Multibody Syst Dyn
DOI 10.1007/s11044-014-9415-x

# A model partitioning method based on dynamic decoupling for the efficient simulation of multibody systems

**Alessandro Vittorio Papadopoulos · Alberto Leva**

Received: 12 August 2013 / Accepted: 19 February 2014


**Abstract** The presence of different time scales in a dynamic model significantly hampers the efficiency of its simulation. In multibody systems the fact is particularly relevant, as the mentioned time scales may be very different, due, for example, to the coexistence of mechanical components controled by electronic drive units, and may also appear in conjunction with significant nonlinearities. This paper proposes a systematic technique, based on the principles of dynamic decoupling, to partition a model based on the time scales that are relevant for the particular simulation studies to be performed and as transparently as possible for the user. In accordance with said purpose, peculiar to the technique is its neat separation into two parts: a structural analysis of the model, which is general with respect to any possible simulation scenario, and a subsequent decoupled integration, which can conversely be (easily) tailored to the study at hand. Also, since the technique does not aim at reducing but rather at partitioning the model, the state space and the physical interpretation of the dynamic variables are inherently preserved. Moreover, the proposed analysis allows us to define some novel indices relative to the separability of the system, thereby extending the idea of "stiffness" in a way that is particularly keen to its use for the improvement of simulation efficiency, be the envisaged integration scheme monolithic, parallel, or even based on cosimulation. Finally, thanks to the way the analysis phase is conceived, the technique is naturally applicable to both linear and nonlinear models. The paper contains a methodological presentation of the proposed technique, which is related to alternatives available in the literature so as to evidence the peculiarities just sketched, and some application examples illustrating the achieved advantages and motivating the major design choice from an operational viewpoint.

**Keywords** Efficient simulation · Weak coupling · Multibody systems

A.V. Papadopoulos (✉)
Department of Automatic Control, Lund University, Lund, Sweden
e-mail: alessandro.papadopoulos@control.lth.se

A. Leva
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy
e-mail: alberto.leva@polimi.it

Published online: 03 April 2014

## 1 Introduction

Over the last years, modeling and simulation have been increasingly permeating the daily work of engineers. Simulation models are nowadays used to take decisions at virtually any stage of a project, and even to stipulate and mutually assess the behavior of parts created by different manufacturers before they are assembled; see, e.g., [4].

Such an evolution has two major consequences. On one side, model creation and management tools have been dramatically improved and at present allow one to construct extremely complex models on lightweight computational platforms, like, e.g., a laptop. On the other side, the need has become more and more strong for manipulation and solution tools that can run on the same platforms and treat those complex models efficiently enough. In other words, in the present *scenario*, the available computational resources often become the bottleneck of simulation-based studies, and therefore achieving an efficient integration of complex models becomes even more important.

The facts just mentioned are particularly relevant when addressing multibody systems, where many different time scales can be observed, and these very often appear together with significant nonlinearities. Also, in many engineering problems, a multibody system is not simulated alone, but rather in conjunction with some (fast) drive electronics and/or some (slow) phenomenon occurring in the process where the multibody system operates, e.g., a thermo-hydraulic plant.

This paper is part of a long-term research aimed at investigating how to manage the aforementioned complexity by devising model analysis, manipulation, simplification, and solution techniques that can be made part of modern modeling and simulation environments, in a view to achieve efficient integration as transparently as possible for the user.

Complexity—in the sense considered in the entirety of this research—can have different sources, the major ones being model dimension, nonlinearities, necessity of different modeling paradigms (e.g., equation- or algorithm-based), and presence of different time scales (i.e., stiffness).

In the literature, those kinds of complexity are addressed with different approaches. Large-scale systems are typically handled by means of model order reduction (MOR) techniques [1]. These are however essentially limited to the linear case, whereas nonlinear extensions are basically heuristic, domain specific, or scenario-based [17]. As for multi-paradigm models, advanced tools—typically object-oriented modeling languages—are inherently conceived to handle them, allowing one, for example, to combine equation and algorithm models [6]; also, cosimulation environments are available to cooperatively employ specialized simulation tools [2, 10]. Finally, the integration of systems with different time scales can be made more efficient by means of approximation techniques, such as the so-called dynamic decoupling [3].

Whatever the source of complexity is, here we take as the main goal of model simplification that of improving computational performance while respecting convenient precision/accuracy constraints for the specific simulation study at hand. In this respect, it is worth noticing that modern tools already allow us to apply some simplification techniques in quite an easy way. For example, environments like Matlab provide many well-established functions for linear MOR, e.g., `balred`. However, to the best of the authors' knowledge, virtually for all the other mentioned techniques, only problem specific solutions are available, and their full integration in modelling environments is still an open problem.

This paper deals with the exploitation of one of those techniques, namely the mentioned dynamic decoupling (DD). The proposed methodology is grounded on an analysis technique, described in the following, which is somehow analogous to eigenvalue analysis but applicable also to nonlinear systems.

The said technique, called *cycle analysis*, is the first contribution of this work. A second contribution, built on the cycle analysis idea, is the proposal of some indices to quantify the "separability" of a model into submodels based on DD. By jointly exploiting the said contributions, the following main advances are obtained beyond the state-of-the-art:

1. if a monolithic solution (i.e., no cosimulation) is required, cycle analysis provides evidence of possible internal weak couplings among dynamic variables, which can be exploited to ease the numerical integration; in the presence of a parallel computing architecture, this is apparently useful also for selecting the simulation threads;
2. if one (further) wants to apply integration schemes tailored to decoupled systems, these can be applied and configured on an objective basis according to structural properties of the model at hand;
3. if a cosimulation setting is considered and some degrees of freedom are available as for the model partitioning, these can be exploited automatically;
4. whatever solution setting is adopted, the proposed indices allow us to take any decision concerning its configuration based on quantities that are easily interpreted by the analyst.

Reference is here made to equation-based object-oriented modeling tools because they are particularly keen to be complemented with the proposed functionalities, but the proposed ideas are completely general and applicable also in different contexts.

The rest of the paper is organized as follows. In Sect. 2, a brief literature review on model simplification techniques is presented. In Sect. 3, the concept of DD is reviewed under a novel viewpoint, whereas Sect. 4 describes the proposed procedure for structural analysis, that is, the cycle analysis. Based on that method, Sect. 5 describes some new synthetic indices to characterize and quantify structural properties of the system, e.g., how much stiff or "separable" a system is, relating those quantities (when possible) to quantities already present in the literature. Section 6 describes how to exploit the results coming from the cycle analysis in a mixed-mode integration scheme, and in Sect. 7, some examples are presented and discussed. Some application-oriented remarks and more general discussion on the proposed method are reported in Sect. 8, whereas Sect. 9 concludes the paper.

## 2 Related work and contribution

In the context of this work, models are natively created in the form of acausal differential-algebraic equation (DAE) systems. The typical chain of operations of a modeling and simulation environment, which starts from the said native model description and ends with the simulation code, can be broadly divided into two parts.

The first part, which we call *acting on the continuous-time equations*, converts the acausal DAE system into a causal ordinary differential equation (ODE) system. This is done without altering the semantic of equations, by resorting to techniques such as the Tarjan algorithm, alias elimination, index reduction, and so forth [6]. The same operation can also be done by accepting some semantic alteration, that is, by altering the continuous-time equations, in exchange for an efficiency improvement. The major techniques for such a purpose are, MOR [1] and scenario-based approximations [16, 17].

The second part, which we call *acting on the discrete-time solution*, consists of taking the ODE model as the basis to generate routines that—once linked to the numeric solver of choice—provide the simulation code. Assuming that acting on the discrete-time solution is done "correctly", that is, preserving numerical stability, also in this case two ways of operating can be distinguished. The first one does not alter the solution semantic, applying

the chosen discretization method as is. In this case, errors in the solution only come from the inherent imperfection of that method. The second way conversely alters the semantic by deliberately deviating from the natural application of the discretization method. Notice that most cosimulation techniques naturally fall into the second class (see, e.g., [10]).

In this paper we concentrate on the last way of operating, for which DD [3, 19] is a powerful technique, albeit not fully exploited in a structured (thus possibly automated) manner. For the purpose of this section, suffice to say that this technique aims at partitioning a model into submodels based on time-scale separation. The method is particularly of interest—as will be better detailed in Sect. 3—because it can be divided into two well-separated phases: an analysis part performed on the overall model and a simulation part that can either be monolithic or make use of cosimulation.

To motivate the choice of focusing on DD, a brief discussion on the major possible alternatives is in order. As already stated, among the techniques that act on the continuous-time equations, MOR ones are the most adopted, and there exists a vast literature on the matter. MOR is based on the idea of approximating a certain part of the high-dimensional state space of the original model with a lower-dimensional state space by performing a projection. Roughly speaking, the main differences among MOR techniques come from the way the projection is performed. In any case, most MOR techniques were developed for linear systems [1], and this hampers their application to complex physical cases, where high dimensions often appear in conjunction with nonlinearities.

In fact, developing effective MOR strategies for large nonlinear systems is quite a challenging and relatively open problem [11]. Some proposals can be found in the literature, based, for example, on linearization or Taylor expansion [8], bilinearization [20], or functional Volterra series expansion [12], followed by a suitable projection. Other proposals worth mentioning are those based on proper orthogonal decomposition (POD) [7], to produce approximate truncated balanced realizations for nonlinear systems [21], often to find approximate Gramians [15]. However, the former type of MOR extensions for the nonlinear case are in practice stuck to quadratic expansions, which strongly limits their applicability. As for the latter type, the cost of evaluating the projected nonlinear operator is often quite high, which reduces computational performance.

Recently, works specifically targeting the reduction of object-oriented models have appeared [16, 17]. The main idea is that one can define some operation to be performed on the nonlinear system—for example, "neglect a term", "linearize a part of the model", and so on—and use some ranking metrics to identify a priori which is the "best" (single) manipulation that can be done on the model. Apparently, the limit of this approach is that ranking all the possible manipulation combinations is not feasible—in fact, the authors try to find out some other heuristics, such as clustering techniques, to reduce the combinatorial part of the approach. Moreover, there is no guarantee that performing the manipulations in the ranked order will eventually lead to the optimal manipulation since they are considered one at a time. Another problem is the high cost of generating the reduced-order models due to necessity of computing "snapshots" in the time domain, that is, simulations of the reduced model to check whether a given error bound is fulfilled, which in turn requires performing numerous simulations of the original nonlinear system. Furthermore, this approach is scenario-based, that is, the simplified model is guaranteed to be good—and the error within the error bound—only for a set of initial conditions, a set of inputs and a time span. If the scenario is changed, the overall manipulation must be performed again, limiting again the applicability of the method.

