Numerical Approximation of Port-Hamiltonian Systems for Hyperbolic or Parabolic PDEs with Boundary Control

Andrea Brugnoli, Ghislain Haine, Anass Serhani, Xavier Vasseur

Scientific Research Publishing

Journal of Applied Mathematics and Physics, 2021, 9, 1278-1321
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

Numerical Approximation of Port-Hamiltonian Systems for Hyperbolic or Parabolic PDEs with Boundary Control

Andrea Brugnoli[1], Ghislain Haine[2*], Anass Serhani[3], Xavier Vasseur[2]

[1]Robotics and Mechatronics Department, University of Twente, Enschede, Netherlands
[2]ISAE-SUPAERO, Université de Toulouse, Toulouse, France
[3]CERFACS, Toulouse, France
Email: *ghislain.haine@isae.fr

How to cite this paper: Brugnoli, A., Haine, G., Serhani, A. and Vasseur, X. (2021) Numerical Approximation of Port-Hamiltonian Systems for Hyperbolic or Parabolic PDEs with Boundary Control. *Journal of Applied Mathematics and Physics*, **9**, 1278-1321.
https://doi.org/10.4236/jamp.2021.96088

Received: March 29, 2021
Accepted: June 15, 2021
Published: June 18, 2021

Abstract

We consider the design of structure-preserving discretization methods for the solution of systems of boundary controlled Partial Differential Equations (PDEs) thanks to the port-Hamiltonian formalism. We first provide a novel general structure of infinite-dimensional port-Hamiltonian systems (pHs) for which the Partitioned Finite Element Method (PFEM) straightforwardly applies. The proposed strategy is applied to abstract multidimensional linear hyperbolic and parabolic systems of PDEs. Then we show that instructional model problems based on the wave equation, Mindlin equation and heat equation fit within this unified framework. Secondly, we introduce the ongoing project SCRIMP (Simulation and Control of Interactions in Multi-Physics) developed for the numerical simulation of infinite-dimensional pHs. SCRIMP notably relies on the FEniCS open-source computing platform for the finite element spatial discretization. Finally, we illustrate how to solve the considered model problems within this framework by carefully explaining the methodology. As additional support, companion interactive Jupyter notebooks are available.

Keywords

Port-Hamiltonian Systems, Partial Differential Equations, Boundary Control, Structure-Preserving Discretization, Finite Element Method

1. Introduction

The efficient numerical simulation of complex multiphysics systems is ubiquitous in Computational Science and Engineering. Although a wide range of me-

thods exists to tackle specific problems, they often lack versatility and adaptability, especially when the modelling is of increasing complexity as in real-world applications.

Infinite-dimensional port-Hamiltonian systems (pHs) have been first introduced in [1] using the language of differential geometry. They provide a powerful tool to model complex multiphysics open systems (whether or not being linear) for control purpose. A wide range of physical systems has been written within this formalism, see e.g. [2] [3] [4]. This twenty-year-old framework [5] enjoys nice properties, such as the relevant physical meaning of the variables, a useful underlying linear structure (namely Stokes-Dirac structure) which encodes the power balance satisfied by the Hamiltonian (often chosen as energy), and last but not least: the interconnection of multiple pHs remains a pHs. This allows "modular" modelling of complex multiphysics systems.

Since then, many researchers have developed numerical methods to discretize these systems in a structure-preserving manner, hence keeping the advantages of the infinite-dimensional pHs. Such methods aim at constructing approximate finite-dimensional pHs at the discrete level. Our aim is to show the versatility of PFEM thanks to a new unified framework and to introduce the ongoing project SCRIMP together with companion Jupyter notebooks [6]. Each particular example discussed here has been treated previously [7] [8] [9] [10]. Here, the existence of an underlying common structure for many pHs is highlighted. Obtaining such a general scheme for infinite-dimensional pHs is of major importance for control purposes [11], and for coupling atomic elements into a more complex system with the guarantee of well-preserved energy exchanges between subsystems [12].

