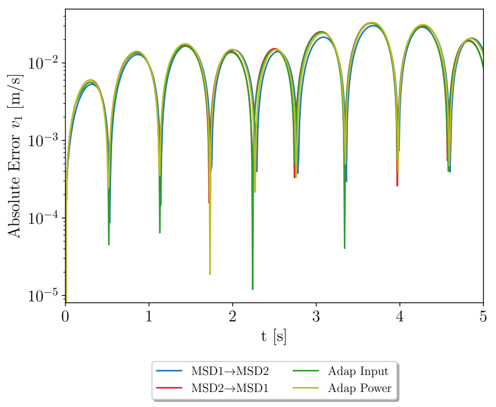
[1] https://github.com/clagms/2023.ANNSIM.ErrorEstimators.Reproducibility



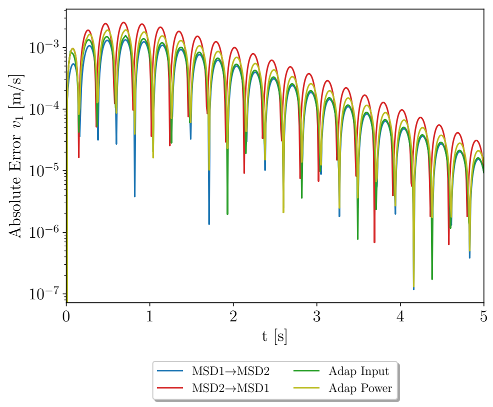
(a) Parameter set A.



(b) Parameter set B.



(c) Parameter set C.



(d) Parameter set D.

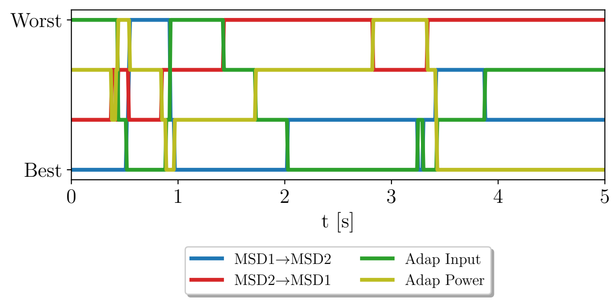
Figure 5: Error in state variable v_1 for adaptive scheduling algorithms.

4.2 Exhaustive Study

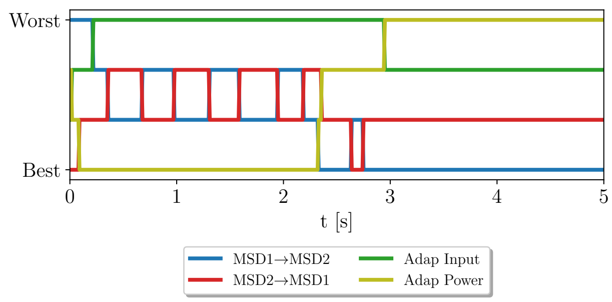
As was shown before, the systems parameters determine whether the algorithm performs well or not. In order to assess how the algorithm performs in practice, we therefore conducted an exhaustive parameter study. Each of the parameters was sampled from a uniform distribution 3 times, totaling $3^8 = 6,561$ experiments. The limits for each parameter are $[1,100]$ kg for masses (m_1, m_2), $[1,1000]$ Ns/m for dampers (d_1,d_2 and d_c) and $[1,1e4]$ N/m for springs (c_1, c_2 and c_c). Each experiment comprises four co-simulations: two with static alternative solution sequences, and two with the adaptive algorithm with alternative error estimators. The analytical error is computed once as a reference for the scoring.

The accuracy of the best co-simulation sequence selected by the adaptive algorithm at each time step is scored by the following formula

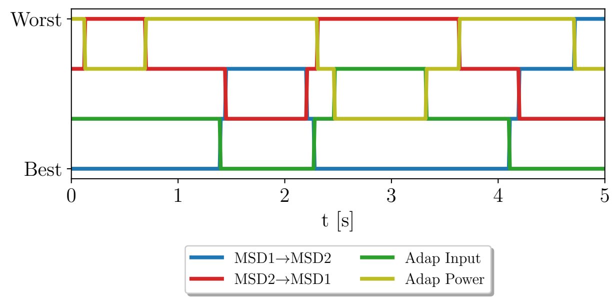
$$\begin{aligned}
(\varepsilon_{sol})_n &= \left|(v_{1\,sol})_n - (v_{1\,ref})_n\right| \\
(\varepsilon_{min})_n &= \min((\varepsilon_{sol})_n), \quad \text{where } sol = [\text{MSD1} \rightarrow \text{MSD2, MSD2} \rightarrow \text{MSD1, adap}] \\
(\varepsilon_{max})_n &= \max((\varepsilon_{sol})_n), \quad \text{where } ref = [\text{analytical}] \qquad (9) \\
\text{score} &= \text{mean}\left(\frac{(\varepsilon_{max})_n - (\varepsilon_{adap})_n}{(\varepsilon_{max})_n - (\varepsilon_{min})_n}\right).
\end{aligned}$$