The quite old idea of DD has thus been recently reconsidered, for example, by the transmission line modeling (TLM) approach of [23, 24]. TLM is based on modeling the propagation of a signal that is limited by the time it takes to travel across a medium. By utilizing

this information it is possible to partition the DAE system into independent blocks that may be simulated in parallel. This leads to improved simulation efficiency since it enables full performance of multicore CPUs. However, it requires the analyst to explicitly introduce the transmission model, that is, the decoupling part, by introducing some additional components based on his/her intuition.

Based on the previous discussion, we now spend some additional words on the advantages of the technique proposed in this work, and sketched out in the introduction, with respect to the analyzed alternatives.

In comparison with MOR, our proposal does not alter the state vector, nor does it involve base changes in the state space, thereby preserving the physical meaning of dynamic variables. Also, instead of attempting to simplify the model in a view to a monolithic solution only, we go exactly in the opposite direction, as the model is not reduced but *partitioned*, with the same *rationale* of [24].

Of course, our proposal is not the only way to partition a system. As an alternative, for example, one may cut the subspace spanned by the eigenvectors associated with its fast eigenvalues. However, this is possible only in the linear case, whereas extensions to nonlinear models require local linearization. This does preserve the dimension of the state space, but to recover the native dynamic variables of the model, a coordinate transformation is necessary at each integration step, to the apparent detriment of simulation efficiency.

No matter how the partition is obtained, it then can be exploited in two ways. One is to ease a monolithic solution, in some sense adapting the model to the used architecture (single solver with a unique integration step). The other is to conversely tailor the solution architecture to the model *as analyzed and partitioned by the method*; this can be used to fruitfully employ parallel simulation or even cosimulation. If the latter route is taken, eigenvalue-based partitioning reveals however another problem since the properties of a so obtained partition may change in time, whereas decoupled integration, let alone cosimulation, requires the same partition to be specified a priori.

As a consequence, for the specific purpose of this work, state selection criteria are preferable to eigenvalue-based ones, also in accordance with [22], and in this context the proposed method exhibits the further advantage of being naturally keen to a nonlinear context.

With respect to scenario-based approximations, the most computing-intensive part of the proposal (as will be explained later on) is simply not scenario-based: information related to the considered *scenarios* come into play only at a later stage, and this separation results in lightening the computing effort. Furthermore, the proposal does not alter the model equations, thus being less exposed to the possible unpredictable effects of local modifications at the overall system level.

Finally, contrary to the TLM approach, this work aims at having decoupling emerge from an automated analysis of the model, and not introduced by the analyst, still having the advantage of exploiting full multicore CPUs performances by parallel simulation.

The contributions of this work can thus now be better qualified as follows.

- Cycle analysis quantitatively characterizes the dynamics of the addressed system, including the numerical integration algorithm, without resorting to eigenvalue-based techniques, therefore applying to both linear and nonlinear cases.
- Some "separability indices" are defined, whose information content extends beyond that of previously introduced quantities, like stiffness coefficients. The proposed indices thus complement traditional "stiffness" measures in basically two senses: (a) they are not tied to the sole idea of "fast" and "slow" dynamics, and (b) they apply also to nonlinear systems.

– The two ideas above are suitably joined to demonstrate, with a proof-of-concept application and some examples, that they can be used to achieve an automatic application of DD, that is, to build a tool that partitions a model requiring the analyst to provide only information that pertains to the physics of the simulated object.

## 3 Dynamic decoupling

Multiphysics models are often made of parts evolving within different time scales, and the core idea of DD is to exploit this partition to enhance simulation efficiency.

In some cases, figuring out how to partition a model can be quite straightforward, but this is not general at all. For example, in mechatronic systems, a "slow" mechanical part is often driven by "fast" electric circuits. However, even if this is the case, characterizing the found time scales quantitatively—for example, to determine whether or not it is really convenient to partition the model, and how to do it—may not be equally simple since the actual evidence of multiple time scales may not only come from the presence of multiple physical domains, but also strongly depend on parameter values. Furthermore, there are cases in which multiple time scales are not originated by multiple physical contexts, but emerge from some structural characteristics of the model that are virtually impossible for the analyst to detect a priori, especially for large models.

As a result, DD is formally based on some characteristics of the mutual relationships among the model state variables that are formulated in an abstracted manner with respect to the underlying physical domain(s). For a short explanation of the DD rationale, consider the generic state equation of a continuous-time ODE model and write it as

$$\phi_i(\mathbf{x})\frac{dx_i(t)}{dt} = \gamma_i(\mathbf{x}, \mathbf{u}), \tag{1}$$

where the function $\phi_i$ plays the role of a time-varying "capacitance" associated with the state variable $x_i$, whereas the function $\gamma_i$ conveys the contributions of all states (including $x_i$) and inputs (variables $\mathbf{u}$) to its variation. Given this, DD can be synthetically expressed as the following two principles.

1. If, in a certain region of the state and input space, some $\gamma_i/\phi_i$ ratio is "small," then in the discrete-time solution it can be acceptable to use the value of $x_i$ computed at the previous integration step, given its "slow" variation.
2. If, in a certain region of the state and input space, the contribution of a certain $x_j$ to $\gamma_i$ is "small," then in the discrete-time solution it can be acceptable to use the value of $x_j$ at the previous integration step, given the "small" error introduced in the computation of the new $x_i$.

The two principles above take different forms in various contexts (see, e.g., [3] for a thermo-hydraulic application) but are per se general. From an operational viewpoint, DD can be thought as composed of two subsequent phases, termed here *structural analysis* and *decoupled integration*. The former is an *offline* activity and consists of identifying in the model possible occurrences of the two principles above. The latter consists of exploiting the analysis outcome to select and suitably configure an integration scheme so as to improve simulation efficiency.

Both phases can be carried out with multiple techniques. For the structural analysis phase, we propose here a novel method, called cycle analysis (CA), described in the following, that

is particularly suited to investigate mutual relationships among dynamic variables independently of the structure of the individual state equations and therefore carries most of the merit for the applicability of the entire technique to the nonlinear case. For the decoupled integration phase, we conversely resort to mixed-mode integration similar to the one proposed in [22], but any cosimulation framework can be used, for example, the one proposed in [10].

A very important point to keep in mind is that pursuing an automatic application of DD is a twofold problem. On one side, the analysis phase needs to be performed by an automatic procedure rather than manually. On the other side, the outcome of the said phase must take a form that is readable for the analyst, who is typically an expert of the addressed physical domain, not of simulation. Such an output is carried out by means of a set of *separability indices*. The following sections thus deal, in this order, with CA, with the correspondingly obtained separability indices and with the use of both for decoupled integration.

## 4 Cycle analysis

### 4.1 Preliminaries and definitions

Consider the generic ODE system

$$\dot{\mathbf{x}}_{CT}(t) = \mathbf{f}\big(\mathbf{x}_{CT}(t), \mathbf{u}_{CT}(t)\big), \tag{2}$$

where $\mathbf{x}_{CT} \in \mathbb{R}^{n_{CT}}$ is the vector of state (i.e., dynamic) variables, and $\mathbf{u}_{CT} \in \mathbb{R}^{m_{CT}}$ is the vector of input variables. Generally speaking, the idea of CA is to obtain from the discretization of (2) a directed graph representing the mutual influence among the dynamic variables along the integration steps and then to compute quantities that generalize—in a sense that will be explained later—the idea of "time constants" for the linear case.

To this end, discretize (2) with any *explicit* method with fixed time step $h$.[1] It is important to notice right from now that the method used in this phase is a "probe method," that is, just functional to the analysis technique, whereas the successive simulation phase is in no sense tied to it. The corresponding discrete-time system can be thus written as

$$\mathbf{x}_{k+1} = \mathbf{F}_{\mathscr{N}}(\mathbf{x}_k, \mathbf{u}_k, h), \tag{3}$$

where $\mathbf{x}_k^T \in \mathbb{R}^n$ with $n = n_{CT}$ is the discrete-time state, whereas the form of the function $\mathbf{F}_{\mathscr{N}}(\cdot,\cdot,\cdot)$ depends on the particular numerical integration method $\mathscr{N}$.

The required dependency directed graph (or digraph) $G$ is formally defined as

$$G = (N, E), \quad N = \{1, \ldots, n\}, \quad E = \left\{e^{i,j}\right\} \subseteq N \times N. \tag{4}$$

The nodes of $G$ are associated with the discrete-time model dynamic variables, whereas its edges are characterized by a source node, a destination node, and a weight defined by the operators

$$\varsigma\left[e^{i,j}\right] := i, \qquad \delta\left[e^{i,j}\right] := j, \qquad \rho\left[e^{i,j}\right] := \frac{\partial F_i}{\partial x_j}. \tag{5}$$

Notice that the construction of $G$ is straightforward based on the structure of system (3).

[1] It is known that any multistep method can be reduced to a single-step method with an increased state space vector. Thus, in this paper we focus only on the case of single-step methods without loss of generality.

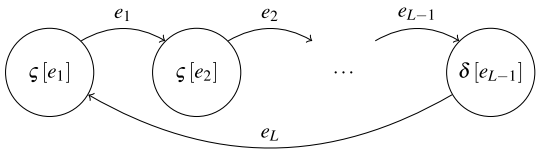


**Fig. 1** Graphical representation of a simple cycle

**Definition 1** A *path* $p$ of length $L$ in a digraph $G = (N, E)$ is an ordered sequence of $L$ edges, where the destination node of each edge is the source node of the following one in the sequence. Formally,

$$p := \langle e_1, e_2, \ldots, e_L \rangle \quad \text{with } e_i \in E \quad \forall i \in \{1, \ldots, L\},$$
$$\text{with } \delta[e_i] = \varsigma[e_{i+1}] \quad \forall i \in \{1, \ldots, L-1\}.$$

A path can be also denoted by means of the ordered sequence of touched nodes, that is,

$$p = \big\langle \varsigma[e_1], \varsigma[e_2], \ldots, \varsigma[e_L], \delta[e_L] \big\rangle.$$

**Definition 2** A path with no repeated nodes is called a *simple path* (or *walk*).

**Definition 3** A *simple cycle* $c$ of length $L$ exists in a digraph $G = (N, E)$ if and only if

1. there exists a simple path $\langle e_1, e_2, \ldots, e_{L-1} \rangle$,
2. there exists one edge $e_L$ from $\delta[e_{L-1}]$ to $\varsigma[e_1]$.

For the sake of clarity, a simple cycle can be graphically represented as shown in Fig. 1. Adopting the same notation used for paths, a simple cycle can be denoted as

$$c = \big\langle \varsigma[e_1], \varsigma[e_2], \ldots, \varsigma[e_{L-1}], \delta[e_{L-1}], \varsigma[e_1] \big\rangle,$$

that is, by listing the ordered sequence of the touched nodes.

Notice that the definition of a simple cycle in terms of edges is unique up to a circular permutation, whereas the definition in terms of touched nodes varies according to which of them is (conventionally) taken as the "first" one in the cycle. This is why we prefer to use the definition in terms of edges.

In the following we shall make reference only to simple cycles, and thus "cycle" and "simple cycle" will be used interchangeably.