The first proposed structure-preserving scheme for pHs dates back to [13], where the authors proposed a mixed finite element spatial discretization for hyperbolic systems of conservation laws. Pseudo-spectral methods relying on higher-order global polynomial approximations were studied in [14]. Unfortunately, this method seems to be limited to the one-dimensional case. A finite difference method with staggered grids was developed in [15] for two-dimensional domains, but complex geometries are then difficult to tackle. Weak formulations leading to Galerkin numerical approximations began to be explored in the past few years. In [16] the prototypical example of hyperbolic systems of two conservation laws has been discretized by a weak formulation. However, the construction of the necessary power-preserving mappings is not straightforward on arbitrary meshes. All these methods require *ad hoc* implementations and are usually restricted to particular cases of pHs. Furthermore, since they do not rely on well-established and versatile numerical libraries, using such techniques remains confined within a small community of experts. We refer the reader for a more complete overview of structure-preserving discretization for pHs to [5] [17] and the references therein.

Thanks to [8] it has become clear that there exists a deep relation between structure-preserving discretization of pHs and Mixed Finite Element Method

(MFEM). Indeed velocity-stress formulations for the wave dynamics [18] and elastodynamics problems are of Hamiltonian type and their mixed discretization preserves this structure for *closed* systems. This leads to the intuition that the MFEM may be used to discretize the underlying geometric structure of pHs in a unified way, even for *open* systems, translating the infinite-dimensional Stokes-Dirac structure into a finite Dirac structure. A first successful application has already been achieved in [19] for discretizing the 1-D Navier-Stokes equations for reactive flows formulated in a port-Hamiltonian formulation. The discretization strategy relies on the partitioned structure of the problem and for this reason, goes under the name of Partitioned Finite Element Method (PFEM). This method proves nice convergence properties, see e.g. [20] for a recent proof on the wave equation, that does not require the fulfilment of the usual *inf-sup* condition for MFEM, generalizing the results cited in ([21], Remark 6).

It has to be pointed out that the core idea of PFEM, *i.e.* performing integration by parts on a *partition* of the weak formulation of the system of equations, has already been proposed for *closed* hyperbolic systems in [21]. Therein, the formulations are called either *primal-dual* or *dual-primal*, depending on the chosen partition of the system.

The major difference between MFEM and PFEM relies on the choice of the test functions in the weak formulation, hence on the finite element form functions. Indeed, in PFEM, they never carry homogeneous boundary conditions. In e.g. ([22], Section 7.1), it is shown that for a Dirichlet control, test functions are taken in the kernel of the Dirichlet trace. As already mentioned, in [21], the proposed *primal-dual* and *dual-primal* discretizations are then suitable for the structure-preserving discretization of *closed* systems. Nevertheless, by keeping these homogeneous conditions in the test functions, not only the application of Dirichlet control is difficult, but the definition of the Neumann observation, necessary for the discrete power balance, would be more complex. PFEM aims at easing the mimicking of the continuous power balance at the discrete level, by relaxing the test functions used in [21]. To the best of our knowledge, this relaxation has not been investigated yet in the case of boundary controlled wave-like systems, and this probably comes from the fact that MFEM has been first developed for elliptic systems. Indeed, in this case, such a boundary condition is mandatory for well-posedness (especially to obtain the *ellipticity in the kernel* condition ([22], Eq. (5.1.7)), while PFEM is made for evolution systems, and more especially for pHs.

In our opinion, the driving forces of PFEM are threefold: first, PFEM takes collocated boundary controls and observations into account in a simple manner; secondly, PFEM is structure-preserving, meaning in particular that the discrete power balance perfectly mimics the continuous one; thirdly, the implementation of PFEM only relies on existing finite element libraries, such as FEniCS [23], selected in the ongoing project SCRIMP for its robustness and efficiency. Last but not least, the pHs point of view allows us to separate axioms of physics (such as

conservation laws) from constitutive laws and equations of state (such as Hooke's law and ideal gas law). PFEM is based on this separation, providing the possibility to tackle parabolic or nonlinear systems, at the price of solving a finite-dimensional *port-Hamiltonian Differential Algebraic Equation* (pHDAE). In the particular case of linear hyperbolic PDE, as shown in Section 2, the constitutive laws can be easily (*i.e.* without matrix inversions) taken into account in order to recover an *Ordinary Differential Equation* (ODE).