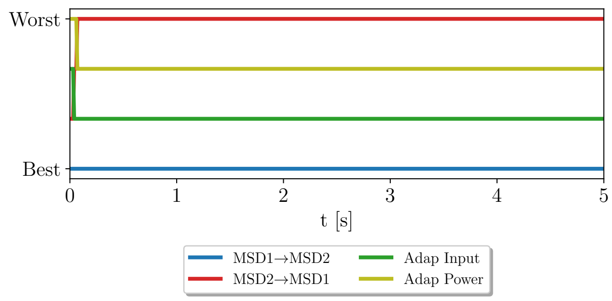
(a) Parameter set A.



(b) Parameter set B.



(c) Parameter set C.



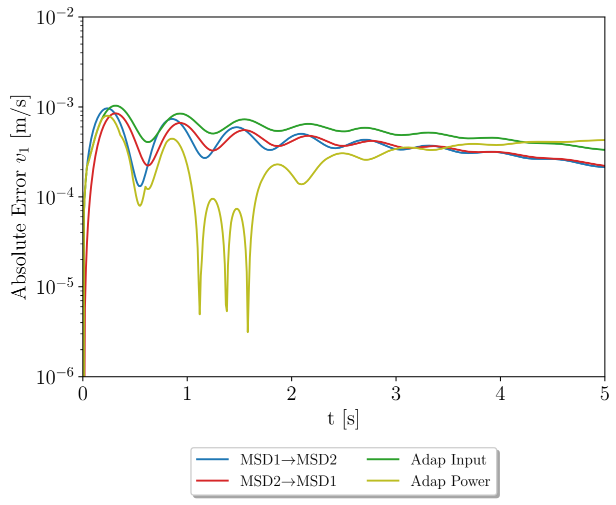
(d) Parameter set D.

Figure 6: Benchmark between adaptive and static co-simulation sequences.

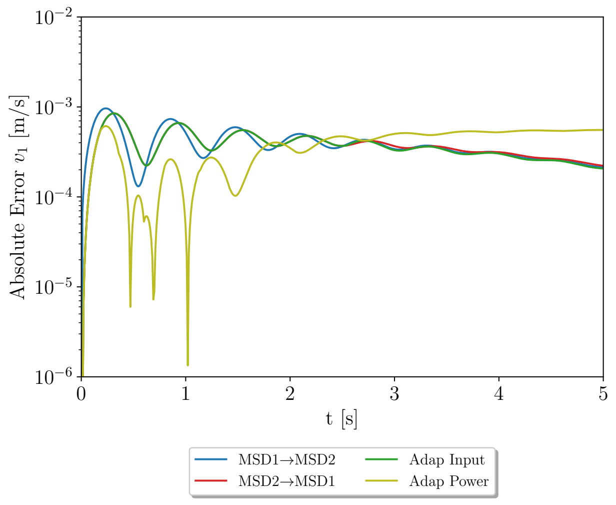
The scoring can be within [0,1]. The maximum score is 1 when the adaptive algorithm always selects *the best co-simulation sequence*. The minimum score is 0 when the adaptive algorithm always selects *the worst co-simulation sequence*. The score between (0,1) means that the adaptive algorithm performs *in-between* the two static algorithms in terms of accuracy. So, if the score is 0.5, the error obtained by the adaptive scheduling algorithm is exactly at the arithmetic mean of the two static sequence algorithms. Although the score represents the medium point, it cannot be achieved by zigzagging between the sequencing algorithms due to discontinuities introduced in input estimation functions between switches. Therefore, our criteria of success is to exceed this value.

The mean and median score of the adaptive scheduling algorithm is 0.62 and 0.67 for input estimation error; and 0.56 and 0.58 for power transfer, respectively. These results are obtained from the co-simulation where the initial sequence was MSD1→MSD2. The scoring of the results for starting from MSD2→MSD1 is 0.64 and 0.69 for input estimation error; and 0.54 and 0.57 for power transfer, respectively.