**Definition 4** The *cycle gain* $\mu_c(h)$ of a cycle $c$ is defined as

$$\mu_c(h) = \prod_{e_i \in c} \rho[e_i]. \tag{6}$$

*4.1.1 An explanatory example*

Let us consider the continuous-time linear time-invariant dynamic system

$$\dot{\mathbf{x}}_{CT} = A\mathbf{x}_{CT} = \begin{bmatrix} -1 & 0.5 & 0 \\ 0.5 & -1.5 & 0.5 \\ 0 & 0.5 & -1 \end{bmatrix} \mathbf{x}_{CT}.$$

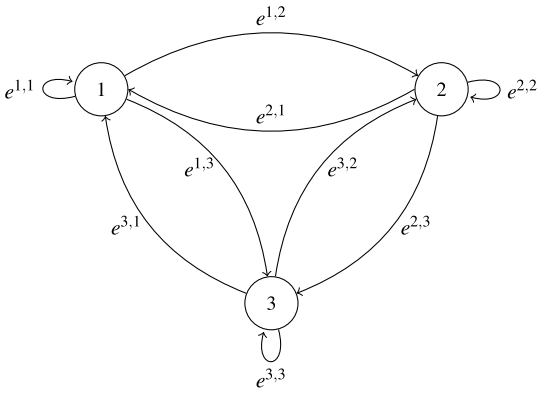


**Fig. 2** Dependency graph associated with the discretized system (7)

Suppose that the discretization of choice for the analysis part is the Heun algorithm [6]. Thus, the corresponding discrete-time system (3) becomes

$$\mathbf{x}_{k+1} = \mathbf{F}_{\text{Heun}}(\mathbf{x}_k, h), \quad \mathbf{F}_{\text{Heun}}(\mathbf{x}_k, h) = \left( I_{3\times 3} + Ah + \frac{(Ah)^2}{2} \right) \mathbf{x}_k, \tag{7}$$

where $I_{3\times 3}$ is a $3 \times 3$ identity matrix, and $\mathbf{x} = \mathbf{x}_{CT}$. Therefore, the dependency graph $G$ associated to the system has the weight matrix

$$W = \frac{\partial \mathbf{F}_{\text{Heun}}}{\partial \mathbf{x}} = I_{3\times 3} + Ah + \frac{(Ah)^2}{2} = I_{3\times 3} + \frac{h}{8} \begin{bmatrix} 5h-8 & 4-5h & h \\ 4-5h & 11h-12 & 4-5h \\ h & 4-5h & 5h-8 \end{bmatrix},$$

yielding a completely connected graph represented in Fig. 2.

In this case, the set of simple cycles $\mathscr{C}$ present in the graph $G$ and the corresponding cycle gains are

$$\begin{aligned}
c_1 &= \langle e^{1,1} \rangle, & \mu_{c_1}(h) &= \frac{5h^2}{8} - h + 1, \\
c_2 &= \langle e^{2,2} \rangle, & \mu_{c_2}(h) &= \frac{h}{8}(11h - 12), \\
c_3 &= \langle e^{3,3} \rangle, & \mu_{c_3}(h) &= \frac{h}{8}(5h - 8), \\
c_4 &= \langle e^{1,2}, e^{2,1} \rangle, & \mu_{c_4}(h) &= \frac{h^2}{64}(4 - 5h)^2, \\
c_5 &= \langle e^{1,3}, e^{3,1} \rangle, & \mu_{c_5}(h) &= \frac{h^4}{64}, \\
c_6 &= \langle e^{2,3}, e^{3,2} \rangle, & \mu_{c_6}(h) &= \frac{h^2}{64}(4 - 5h)^2, \\
c_7 &= \langle e^{1,2}, e^{2,3}, e^{3,1} \rangle, & \mu_{c_7}(h) &= \frac{h^4}{512}(4 - 5h)^2, \\
c_8 &= \langle e^{1,3}, e^{3,2}, e^{3,3} \rangle, & \mu_{c_8}(h) &= \frac{h^4}{512}(4 - 5h)^2.
\end{aligned}$$

Notice that if the matrix $W$ is symmetric, it is sufficient to consider only its lower triangular part (including the diagonal).

### 4.2 The analysis technique

As anticipated, the ultimate goal of the analysis is to (automatically) recognize the presence in the model of different time scales and cluster the dynamic variables accordingly. The underlying rationale of the approach is based on a convenient interpretation of the cycle gains of Definition 4.

To provide this interpretation, let us consider system (3) at an asymptotically stable equilibrium, that is, $\mathbf{x}_{k+1} = \mathbf{x}_k$. Suppose to apply a small impulsive perturbation to one state variable $x_i$. A transient will then occur, and two things may happen:

- the perturbation affects the other state variables, without reaffecting $x_i$, that is, in the associated model digraph $G$, there is no cycle involving node $i$;
- the perturbation, after some integration steps, reaffects $x_i$, that is, there exists at least one dependency cycle involving node $i$.

In the first case, no numerical instability can be introduced by the integration method. This is conversely possible in the second case and occurs if the perturbation undergoes a sufficient amplification along at least one of the involved cycles. Since that amplification is quantified by the corresponding cycle gain, we can conjecture that the perturbation vanishes if all the gains of the involved cycles are in magnitude less than a certain $\underline{\mu}$, whereas instability arises if at least one of the said gains is larger in magnitude than a certain $\overline{\mu} > \underline{\mu}$.

It is now worth recalling that, considering an ODE system at a certain asymptotically stable operating point, in the vicinity of the said point (i.e., near enough to it for the linearization of the original system to be sufficiently precise) there exists one value of $h$ that constitutes the boundary between a stable and an unstable behavior of the discrete-time solution.

It is also well known that with explicit methods, instability originates from model dynamics that has a fast time scale with respect to the employed integration step. Since the cycle gains depend on $h$, if an unstable behavior is observed, it is legitimate to state that the dynamic variables involved in the cycles that provide the excessive amplification are evolving with a time scale that is "fast" with respect to $h$.

Based on the discussion above, we can now describe the analysis procedure as follows.

1. Select an explicit fixed-step integration method. It is worth stressing that this method is only functional to the analysis and in no sense constrains the choice of the method(s) used for the subsequent decoupled integration.
2. Discretize the system.
3. Construct the digraph.
4. Perform a topological analysis to find the set $\mathscr{C}$ of all the (simple) cycles. The potential complexity of the cycle search operation will be discussed later on.
5. Express the cycle gains as per (6).
6. Construct a set of inequalities in the form

$$\left|\mu_c(h)\right| \leq \alpha \quad \forall c \in \mathscr{C}, \tag{8}$$

where $\alpha$ is the single real parameter of the analysis, to be discussed in the following.

7. Solve each inequality individually for $h$, thereby associating with each cycle a value for the integration step that produces a low enough magnitude of the corresponding gain.
8. Associate each variable $x_i$ with the most restrictive constraint $h_{x_i}$ on $h$ among the set of cycles $\mathfrak{C}_{x_i} = \{c \in \mathscr{C} | x_i \in c\}$, that is, find the maximum value of $h$ that fulfils all the constraints (8) on the cycle gain associated with $x_i$. Formally,

$$\begin{aligned} \overline{h}_{x_i} = \max \quad & h \\ \text{s.t.} \quad & h > 0, \\ & \left|\mu_c(h)\right| < \alpha \quad \forall c \in \mathfrak{C}_{x_i}. \end{aligned}$$

The final result of the analysis is thus having each dynamic variable associated with a time scale. More precisely, if $\alpha$ was correctly chosen (in a sense to be discussed), it is guaranteed that if the integration step is set below a certain $\overline{h}_i$, then the discretized ODE equation that computes $x_{i,k+1}$ cannot be responsible for possible instabilities.

Ranking the dynamic variables by $\overline{h}_i$ will provide the basis for the subsequent decoupled integration. Before that, however, it is convenient to show how the procedure just sketched can be specialized and implemented with an integration method of the considered class. For the sake of simplicity, here we select the explicit Euler one.

### 4.3 A possible analysis implementation

Taking the explicit Euler (EE) as the "probe" integration method—see the remark before (3)—the discretized system of the same equation specializes to

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h \cdot \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k). \tag{9}$$

Thus, the edge weights of the associated digraph take the form

$$\rho\left[e^{i,j}\right](h) = \begin{cases} 1 + h \cdot \dfrac{\partial f_i}{\partial x_i} & \text{if } i = j, \\ h \cdot \dfrac{\partial f_i}{\partial x_j} & \text{if } i \neq j. \end{cases}$$

As a consequence, the cycle gains (6) can be computed as

$$\mu_c(h) = \begin{cases} 1 + h \cdot \dfrac{\partial f_i}{\partial x_i} & \text{if } L = 1 \text{ and } \dfrac{\partial f_i}{\partial x_i} < 0, \\ h^L \displaystyle\prod_{e^{i,j} \in c} \dfrac{\partial f_i}{\partial x_j} & \text{otherwise,} \end{cases} \tag{10}$$

resulting in a set of constraints

$$\left|\mu_c(h)\right| \leq \alpha \quad \Rightarrow \quad \begin{cases} 0 < h \leq (1+\alpha) \left| \dfrac{\partial f_i}{\partial x_i} \right|^{-1} & \text{if } L = 1 \text{ and } \dfrac{\partial f_i}{\partial x_i} < 0, \\ 0 < h \leq \sqrt[L]{\alpha} \cdot \left| \displaystyle\prod_{e^{i,j} \in c} \dfrac{\partial f_i}{\partial x_j} \right|^{-\frac{1}{L}} & \text{otherwise.} \end{cases} \tag{11}$$

As can be noticed, in that very simple case, the set of constraints (11) can be solved analytically in a closed form. This is one of the advantages of adopting EE instead of a more complex integration method for the analysis part.

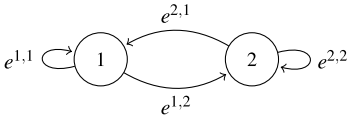


**Fig. 3** Dependency graph associated with system (12) discretized with EE

An additional interesting remark is that, in the nonlinear case, CA produces results that depend on the considered equilibrium. Since using EE as a probe method allows one to express the constraints on $h$ as a function of $\alpha$ in a closed form, this opens the possibility of performing a parametric analysis depending on the equilibrium point of interest, for example, the working point of the system. In many practical cases this will not significantly affect the result of the analysis since it is just a means of associating each variable with a time scale, which usually does not vary abruptly during the simulation. However, this possibility widens the applicability of the proposal.

## 4.4 Cycle analysis and eigenvalue analysis

In the literature, there are two major techniques to serve an analogous purpose, concerning time scale analysis, as that of this paper: eigenvalue [22] and Lyapunov exponent analysis [14, 27]. This section compares our technique to eigenvalue analysis, spending also some words on the Lyapunov exponent subject, as for the problem of guaranteeing the stability of the discrete-time solution. Doing so, we also provide the background for the subsequent discussion of Sect. 5 on how the analyzed techniques can lead to a suitable partition of the system, in a view to a decoupled solution. To this end, we first go through a representative example and then draw the necessary general conclusions.