PFEM could also be named e.g. *extended MFEM* or *relaxed MFEM*. Since only evolution systems are considered (not necessarily of hyperbolic type, see e.g. Section 3.4), relaxed conditions for the selection of the test functions hold, hence for the finite elements as well. We choose to follow the terminology introduced in [8] and widely used since then. Furthermore, it emphasizes the pH formulation of the initial system to discretize.

Main contributions

We first aim at presenting the strategy of the structure-preserving discretization PFEM, in a new unified abstract framework, allowing for an easy application to a wide class of boundary controlled partial differential equations. Then, in order to show the versatility of our approach, we successively apply PFEM to the boundary controlled wave equation, the boundary controlled Mindlin plate model, and the boundary controlled heat equation with a thermodynamically well-founded Hamiltonian (namely the *internal energy*, instead of the quadratic functional commonly used). Taking advantage of the strong underlying structure, we finally describe a unified object-oriented implementation of these models *via* PFEM. Companion interactive Jupyter notebooks [6] are discussed to illustrate our methodology. For the purpose of this manuscript, smooth functions are selected for the physical parameters of the problem under considerations. Nothing prevents from choosing less regular coefficients. This will of course affect the global convergence of the underlying finite elements [20].

Structure of the manuscript

The manuscript is organized as follows. In Section 2 the abstract pHs framework is introduced, with a particular focus on both hyperbolic and parabolic linear systems of partial differential equations. In Section 3 the general structure-preserving discretization is presented, and then specialized on the two cases previously mentioned. In Section 4 the ongoing environment SCRIMP is described in detail. In Section 5 the three companion interactive Jupyter notebooks [6] are thoroughly explained. Conclusions and perspectives are finally drawn in Section 6.

2. Definition of the General Framework

In this section, we introduce an abstract class of pHs and their underlying geometric structure: the Dirac structure for the finite-dimensional case and the Stokes-Dirac structure for the infinite-dimensional case. For the infinite-dimensional case it is shown how hyperbolic and parabolic systems easily fit into this framework.

2.1. Finite Dimensional Port-Hamiltonian Systems

State representation

Let us begin with a classical definition of a pHs in finite dimension. Consider the time-invariant dynamical system [24]:

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \left(\mathbf{J}(\mathbf{x}) - \mathbf{R}(\mathbf{x})\right)\nabla H(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{u}, \\ \mathbf{y} = \mathbf{B}(\mathbf{x})^{\mathrm{T}}\nabla H(\mathbf{x}), \end{cases} \tag{1}$$

where $H(\mathbf{x}): X \simeq \mathbb{R}^n \to \mathbb{R}$, the Hamiltonian, is a real-valued function of the vector of *energy variables* $\mathbf{x}$, bounded from below. Matrix-valued functions $\mathbf{J}(\mathbf{x})$ (the *structure operator*) and $\mathbf{R}(\mathbf{x})$ (the *dissipative* or *resistive operator*) are skew-symmetric and symmetric positive semi-definite respectively. The control $\mathbf{u} \in \mathbb{R}^m$ is applied thanks to the matrix-valued control function $\mathbf{B}(\mathbf{x})$ of size $n \times m$. Variable $\mathbf{y} \in \mathbb{R}^m$ is the power conjugated output to the input.

Such a system is called a *port-Hamiltonian system*, as it arises from the Hamiltonian modelling of a physical system and it interacts with the environment *via* the input $\mathbf{u}$ and the output $\mathbf{y}$, included in the formulation. The vector $\nabla H(\mathbf{x})$ is made of the *co-energy variables*.

Due to the structural properties of $\mathbf{J}(\mathbf{x})$ and $\mathbf{R}(\mathbf{x})$, the port-Hamiltonian system enjoys the nice following *power balance*:

$$\frac{\mathrm{d}H}{\mathrm{d}t} = -\left(\nabla H(\mathbf{x})\right)^{\mathrm{T}}\mathbf{R}(\mathbf{x})\nabla H(\mathbf{x}) + \mathbf{u}^{\mathrm{T}}\mathbf{y} \le \mathbf{u}^{\mathrm{T}}\mathbf{y}, \tag{2}$$

meaning that $\mathbf{R}(\mathbf{x})$ accounts for dissipation, and that the input-output product corresponds to the power supplied to (or taken from) the system, through the control $\mathbf{u}$.