So far, it is shown how well the adaptive scheduling algorithms can select the best sequence with the parameter sweep study. However, it is required to be sure that the algorithm performs as intended (avoids the worst result) not by coincidence or by averaging the errors of two static algorithms. Therefore, we reverted the decision basis of each algorithm to select the worst sequence at a time. The mean and median scores of the simulations with reverse selection criteria are 0.28 and 0.24 for input estimation error; and 0.32 and 0.24 for power transfer criteria as error estimators. These results show that we can harm the accuracy of the simulation as well as heal. Therefore, it is fair to conclude that the adaptive sequence selection has a positive effect. By looking at the scores, we can also conclude that using input estimation error as co-simulation sequence selection criteria performs better than power transfer, although the margin between them in terms of accuracy is not significant.



(a) Initial sequence: s1 $\longrightarrow$ s2.



(b) Initial sequence: s2 $\longrightarrow$ s1.

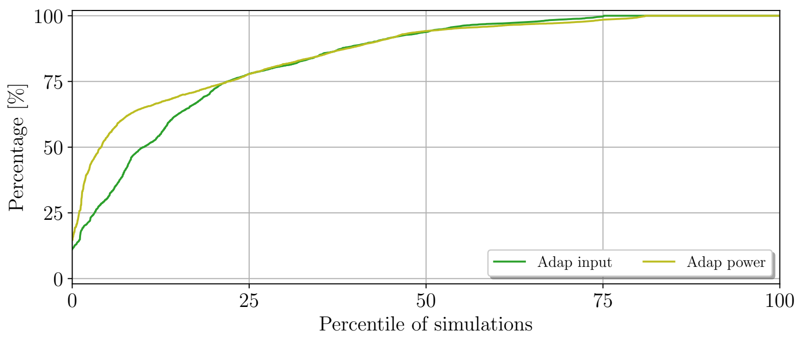
Figure 7: The effect of initial solution sequence for adaptive algorithms for parameter set B.

Table 2: Average CPU Time of the co-simulations obtained with single core simulation with Processor Intel(R) Core(TM) i5-7300U CPU @ 2.60GHz, 2701 Mhz, 2 Core(s), 4 Logical Processor(s).

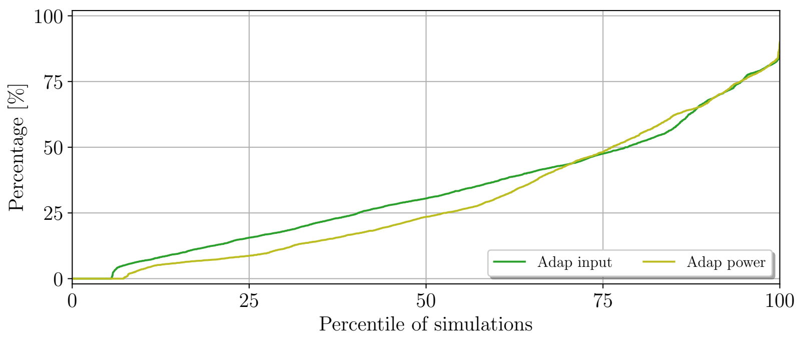
static sequence algorithms	adaptive algorithm with GIEE	adaptive algorithm with PT
14.57s	15.13s	15.06

The scoring in Equation (9) considers only the mean and median values over the simulation. However, the frequency of the adaptive scheduling algorithm being the best simulation or between them is also important. The frequency of the good or bad results throughout the simulation is also important as not all simulations might be run only for 5 seconds and the best and the worst sequence change over time. Therefore, it is required to measure how long can we avoid the worst static simulation (the primary objective of adaptive scheduling) or even further improve the best static simulation (the secondary objective of adaptive simulation).

Figure 8 compares the performances of the selection mechanisms for both objectives. Figure 8a shows that the adaptive scheduling algorithm performs the best or in-between throughout the simulation for 25% and 20% of the simulations for global input estimation error (GIEE) and power transfer (PT) methods, respectively. The same figure also indicates that adaptive scheduling avoids the worst sequence 75% of the simulation time for more than 75% of all simulations for both methods. However, the adaptive scheduling with power transfer avoids the worst sequence better in the remaining 25% of the simulations. The adaptive algorithm selects the worst solution sequence more than half of the simulation time only 4% of the simulations with PT whereas this figure is 11% for GIEE. For the random parameters selected for the parameter study, the adaptive scheduling algorithm performs the best throughout the simulation for none of the simulations, see Figure 8b. For the 50 percentile of the simulations, it selects the best sequence at around 30% and 25% of the simulation time for GIEE and PT, respectively. But both methods never select the worst sequence throughout the simulation either.



(a) Frequency of being the best or in-between co-simulation sequence.



(b) Frequency of being the best co-simulation sequence.

Figure 8: Frequency study for adaptive scheduling using input estimation error and power transfer.