### *4.4.1 An example: loosely damped models and stability issues*

Consider the linear time-invariant autonomous system

$$\dot{\mathbf{x}} = \begin{bmatrix} -\omega_n \xi & -\omega_n \sqrt{1-\xi^2} \\ \omega_n \sqrt{1-\xi^2} & -\omega_n \xi \end{bmatrix} \mathbf{x}, \tag{12}$$

which has the two complex conjugate eigenvalues

$$\lambda_{1,2} = -\omega_n \cdot \left(\xi \pm \imath \sqrt{1-\xi^2}\right)$$

with natural frequency $\omega_n > 0$ and damping factor $0 < \xi \leq 1$, thus being asymptotically stable. If (12) is discretized with the EE method, the eigenvalues of the corresponding discrete-time system provide the stability condition

$$h < 2\frac{\xi}{\omega_n} := \overline{h}_s. \tag{13}$$

Applying CA, the digraph of Fig. 3 is readily built, and the cycle gains turn out to be

$$\mu_{\langle e^{1,1} \rangle} = \rho\left[e^{1,1}\right] = 1 + h\frac{\partial f_1}{\partial x_1} = 1 - h\omega_n \xi,$$

$$\mu_{\langle e^{2,2} \rangle} = \rho\left[e^{2,2}\right] = 1 + h\frac{\partial f_2}{\partial x_2} = 1 - h\omega_n \xi,$$

$$\mu_{\langle e^{1,2},e^{2,1}\rangle} = \rho\left[e^{1,2}\right]\cdot\rho\left[e^{2,1}\right] = h^2\cdot\frac{\partial f_1}{\partial x_2}\cdot\frac{\partial f_2}{\partial x_1} = -h^2\omega_n^2\left(1-\xi^2\right),$$

leading to the $\alpha$-dependent constraints

$$\begin{aligned}
|\mu_{\langle e^{1,1}\rangle}| < \alpha \quad &\Rightarrow \quad h \le \frac{1+\alpha}{\omega_n\xi} := \overline{h}_{c,1},\\
|\mu_{\langle e^{2,2}\rangle}| < \alpha \quad &\Rightarrow \quad h \le \frac{1+\alpha}{\omega_n\xi} := \overline{h}_{c,2},\\
|\mu_{\langle e^{1,2},e^{2,1}\rangle}| < \alpha \quad &\Rightarrow \quad h \le \frac{1}{\omega_n}\cdot\sqrt{\frac{\alpha}{1-\xi^2}} := \overline{h}_{c,3}.
\end{aligned} \tag{14}$$

It is then interesting to compare the stability bounds on $h$ provided by eigenvalue analysis and those that limit the magnitude of the cycle gains provided by CA. In particular, the CA bounds on $h$ are looser than the eigenvalue-related bounds (thus, CA does not guarantee discrete-time stability) if $\overline{h}_s \le \overline{h}_{c,i}$, that is,

$$\begin{cases} 2\dfrac{\xi}{\omega_n} \le \dfrac{1+\alpha}{\omega_n\xi}, \\ 2\dfrac{\xi}{\omega_n} \le \dfrac{1}{\omega_n}\cdot\sqrt{\dfrac{\alpha}{1-\xi^2}}. \end{cases} \quad \Rightarrow \quad \begin{cases} \alpha \ge 2\xi^2 - 1, \\ \alpha \ge 4\xi^2(1-\xi^2). \end{cases} \tag{15}$$

### *4.4.2 Discussion*

In the example—but this is intuitively general—a value of $\alpha$ can be found so that the CA constraints also guarantee stability, as the eigenvalue ones do. In particular, there exists an $\alpha$ that makes the two upper bounds on $h$ coincident. Below the said value, $\alpha$ however provides to CA an additional degree of freedom with respect to eigenvalue analysis, and this degree of freedom can be exploited to attenuate the effects of mutual dependencies among the discrete-time dynamic variables—a purpose that is apparently decoupling-related and not natural to pursue with the eigenvalue-based approach.

Coming back to the example, we can notice that the value of $\alpha$ that makes the two bounds on $h$ coincide depends only on $\xi$ and not on $\omega_n$. This is more relevant than it may seem at a first glance since if we focus on which of the two constraints in (15) dominates, depending on $\xi$, we notice that for high damping factors, this is the one related to the two cycles with $L = 1$, relating each dynamic variable to itself, whereas for low damping factors, the dominant constraint comes from the cycle with $L = 2$, involving both dynamic variables; this is illustrated in Fig. 4.

In other words, while a reduction of $\xi$, viewed from the eigenvalue standpoint, appears just as a stability degree reduction, the same fact—observed conversely by CA—reveals its nature of a stronger coupling between parts of the system. In this sense, therefore, CA provides stability-related information in a way that is particularly keen to be used for system partitioning in a view to decoupled integration.

To see the same matter from another viewpoint, one can notice that for a (linear) system of order $n$, eigenvalue analysis provides $n$ constraints on $h$, one per each of the system modes, whereas CA provides *at least n* constraints, one constraint per system cycle. In other words, with eigenvalue analysis one observes the system mode by mode, implicitly considering a state space where all those modes are decoupled (the examples showed this only for a couple of complex modes, but the generalization is straightforward). With CA, on the

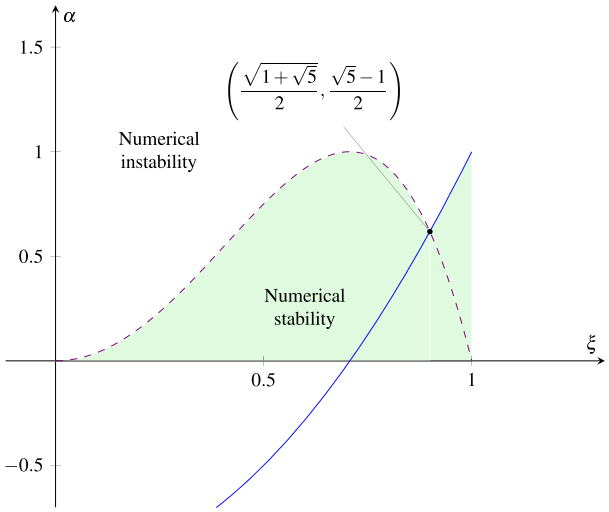


**Fig. 4** Stability conditions on the parameter $\alpha$ w.r.t. $\xi$

contrary, the same information is split to explicitly evidence the couplings that eigenvalue analysis conceals in the above sense.

Incidentally, in the linear case, CA provides exactly $n$ constraints in the case of a triangular system with real eigenvalues since in such a case the said eigenvalues appear in the diagonal of the dynamic matrix; in this case, quite obviously, the value of $\alpha$ discriminating stability from instability is the unity.

## 5 Separability indices

The result of CA is associating each dynamic variable with an upper bound of the integration step, thus with a quantity related to its time scale. The variables can then be ordered, and possibly clustered, by increasing value of $\overline{h}_{x_i}$. Based on this, some synthetic indices will now be defined, useful for deciding how to partition the original model in weakly coupled submodels. It will also be shown how such indices extend the idea of "stiffness," like CA was shown to evidence more decoupling-related information than eigenvalue analysis.

To start, consider the classical stiffness indicator based on eigenvalues analysis, that is, the *stiffness ratio*.

**Definition 5** (Stiffness ratio) The *stiffness ratio* $\sigma_R$ [6] is defined as the ratio between the absolute largest real part and the absolute smallest real part of any eigenvalue, that is,

$$\sigma_R = \frac{\max_i |\Re\{\lambda_i\}|}{\min_i |\Re\{\lambda_i\}|}.$$

Highly stiff systems are associated with high values of $\sigma_R$. The definition of $\sigma_R$, however, cannot be applied to any stiff system, excluding, for example, those of order 1. In addition, for systems with eigenvalues on or very close to the imaginary axis, $\sigma_R$ can be misleading since it just considers the real part of the eigenvalues and may flag as nonstiff a system with a highly oscillatory behavior.

Apparently, the stiffness ratio is defined for a linear (or linearized) system and indicates how much the smaller time scale differs from the larger one. It is thus a good index for

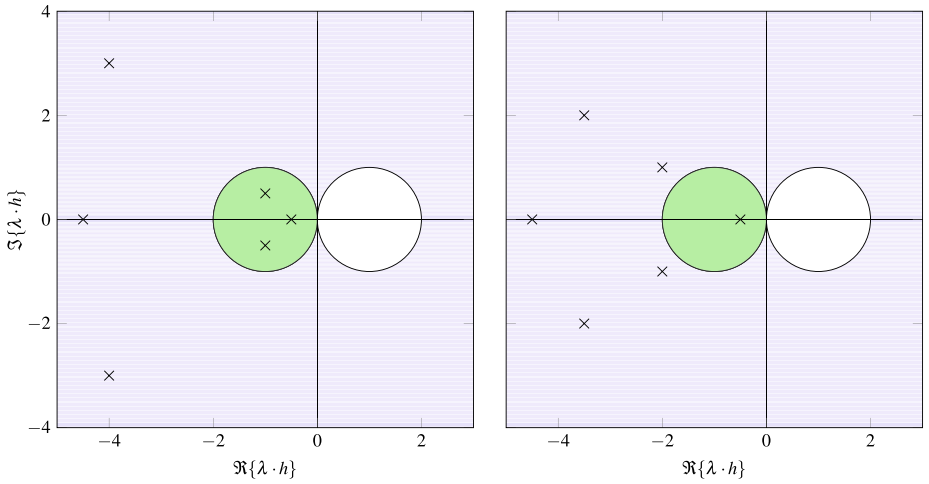


**Fig. 5** Two cases of linear systems with the same stiff ratio

understanding whether or not to use an integration method for stiff systems on the entire model but gives no information on how many "clusters of time scales" are present in it, nor on which dynamic variable belongs to which cluster.

To exemplify, let us limit to the linear case and consider Fig. 5. In the left graph, the continuous-time eigenvalues of the system (indicated with the cross) are not equally spaced in the left-half-plane and can be divided into two clusters: those that are close to the origin are associated with "slow dynamics," whereas the others are associated with "fast dynamics." The presence of the two different time scales is also evidenced by computing the stiffness ratio of Definition 5. Let us now consider the right graph of the same figure. In this case, the stiffness ratio is the same since the closest and the farthest eigenvalues from the origin are the same, while the eigenvalues of the system are almost equally distributed in the left half-plane. This feature of the system is strictly related to how much the system can be "separable" and is not evidenced in any way by the stiffness ratio.

Coming back to the CA approach, two different indices based on it can be defined. One (the *stiffness index*, see Definition 6) quantifies the span of the time scales in the model, analogously to the one of Definition 5. The other (the *separability index*, see Definition 8) indicates to what extent the clusters of dynamic variables corresponding to those time scales of the system can be computed in a decoupled manner. Both indices are function of $\alpha$, and being based on CA, they can be computed also for nonlinear systems.

Denote by $\mathscr{H}$ the set of integration steps $h_{x_i}$ associated with each dynamic variable, and assume that $\mathscr{H}$ ordered by ascending values of $h$, that is,

$$\mathscr{H} = \{h_1 \leq h_2 \leq \cdots \leq h_N\}.$$

Based on that, the following definitions can be given.

**Definition 6** (Stiffness index) The *stiffness index* for a given $\alpha$ is the ratio between the minimal and maximal integration steps found with the CA, that is,

$$\sigma(\alpha) = \frac{h_{\max}(\alpha)}{h_{\min}(\alpha)}. \tag{16}$$

Analogously to the stiffness ratio $\sigma_R$, also for the stiffness index, highly stiff systems are associated with high values of $\sigma$.

**Definition 7** (Separability term) The *separability term* for a given $\alpha$ and for a given couple of variables $x_i$ and $x_j$ is

$$s_\alpha(i,j) = \frac{|h_i(\alpha) - h_j(\alpha)|}{\max_m(h_{m+1}(\alpha) - h_m(\alpha))}, \quad h_i, h_j \in \mathscr{H}.$$

**Definition 8** (Separability index) The *separability index* for a given $\alpha$ is one minus the ratio between the maximal and average differences between two subsequent values of the time scales, that is,

$$s(\alpha) = 1 - \frac{\frac{1}{N-1}\sum_{i=1}^{N-1} h_{i+1}(\alpha) - h_i(\alpha)}{\max_i(h_{i+1}(\alpha) - h_i(\alpha))} = 1 - \frac{1}{N-1}\sum_{i=1}^{N-1} s_\alpha(i+1,i).$$

Apparently, high values of $s(\alpha) \in (0,1)$ indicate that the time scales involved in the system are different enough to be effectively separated.

In Sect. 7, the presented indices will be used to evaluate the level of stiffness and separability of the considered examples. Summarizing, the stiffness ratio and index are comparable, as well as synthetic descriptions of the separation between the maximum and minimum model time scales, not suited however for understanding whether the said model can be partitioned. The separability index is another synthetic one, but it is specifically targeted at quantifying the possibility of such a separation. The separability term is a local index to a couple of adjacent time scales, and an analysis of its behavior can easily suggest possible separation points. The separability term will be used in the following to perform a parametric analysis with respect to $\alpha$, allowing us to identify the aforementioned separation points.

## 6 Decoupled integration

Referring to the partition of DD into structural (cycle) analysis and decoupled integration, one can notice that even the adoption of the sole first part yields performance improvements independently of the adopted integration scheme. In this section we concentrate on the second part, describing a possible decoupled integration method that exploits the partition coming from CA. More precisely, we consider here the use of a mixed-mode integration method, but alternative ones may be considered, for example, cosimulation architectures.

The underlying idea is that implicit methods are able to simulate stiff systems with larger integration periods at the cost of solving a nonlinear set of algebraic equations at each step, whereas explicit methods are better in terms of performance, but cannot deal with stiff systems equally well. Having separated the system in (at least) two parts with different time scales, it is possible to use an implicit method for the fast part(s) and an explicit one for the slow part(s), exploiting the advantages of both kinds of integration algorithms.

If this approach is taken, using for the mixed-mode integration the implicit Euler (IE) and the EE methods, the discrete-time system associated with the continuous-time one reads

$$\begin{cases} \mathbf{x}^s_{k+1} = \mathbf{x}^s_k + h \cdot \mathbf{f}\big(\mathbf{x}^s_k, \mathbf{x}^f_k, \mathbf{u}_k\big), \\ \mathbf{x}^f_{k+1} = \mathbf{x}^f_k + h \cdot \mathbf{f}\big(\mathbf{x}^s_{k+1}, \mathbf{x}^f_{k+1}, \mathbf{u}_{k+1}\big). \end{cases} \tag{17}$$

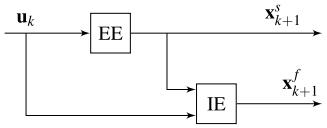


**Fig. 6** Explicit/Implicit Euler integration scheme

As can be seen, in (17) the fast component $\mathbf{x}^f_{k+1}$ can be computed considering $\mathbf{x}^s_{k+1}$ as an input. Figure 6 shows the resulting mixed-mode integration scheme.

Of course, the achieved efficiency improvement depends on the chosen methods. However, the complexity of implicit methods is typically $\mathcal{O}(n^3)$, where $n$ is the dimension of the model, whereas that of explicit ones is typically $\mathcal{O}(1)$. Thus, even having part of the model integrated with an explicit method, the step size automatically tailored to the required precision easily results in a relevant increase of the simulation speed.

## 7 Examples

### 7.1 Double-mass, triple spring-damper

This example refers to a simple test problem, similar to that presented in [10]. The considered system is composed of two masses and three parallel spring-damper elements, connected as shown in Fig. 7 and moving in a horizontal plane (i.e., gravity has no effect). The model includes also the vertical displacement dynamics of the two masses $y_1$ and $y_2$, which are included in the analysis, even if in the addressed scenario is not considering gravity. Both elasticity and damping friction are assumed to be linear phenomena, so that the couplings between the dynamic variables can be easily determined by acting on the elastic constants $k_i$ and the damping factors $d_i$. In particular, in the reported test, $M_1 = M_2 = 1$ kg, $k_1 = 500$ N/m, $d_1 = 5$ N s/m, $k_2 = 1$ N/m, $d_2 = 1$ N s/m, $k_3 = 5$ N/m, and $d_3 = 1$ N s/m.

Letting $x_1$ and $x_2$ be the horizontal positions of the two masses represented in Fig. 7, the model can be written as

$$\begin{cases} M_1\ddot{x}_1 = -(d_1 + d_2)\dot{x}_1 + d_2\dot{x}_2 - (k_1 + k_2)x_1 + k_2x_2, \\ M_2\ddot{x}_2 = d_2\dot{x}_1 - (d_2 + d_3)\dot{x}_2 + k_2x_1 - (k_2 + k_3)x_2. \end{cases} \tag{18}$$

A preliminary analysis is needed so as to understand if the model with the given set of parameters is suited to be partitioned. This is carried out by means of a parametric CA, where EE is chosen as the probe integration method, that is, by exploiting the closed-form solution (11) and by computing the separability terms of Definition 7. The result is shown in Fig. 8, where the numbers on the vertical axis index the variables ordered by increasing time scale.

It is apparent from the figure that the highest separability term is obtained between the 4th and 5th variables, suggesting where to partition the model for the decoupled integration.

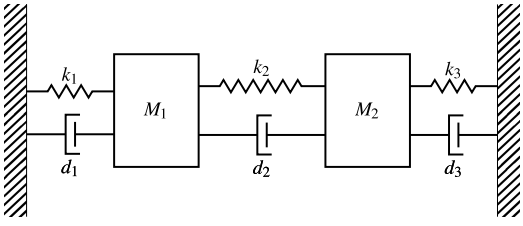


**Fig. 7** Double-mass, triple spring-damper

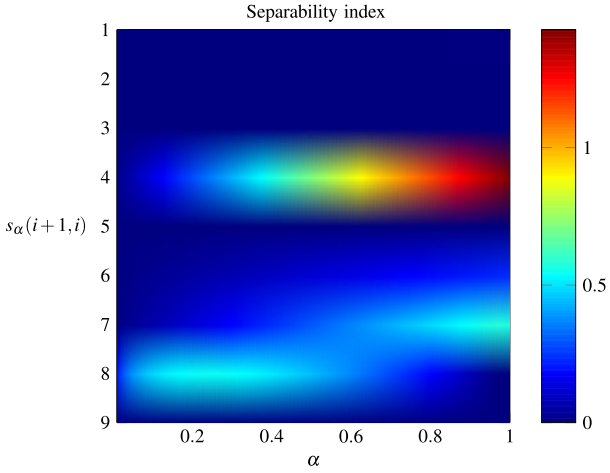


**Fig. 8** Separability parametric analysis of the double-mass, triple spring-damper system

According to CA, there are 17 cycles in the model digraph, and choosing $\alpha = 0.5$, the following constraints on the integration step are obtained:

$$\begin{aligned} \dot{x}_1 &: \quad h \leq 0.0315912, \qquad \dot{x}_2 : \quad h \leq 0.288675, \\ x_1 &: \quad h \leq 0.0315912, \qquad x_2 : \quad h \leq 0.288675, \\ \dot{y}_1 &: \quad h \leq 0.0446767, \qquad \dot{y}_2 : \quad h \leq 0.408248, \\ y_1 &: \quad h \leq 0.0446767, \qquad y_2 : \quad h \leq 0.408248. \end{aligned} \tag{19}$$

Based on the aforementioned analysis, we can partition the model into two submodels by separating the first four variables, the set of the ones on the left, which are considered fast, from the other four, the set of the ones on the right, which are considered slow. Hence, the integration step can be chosen as $h = 0.05$. Notice that incidentally the analysis brought to an indeed intuitive result, that is, separation of the two sets of equations associated to the two masses without any a priori suggestion to the method of the physical structure of the system.

Figure 9 shows the numerical results of the mixed-mode integration method, whereas Table 1 presents some comparative simulation statistics. Vertical displacements are not reported here since they are all zero since gravity is not considered.

To evaluate the effectiveness of the approach, we compared mixed-mode simulation results to a solution taken as reference, which, in the absence of an analytical one, is that coming from a highly accurate, variable-step method with tight enough tolerances, that is, in this case LSODAR, with relative and absolute tolerances set to $10^{-6}$.

The mixed-mode integration method is able to capture the system behavior, especially for the steady-state, whereas the choice of a "large" integration step has the effect that the "fast" dynamics, that is, the transient oscillations, are approximated by a slower dynamics. Furthermore, simulation statistics show that also in this first (linear) example, performance is slightly improved with respect to other first-order methods, that is, explicit and implicit Euler. In addition, the accuracy is really close to that obtained with a pure IE.