5 CONCLUSION

This manuscript proposes an alternative error estimation procedure for determining the best order for an adaptive co-simulation algorithm, based on two subsystems. Compared to the input estimation error proposed in prior work, the exhaustive study on the linear numerical example shown that input estimation error selects the best solution sequence better. However, the power transfer method proposed here avoids the worst solution sequence more often, albeit the difference between the results not being very significant. Despite its limitations, the adaptive scheduling algorithm mitigates the risk of co-simulation being unstable if one of the sequences produces a greater error, e.g. due to dominant transient dynamics at one of the subsystems. Considering the negligible computational overhead, shown in Table 2, it is recommended to use the adaptive scheduling algorithm to ascertain the accuracy of a co-simulation.

Limitations & Future Work. As pointed out by the reviewers, we focus on coupled continuous systems. The reason is that it is in these systems that the ordering of the co-simulation algorithm can be varied. For software systems, the order is typically set according to the deployment scenario of the software. Nevertheless, our contribution can be applied to those parts of the system that are continuous, but is still restricted to pairs of systems. The exhaustive study in the current manuscript and the work in (Inci et al. 2021) show that it is challenging to select the best sequence, even for a very simple system. In the future, we will work towards generalizing the algorithm to more than two subsystems. This is a non-trivial task since it introduces trade-offs in the best ordering of the subsystems. Another interesting direction, as pointed out by a reviewer, is to create a more general definition of approximation error, than can be tailored to different kinds of models, which can then be fed into the algorithm proposed in (Inci et al. 2021).

REPRODUCIBILITY

A reproducibility package is available online[2].

REFERENCES

Busch, M., and B. Schweizer. 2010. "Numerical stability and accuracy of different co-simulation techniques: analytical investigations based on a 2-DOF test model". In *1st Joint International Conference on Multibody System Dynamics*, pp. 25–27.

Gomes, C., C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. 2018. "Co-Simulation: A Survey". *ACM Computing Surveys* vol. 51 (3), pp. 1–33.

Gomes, C., C. Thule, P. G. Larsen, J. Denil, and H. Vangheluwe. 2018. "Co-Simulation of Continuous Systems: A Tutorial". Technical Report arXiv:1809.08463, University of Antwerp, Belgium.

González, F., S. Arbatani, A. Mohtat, and J. Kövecses. 2019, jan. "Energy-leak monitoring and correction to enhance stability in the co-simulation of mechanical systems". *Mechanism and Machine Theory* vol. 131, pp. 172–188.

Inci, E. O., J. Croes, W. Desmet, C. Gomes, C. Thule, K. Lausdahl, and P. G. Larsen. 2021. "The Effect and Selection of Solution Sequence in Co-Simulation". In *2021 Annual Modeling and Simulation Conference (ANNSIM)*, pp. 1–12, SCS.

Kübler, R., and W. Schiehlen. 2000. "Two Methods of Simulator Coupling". *Mathematical and Computer Modelling of Dynamical Systems* vol. 6 (2), pp. 93–113.

Li, P. 2017. *On the Numerical Stability of Co-Simulation Methods*. Ph. D. thesis, Technische Universität Darmstadt.

Sadjina, S., L. T. Kyllingstad, S. Skjong, and E. Pedersen. 2017. "Energy conservation and power bonds in co-simulations: non-iterative adaptive step size control and error estimation". *Engineering with Computers* vol. 33 (3), pp. 607–620.

AUTHOR BIOGRAPHIES

EMIN OGUZ INCI is a research associate in Mechanical Engineering Department at KU Leuven. He holds an M.Sc. from Computational Mechanics at TU Munich and an B.Sc. from Mechanical Engineering at METU. His research interests include mechatronic design, simulation of multi-physical systems and design space exploration.

CLÁUDIO GOMES is an assistant professor at Aarhus University (Denmark). He received his PhD at the University of Antwerp, for his work on the foundations of co-simulation. His web address is https://pure.au.dk/portal/da/claudio.gomes@ece.au.dk.

JAN CROES is the co-founder of Forcebit NV and a guest researcher at KU Leuven. He predominantly focusses on application oriented research in the field of system level modelling of mechatronic drive lines in combination with model based sensing techniques.

WIM DESMET is a full professor at the Department of Mechanical Engineering and serve as the Managing Director of KU Leuven. His research specializes in mechatronic system dynamics and vibroacoustics. Professor Desmet has been chair of the IOF Council at KU Leuven since 2015. He also leads a research group at Flanders Make, the strategic research center for the manufacturing industry.

[2]https://github.com/clagms/2023.ANNSIM.ErrorEstimators.Reproducibility