To complete the example, the proposed indices proposed in Sect. 5 are here computed. In particular, computing the time scales (19) of model (18) by means of CA yield the following indices (notice that $\sigma_R$ cannot be computed since there are purely imaginary eigenvalues in

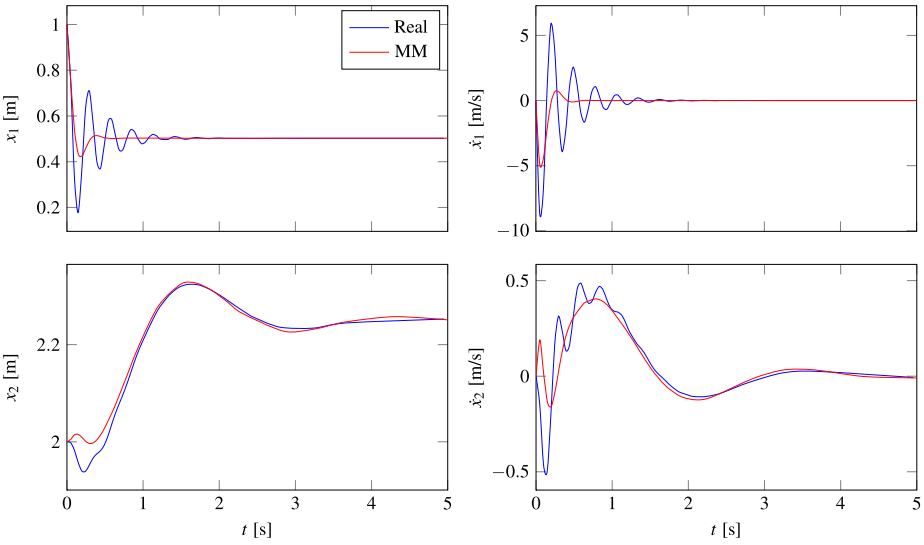


**Fig. 9** Simulation results of the double-mass, triple spring-damper system

**Table 1** Simulation statistics double-mass, triple spring-damper

| | Mixed-mode | LSODAR | IE | EE |
|---|---|---|---|---|
| # Steps | 101 | 483 | 101 | 500 |
| # Function ev. | 302 | 949 | 302 | – |
| # Jacobian ev. | 5 | 28 | 5 | – |
| # Fun. ev. in Jac. ev. | 25 | – | 45 | – |
| # Newton iterations | 201 | – | 201 | – |
| # Newton fail | 0 | – | 0 | – |
| Accuracy | 5.969 | – | 5.964 | 14.472 |
| Sim time | 0.10 s | 0.12 s | 0.11 s | 0.13 s |

the system):

$$\sigma(0.5) = 12.923, \qquad s(0.5) = 0.779.$$

The stiffness index $\sigma(\alpha)$ indicates that the system is highly stiff. On the other hand, the separability index $s(\alpha)$ shows that the considered example has dynamics evolving with quite different time scales, thus making it effective to partition the model into subsystems. Finally, to decide how to obtain that partition, adequate clues are provided by Fig. 8.

### 7.2 Mechanical system with brake

In this example, the system of Fig. 10 is considered. A body of mass $M$ moves on a horizontal guide subject to an exogenous motor torque command $\tau^o(t) = 10\sin(2\pi t/5)$ and to friction acting on the wheels. The motor is not modeled for simplicity, and the relationship between the torque command and the actual torque $\tau(t)$ is simply represented by a unity-gain, first-order continuous-time system. Also, the motor-wheel system compliance is lumped in a single rotational elasticity, $\delta\varphi$ indicating the angle difference between its sides.

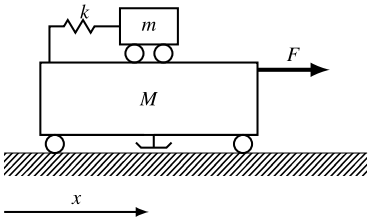


**Fig. 10** Mechanical system with brake

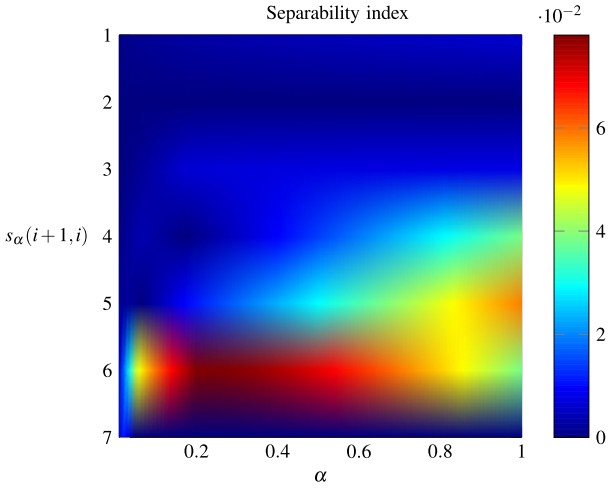


**Fig. 11** Separability analysis of the mechanical system with brake

Another body of mass $m$ is connected to the first one by a spring and is also subject to friction with the former. The system also contains a brake, mounted on mass $M$ and acting on the guide, thus introducing an *input-by-state nonlinearity*.

In the following, $x_M$ denotes the position of mass $M$, $x_m$ that of mass $m$, $\varphi$ the angle of the wheels, and $\omega$ their angular velocity.

Similarly to what have been done for the previous example, a preliminary analysis keeping $\alpha$ as a parameter is needed to understand if the system at hand exhibits quite different time scales. The result of this parametric analysis is presented in Fig. 11.

In this case, CA detects 19 cycles. The highest separability term depends on the choice of $\alpha$ since for values close to 1, the separability terms assume higher values for $i = 5$, that is, partitioning the system between the 5th and the 6th variables, whereas for lower values of $\alpha$, the highest separability term is obtained for $i = 6$. Assuming that it is preferable to keep the fast subsystem as small as possible, in this case we proceed with $\alpha = 1.0$:

$$\begin{aligned}
\dot{x}_M: &\quad h \leq 9.9988 \times 10^{-5}, & \delta\varphi: &\quad h \leq 0.111111,\\
x_m: &\quad h \leq 0.00632456, & \varphi: &\quad h \leq 0.149535,\\
\dot{x}_m: &\quad h \leq 0.00632456, & \omega: &\quad h \leq 0.149535,\\
x_M: &\quad h \leq 0.0141421,\\
\tau_t: &\quad h \leq 0.0526316.
\end{aligned}$$

Partitioning the system as suggested by the parametric analysis means, for example, choosing an integration step $h = 0.06$ for the mixed-mode integration method and obtaining

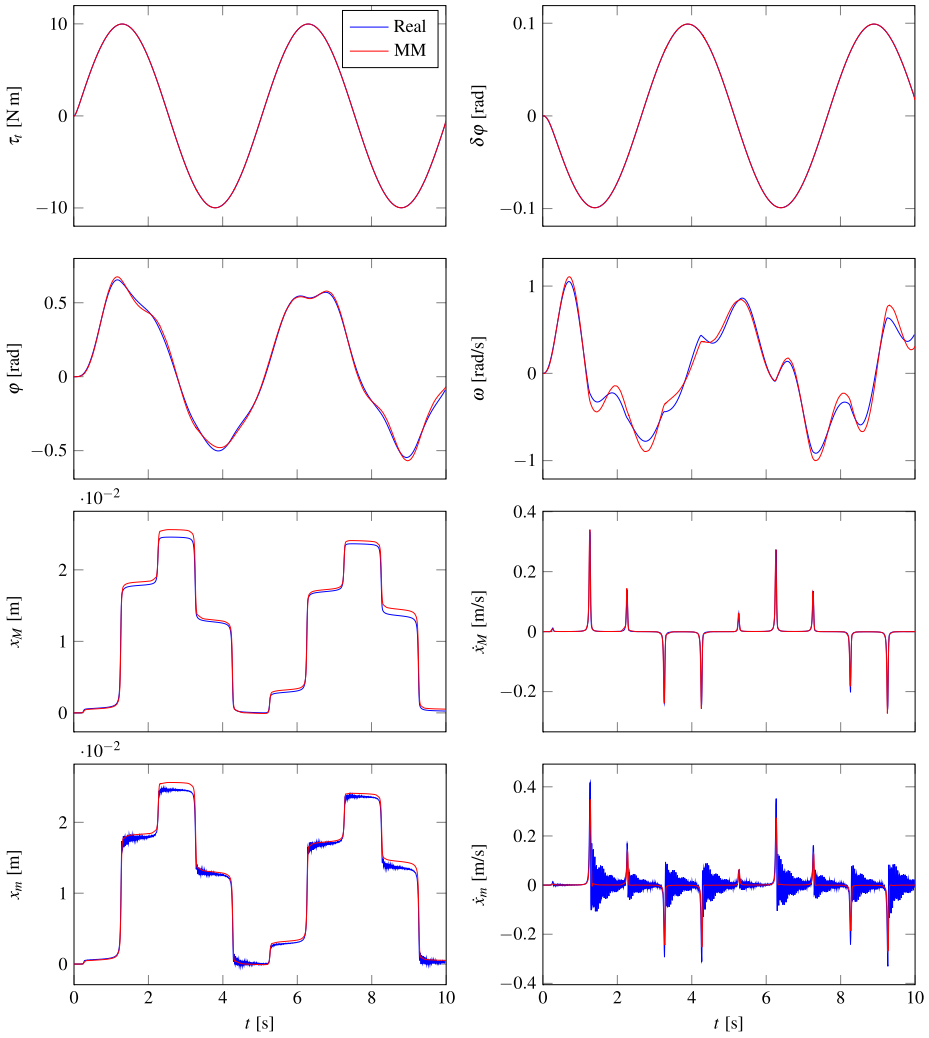


**Fig. 12** Simulation results of the mechanical system with brake

as fast variables the set of the ones on the left and as slow variables the set of the ones on the right.

Figure 12 shows the numerical results of the mixed-mode integration method, whereas Table 2 presents some comparative simulation statistics.

As in the previous case, the LSODAR solution is taken as a reference to compare the obtained results. Also in this case, the mixed-mode integration method is able to capture the main dynamics in accordance with the chosen separation time scale. Furthermore, simulation statistics show that also in this example, which has an input-by-state nonlinearity, there is an improvement in terms of performance with respect to other methods, especially for LSODAR, which is a variable-step one.

To complete the example, the proposed indices proposed in Sect. 5 are here computed, yielding the following indices (notice that due to the nonlinearity of the system, $\sigma_R$ cannot

**Table 2** Simulation statistics for the mechanical system with brake

| | Mixed-mode | LSODAR | IE | EE |
|---|---|---|---|---|
| # Steps | 168 | 8363 | 168 | $10^5$ |
| # Function ev. | 1296 | 18816 | 1293 | – |
| # Jacobian ev. | 103 | 928 | 103 | – |
| # Fun. ev. in Jac. ev. | 618 | – | 927 | – |
| # Newton iterations | 1128 | – | 1125 | – |
| # Newton fail | 102 | – | 102 | – |
| Accuracy | 9.962 | – | 1.480 | 10.567 |
| Sim time | 0.45 s | 2.17 s | 0.50 s | 23.9 s |

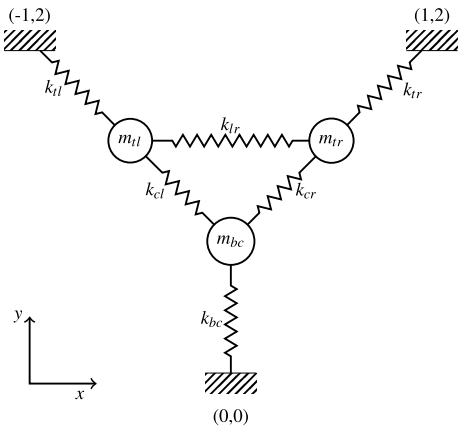


**Fig. 13** The "triangle of masses" system (dampers are not represented to simplify the drawing)

be computed):

$$\sigma(1.0) = 1495.528, \qquad s(1.0) = 0.635.$$

The stiffness $\sigma(\alpha)$ index shows that the system is highly stiff, whereas the separability one shows that it is also well suited to be partitioned.

### 7.3 Triangle of masses

The considered system is composed of three masses, moving in a vertical plane subject to gravity and to the action of six spring-damper elements, as shown in Fig. 13 (a two-dimensional model was created for simplicity). Notice that this model is strongly nonlinear due to the spring equations in two dimensions, and thus the eigenvalue analysis is not applicable.

In the following, $x_i$ and $y_i$ represent the horizontal and vertical displacements of the three masses, whereas the indices $b$, $c$, $l$, $r$, and $t$ indicate respectively bottom, center, left, right, and top. The spring elasticity coefficients $k_i$ and the damping factors $d_i$ are reported in Table 3, whereas the three masses are $m_{bc} = m_{tl} = m_{tr} = 1$ kg.

Also in this case, a parametric CA is performed so as to analyze the structure of the system, and Fig. 14 shows its result.

In the "triangle of masses" system, the computed separability terms evidence that there is a neat separation between the 8th and 9th variables, suggesting this as a good point for the partition.

**Table 3** The "triangle of masses" system parameters

| Parameter | Value | Parameter | Value |
|---|---|---|---|
| $k_{tl}$ | 1 N/m | $d_{tl}$ | 1 N s/m |
| $k_{tr}$ | 1 N/m | $d_{tr}$ | 1 N s/m |
| $k_{bc}$ | 1 N/m | $d_{bc}$ | 1 N s/m |
| $k_{lr}$ | 1 N/m | $d_{lr}$ | 2 N s/m |
| $k_{cl}$ | 10 N/m | $d_{cl}$ | 1 N s/m |
| $k_{cr}$ | 1 N/m | $d_{cr}$ | 1 N s/m |

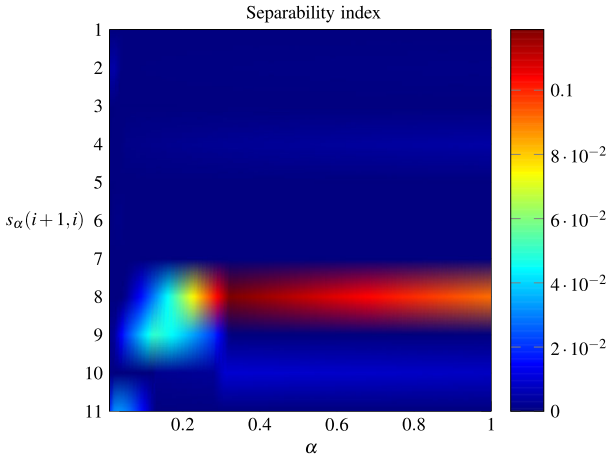


**Fig. 14** Separability analysis of the triangle of masses system

The CA detected 3984 cycles, evidencing how significant the impact of the system's degrees of freedom can be on the number of cycles. Anyway, all cycles are found in less than 1 s. The choice of $\alpha$ can be made on the basis of Fig. 14, trying to maximize the separability term, for example, by choosing $\alpha = 0.5$. The resulting constraints are thus

$$\begin{aligned}
\dot{y}_{bc}: &\quad h \leq 0.0832703, & \dot{x}_{tl}: &\quad h \leq 0.207409,\\
y_{bc}: &\quad h \leq 0.0832703, & \dot{y}_{tl}: &\quad h \leq 0.207409,\\
\dot{x}_{bc}: &\quad h \leq 0.0838537, & x_{tl}: &\quad h \leq 0.207409,\\
x_{bc}: &\quad h \leq 0.0838537, & y_{tl}: &\quad h \leq 0.207409,\\
\dot{x}_{tr}: &\quad h \leq 0.0868503, & &\\
x_{tr}: &\quad h \leq 0.0868503, & &\\
\dot{y}_{tr}: &\quad h \leq 0.0870024, & &\\
y_{tr}: &\quad h \leq 0.0870024. & &
\end{aligned}$$

As in the other examples, the variables on the left are the fast ones, and those on the right the slow ones. The choice of $h = 0.1$ partitions the model in those two subsystems for the mixed-mode integration, obtaining the numerical simulation represented in Fig. 15 (simulation statistics are reported in Table 4). Notice that for EE, a smaller integration step ($h = 0.01$) was used for numerical stability reasons.

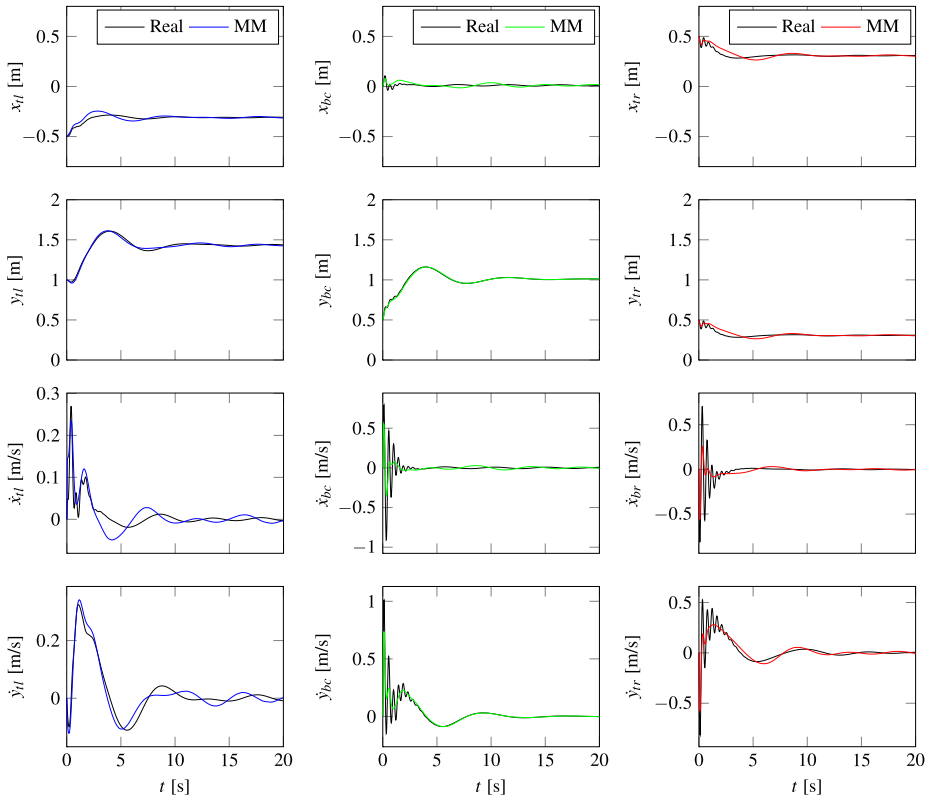


**Fig. 15** Simulation results for the triangle of masses system

**Table 4** Simulation statistics for the masses triangle

| | Mixed-mode | LSODAR | IE | EE |
|---|---|---|---|---|
| # Steps | 200 | 596 | 200 | 2000 |
| # Function ev. | 1236 | 1232 | 857 | – |
| # Jacobian ev. | 13 | 38 | 13 | – |
| # Fun. ev. in Jac. ev. | 117 | – | 169 | – |
| # Newton iterations | 1036 | – | 657 | – |
| # Newton fail | 5 | – | 4 | – |
| Accuracy | 0.871 | – | 0.780 | 1.337 |
| Sim time | 0.38 s | 0.51 s | 0.4 s | 0.48 s |

LSODAR is also taken as the reference solution, and the mixed-mode integration presents an improvement in terms of simulation speed, whereas the accuracy is close to the one of IE.

To complete the example, the indices proposed in Sect. 5 are here computed, yielding the following results (notice that also in this case, due to the nonlinearity of the system, $\sigma_R$ cannot be computed):

$$\sigma(0.5) = 2.491, \qquad s(0.5) = 0.899.$$

The stiffness $\sigma(\alpha)$ index shows that the considered system is sufficiently stiff, but nonetheless the separability one shows that it is suited for the partition since its time scales are well separated.

### 7.4 Discussion

After showing the examples, their collective outcome could be summarized as follows. First, when there is an evident dynamic separation in the system, the proposed technique finds it without requiring a priori information on the part of the user. In other words, the technique is backed up by observing that the produced results are in accordance with intuition when intuition can figure them out.

Also, and in some sense a complement, the proposed indices allow one to synthetically appreciate the possible internal model couplings that can be exploited via DD, even when these are not apparent at all.

Moreover, and specific to the use of the technique for mixed-mode integration, its characteristics are very suited to the typical studies that are required to really have simulation follow the life-cycle of the project, as envisaged in the introduction. On this final point, however, some more words are in order.

When a simulation study is required to answer a specific question, most frequently the focus is on part of the system or, somehow equivalently, on part of the phenomenon occurring in it. In such a very frequent case, the rest of the system does not need to be simulated accurately, provided that the boundary conditions presented to the part that is relevant for the study allow for a precise evaluation of the investigated quantities. In many situations of the type just mentioned, the interest of the analyst is on certain time scales on the system phenomena, and provided that these are well reproduced, loosing faster behaviors is not only acceptable, but in fact necessary to achieve the desired performance. In fact, the same remark holds also for almost the totality of MOR techniques, where low-frequency approximations of the original model are the typical result, and the quality of a reduction is not evaluated in terms of time error, which is typically large due to the transients, but rather in terms of the $H_\infty$-norm of the difference between the original and reduced models, that is, in the frequency domain. This is totally analogous to the proposed approach, where a good approximation is not strictly related to a small simulation error, but to a good representation of the time scales of interest.

## 8 Application-oriented remarks

After presenting the proposed DD-based technique in its entirety, a few words are in order to motivate some of the adopted choices and discuss its practical use.

Starting from CA, its application requires to select the "probe" discretization method. The choice made in this work is the EE one, and some motivation for that is in order.

In fact, after describing the CA technique, we could observe that the ranking of the dynamic variables by time scale was obtained by exploiting a known weakness of explicit fixed-step integration (probe) methods, that is, their fairly small region of numerical stability; see, for example, Fig. 16 for the explicit Runge–Kutta (ERK) family [6].

The natural method selection guidelines are therefore the conservatism of the obtained ranking and the ease in writing and solving the constraint inequalities on $h$. By the way, the second guideline is the major reason why explicit methods are considered since in the opposite case it would be necessary to solve those inequalities numerically.

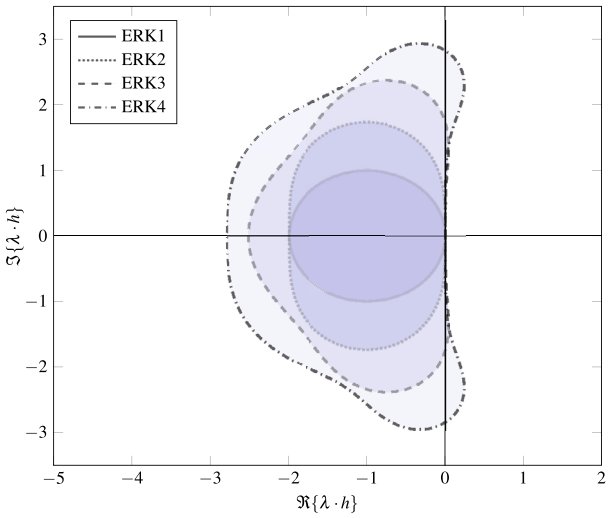


**Fig. 16** Numerical stability domains of ERK (interior of the curves)

Having so motivated the choice for explicit methods and given that CA was shown to be applicable to any of them, the method selection problem is reduced to its core. Among all the methods that are still candidates at this point, the EE one has the advantage of always permitting an analytical closed-form solution of the constraint inequalities while exhibiting a stability region small enough to provide a conservative ranking (see again Fig. 16, recalling that EE coincides with ERK1). From a practical standpoint, the authors cannot see any reason for the use of different methods, except possibly for those of Adams–Bashforth type in the case of extremely loosely damped dynamics, and in general the EE method performed satisfactorily in all the numerous applications considered so far.

A possible issue with CA is the computational complexity of the method. Unfortunately, the problem of finding all the cycles in a digraph has complexity $\mathcal{O}(2^{|E|-|N|+1})$ and is a well-known and studied problem in the operations research community [9, 13, 25, 26]. This is of course a limitation with strongly connected digraphs, which are however seldom encountered when modeling physical systems, especially in the multibody case. Just to give an idea, a python implementation of CA takes no longer than 1 s to perform the overall search and analysis in all the considered examples.

Apart from the last remark, in the first place, CA is an offline activity with respect to simulations and needs to be performed only once for a given model. Then, optimizations are possible for the search procedure so as to make the required computation time well acceptable, achieving a detection rate of thousands of cycles per second (see the remark in the example of Sect. 7.3). Describing the software implementation of CA is not within the scope of this paper; it is however worth mentioning that the used one is still a proof-of-concept prototype. See [18] for some ideas and ongoing research on CA performance improvement and software details.

To conclude this point, it is worth evidencing that the possibly incurred computational complexity is paid back, as anticipated, in terms of the information coming from CA. In particular, CA dictates not only the time scales associated with each state variable, but also the variables that are mutually interacting. This information can be used to identify independent components in the model—the strongly connected components of the dependency graph—to make the simulation code parallel, possibly combining this work with [5] (a matter deferred to future research).

Another point to discuss is the choice of $\alpha$, which is the only design parameter of the method and controls the tradeoff between the accuracy of the resulting simulation, and the achieved degree of decoupling. Specifically, lower values of $\alpha$ result in a higher simulation accuracy, but also in a reduced capability to detect weakly coupled components.

At this stage of the research, in the choice of $\alpha$, some heuristics is still required. According to experience, we could say that a reasonable default choice for $\alpha$ is the unity in the presence of systems exhibiting only overdamped dynamics, whereas things can be more critical, requiring lower values in the presence of loosely damped behaviors. Further investigation of this matter is devoted to future works, but it can already be stated that suitable guidelines for the choice of $\alpha$, possibly problem-specific as just suggested, can be devised quite easily. It is also worth noticing that the computationally intensive part of the method is the analysis of the model digraph, which does not depend on $\alpha$: if needed, performing multiple analysis runs with different values for that parameter, until a reasonable accuracy/separability compromise is found, is therefore an affordable task. Even more specifically, if EE is chosen as the probe discretization method, stiffness index, separability terms, and index can be computed as functions of $\alpha$, allowing for a parametric analysis as performed in the previous section.

On the same front, we could thus better qualify the statement made in Sect. 2 that the presented technique "is not scenario-based." In fact the *result* of the technique, that is, the model partition, does depend on the considered operating point, but (again, if a convenient probe method like EE is used) this dependence just means that the ranking of the dynamic variables may need to be recomputed, whereas the analysis is done only once. This is not true for other scenario-based techniques (see, e.g., [17]), where the entire procedure has to be repeated.

As a final remark, although the matter rigorously strays from the scope of this paper, the very relevant problem of model initialization in a cosimulation context [2] is tendentiously easier to handle if one *first* obtains and initializes a monolithic model and *then* partitions it. The advantages of the presented technique in this respect should be quite evident.

## 9 Conclusion and future work

A technique for the automatic application of dynamic decoupling was presented, which is naturally applicable to nonlinear models and presents to the analyst information in the form of time scales corresponding to the dynamic variables, which is easy enough to interpret and handle also for nonspecialists.

With respect to the major available alternatives, the presented technique has more than one advantage. It preserves the state space of the model, can be considered scenario-free, and is applicable both in the case of a monolithic solution and of cosimulation.

Examples were presented and commented to explain the rationale of the technique and also to illustrate its operation in a few representative situations.

Future work deals with further investigation of the physical interpretation of $\alpha$ and on how to formally relate it to the simulation accuracy. More in perspective, one of the long-term goals of this research is integrating the proposed technique—and also alternative ones like MOR, TLM, and so forth—with modern modeling and simulation tools, with a particular attention to Modelica-based ones, by setting up a unifying approximation framework, as sketched in [18], so as to provide to the final user a larger set of functionalities that can be used in an automatic and transparent way.

## References


1. Antoulas, A.: Approximation of Large-Scale Dynamical Systems, vol. 6. SIAM, Philadelphia (2005)
2. Arnold, M., Schiehlen, W.O.: Simulation Techniques for Applied Dynamics, vol. 507. Springer, Berlin (2009)
3. Bartolini, A., Leva, A., Maffezzoni, C.: A process simulation environment based on visual programming and dynamic decoupling. Simulation **71**(3), 183–193 (1998)
4. Blochwitz, T., Otter, M., Åkesson, J., Arnold, M., Clauß, C., Elmqvist, H., Friedrich, M., Junghanns, A., Mauss, J., Neumerkel, D., Olsson, H., Viel, A.: Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: 9th International Modelica Conference, Münich, Germany, pp. 173–184 (2012). doi:10.3384/ecp12076173
5. Casella, F.: A strategy for parallel simulation of declarative object-oriented models of generalized physical networks. In: 5th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, pp. 45–51 (2013)
6. Cellier, F., Kofman, E.: Continuous System Simulation. Springer, Berlin (2006)
7. Chen, J., Kang, S.M.: Model-order reduction of nonlinear MEMS devices through arclength-based Karhunen-Loeve decomposition. In: The 2001 IEEE Int. Symposium on Circuits and Systems. ISCAS 2001, vol. 3, pp. 457–460 (2001). doi:10.1109/ISCAS.2001.921346
8. Chen, J., Kang, S.M., Zou, J., Liu, C., Schutt-Aine, J.: Reduced-order modeling of weakly nonlinear MEMS devices with Taylor-series expansion and Arnoldi approach. J. Microelectromech. Syst. **13**(3), 441–451 (2004). doi:10.1109/JMEMS.2004.828704
9. Goldberg, L., Ann, G.: Efficient Algorithms for Listing Combinatorial Structures, vol. 5. Cambridge University Press, Cambridge (2009)
10. González, F., Naya, M.Á., Luaces, A., González, M.: On the effect of multirate co-simulation techniques in the efficiency and accuracy of multibody system dynamics. Multibody Syst. Dyn. **25**(4), 461–483 (2011). doi:10.1007/s11044-010-9234-7
11. Gu, C.: Model order reduction of nonlinear dynamical systems. Ph.D. thesis, University of California, Berkeley (2011)
12. Innocent, M., Wambacq, P., Donnay, S., Tilmans, H., Sansen, W., De Man, H.: An analytic Volterra-series-based model for a MEMS variable capacitor. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **22**(2), 124–131 (2003). doi:10.1109/TCAD.2002.806603
13. Johnson, D.: Finding all the elementary circuits of a directed graph. SIAM J. Comput. **4**(1), 77–84 (1975)
14. Kuznetsov, Y.: Elements of Applied Bifurcation Theory. Applied Mathematical Sciences, vol. 112. Springer, Berlin (2004)
15. Lall, S., Marsden, J., Glavaski, S.: A subspace approach to balanced truncation for model reduction of nonlinear control systems. Int. J. Robust Nonlinear Control **12**, 519–535 (2002)
16. Mikelsons, L., Brandt, T.: Symbolic model reduction for interval-valued scenarios. In: ASME Conf. Proc., ASME, vol. 49002, pp. 263–272. (2009). doi:10.1115/DETC2009-86954
17. Mikelsons, L., Brandt, T.: Generation of continuously adjustable vehicle models using symbolic reduction methods. Multibody Syst. Dyn. **26**, 153–173 (2011)
18. Papadopoulos, A.V., Leva, A.: Automating dynamic decoupling in object-oriented modelling and simulation tools. In: 5th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, pp. 37–44 (2013)
19. Papadopoulos, A.V., Åkesson, J., Casella, F., Leva, A.: Automatic partitioning and simulation of weakly coupled systems. In: 2013 IEEE 52nd Annual Conference on Decision and Control (CDC), pp. 3172–3177 (2013)
20. Phillips, J.R.: Projection frameworks for model reduction of weakly nonlinear systems. In: Proc. of the 37th Annual Design Automation Conf., DAC'00, pp. 184–189. ACM, New York (2000). doi:10.1145/337292.337380
21. Scherpen, J.: Balancing for nonlinear systems. Syst. Control Lett. **21**(2), 143–153 (1993)
22. Schiela, A., Olsson, H.: Mixed-mode integration for real-time simulation. In: Modelica Workshop 2000 Proceedings, pp. 69–75 (2000)
23. Sjölund, M.: TLM and parallelization. Technical report, Linköping University (2012)
24. Sjölund, M., Braun, R., Fritzson, P., Krus, P.: Towards efficient distributed simulation in Modelica using transmission line modeling. In: 3rd Int. Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, pp. 71–80 (2010)
25. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM J. Comput. **1**(2), 146–160 (1971)
26. Tarjan, R.: Enumeration of the elementary circuits of a directed graph. SIAM J. Comput. **2**(3), 211–216 (1972)
27. Wolf, A., Swift, J.B., Swinney, H.L., Vastano, J.A.: Determining Lyapunov exponents from a time series. Phys. D, Nonlinear Phenom. **16**(3), 285–317 (1985). doi:10.1016/0167-2789(85)90011-9