Flow-effort representation

Consider two finite-dimensional vector spaces $E = F \simeq \mathbb{R}^n$. The elements of F are called *flows*, while the elements of E are called *efforts*. Those are port variables and their combination gives the power flowing inside the system. The space $B := F \times E$ is called the bond space of power variables. Therefore, identifying E as the dual of F, the power is defined as $\langle \mathbf{e}, \mathbf{f} \rangle := \mathbf{e}(\mathbf{f})$.

Definition 1 ([25], Def. 1.1.1) *Given the finite-dimensional space F and its dual E with respect to the inner product $\langle \cdot, \cdot \rangle_F : F \times F \to \mathbb{R}$, consider the symmetric bilinear form*:

$$\left\langle\!\left\langle (\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2) \right\rangle\!\right\rangle := \langle \mathbf{e}_1, \mathbf{f}_2 \rangle + \langle \mathbf{e}_2, \mathbf{f}_1 \rangle, \quad \text{where } (\mathbf{f}_i, \mathbf{e}_i) \in B,\ i = 1, 2.$$

A Dirac structure on $B := F \times E$ is a subspace $D \subset B$, which is maximally isotropic under $\langle\!\langle \cdot, \cdot \rangle\!\rangle$. Equivalently, a Dirac structure on $B := F \times E$ is a subspace $D \subset B$ which equals its orthogonal companion with respect to $\langle\!\langle \cdot, \cdot \rangle\!\rangle$, *i.e.* $D = D^{[\perp]}$, where:

$$D^{[\perp]} := \left\{ (\mathbf{f}, \mathbf{e}) \in B \,|\, \left\langle\!\left\langle (\mathbf{f}, \mathbf{e}), (\tilde{\mathbf{f}}, \tilde{\mathbf{e}}) \right\rangle\!\right\rangle = 0,\ \forall (\tilde{\mathbf{f}}, \tilde{\mathbf{e}}) \in D \right\}.$$

The connection between the concept of Dirac structure and pHs in its canon-

ical form (1) is achieved by considering the following *ports*:

- the *storage ports* $(\mathbf{f}_\mathbf{x}, \mathbf{e}_\mathbf{x}) := \left(\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}, \nabla H(\mathbf{x})\right) \in \mathbb{R}^n \times \mathbb{R}^n$, made of the *storage flow* $\mathbf{f}_\mathbf{x}$ (time-derivative of the energy variables) and *storage effort* $\mathbf{e}_\mathbf{x}$ (the co-energy variables);
- the *resistive* (*or dissipative*) *ports* $(\mathbf{f}_\mathbf{R}, \mathbf{e}_\mathbf{R}) \in \mathbb{R}^k \times \mathbb{R}^k$, made of the *resistive* (*or dissipative*) *flow* $\mathbf{f}_\mathbf{x}$ and *resisitive* (*or dissipative*) *effort* $\mathbf{e}_\mathbf{R}$;
- the *interconnection ports* $(\mathbf{f}_\mathbf{u}, \mathbf{e}_\mathbf{u}) := (-\mathbf{y}, \mathbf{u}) \in \mathbb{R}^m \times \mathbb{R}^m$, made of the *interconnection flow* $\mathbf{f}_\mathbf{u}$ and *interconnection effort* $\mathbf{e}_\mathbf{u}$.

Assuming that the matrix $\mathbf{R}(\mathbf{x})$ has constant rank, from classical matrix factorizations there exist matrices $\mathbf{G}$ (not necessarily square, of size $k \times n$) and $\mathbf{K}$ symmetric positive semi-definite of size $k \times k$ such that $\mathbf{R} = \mathbf{G}^\mathrm{T}\mathbf{K}\mathbf{G}$. These notations at hand, the pHs (1) rewrites:

$$\begin{pmatrix} \mathbf{f}_\mathbf{x}(\mathbf{x}) \\ \mathbf{f}_\mathbf{R}(\mathbf{x}) \\ \mathbf{f}_\mathbf{u}(\mathbf{x}) \end{pmatrix} = \underbrace{\begin{bmatrix} \mathbf{J}(\mathbf{x}) & -\mathbf{G}(\mathbf{x})^\mathrm{T} & \mathbf{B}(\mathbf{x}) \\ \mathbf{G}(\mathbf{x}) & \mathbf{0} & \mathbf{0} \\ -\mathbf{B}(\mathbf{x})^\mathrm{T} & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathbf{J}_e} \begin{pmatrix} \mathbf{e}_\mathbf{x}(\mathbf{x}) \\ \mathbf{e}_\mathbf{R}(\mathbf{x}) \\ \mathbf{e}_\mathbf{u}(\mathbf{x}) \end{pmatrix}, \tag{3}$$

together with the (*dissipative or resistive*) *constitutive relation*:

$$\mathbf{e}_\mathbf{R}(\mathbf{x}) = \mathbf{K}(\mathbf{x})\mathbf{f}_\mathbf{R}(\mathbf{x}). \tag{4}$$

It is clear that the *extended structure operator* $\mathbf{J}_e$ appearing in (3) is skew-symmetric of size $(n+k+m)\times(n+k+m)$. Its graph is a Dirac structure with respect to the Euclidean inner product, as a kernel representation, see [24]. Hence, it comes:

$$\langle \mathbf{e}_\mathbf{x}(\mathbf{x}), \mathbf{f}_\mathbf{x}(\mathbf{x}) \rangle_{\mathbb{R}^n} + \langle \mathbf{e}_\mathbf{R}(\mathbf{x}), \mathbf{f}_\mathbf{R}(\mathbf{x}) \rangle_{\mathbb{R}^k} + \langle \mathbf{e}_\mathbf{u}(\mathbf{x}), \mathbf{f}_\mathbf{u}(\mathbf{x}) \rangle_{\mathbb{R}^m} = 0.$$

Noting that $\frac{\mathrm{d}H}{\mathrm{d}t} = \langle \mathbf{e}_\mathbf{x}(\mathbf{x}), \mathbf{f}_\mathbf{x}(\mathbf{x}) \rangle_{\mathbb{R}^n}$ (by definition of the storage port) leads to:

$$\frac{\mathrm{d}H}{\mathrm{d}t} = -\langle \mathbf{e}_\mathbf{R}(\mathbf{x}), \mathbf{f}_\mathbf{R}(\mathbf{x}) \rangle_{\mathbb{R}^k} - \langle \mathbf{e}_\mathbf{u}(\mathbf{x}), \mathbf{f}_\mathbf{u}(\mathbf{x}) \rangle_{\mathbb{R}^m},$$

and thanks to the symmetry of $\mathbf{R}$, (4) gives:

$$\frac{\mathrm{d}H}{\mathrm{d}t} = -\langle \mathbf{f}_\mathbf{R}(\mathbf{x}), \mathbf{K}(\mathbf{x})\mathbf{f}_\mathbf{R}(\mathbf{x}) \rangle_{\mathbb{R}^m} - \langle \mathbf{e}_\mathbf{u}(\mathbf{x}), \mathbf{f}_\mathbf{u}(\mathbf{x}) \rangle_{\mathbb{R}^m}.$$

Finally, from (3) and the definition of the storage and interconnection ports, the power balance (2) is recovered.

The relation between (1) and (3)-(4) can be understood as follows: the power balance (2) is *encoded* in the Dirac structure (obtained from the extended structure operator $\mathbf{J}_e$) together with the resistive constitutive relation.

Remark 1. *The canonical Euclidean inner product has been used here, but other inner products are allowed to take into account mass matrices* (*symmetric positive definite*) *on the left-hand side of* (1), (3), *and* (4). *This is crucial after the spatial discretization procedure. This corresponds to a kernel representation of*

Dirac structure [24].

System 1 is a pHs in canonical form. Recently, finite-dimensional differential algebraic port-Hamiltonian systems (pHDAE) have been introduced both for linear [26] and nonlinear systems [27]. This enriched description shares not only all the crucial features of ordinary pHs, but also easily accounts for algebraic constraints, time-dependent transformations and explicit dependence on time in the Hamiltonian. The application of the proposed discretization method naturally leads to pHDAEs. Indeed, a constitutive relation between $\mathbf{f}_x := \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}$ and $\mathbf{e}_x := \nabla H(\mathbf{x})$ is needed to be well-defined. But PFEM takes into account constitutive relations apart from (3) as constraints. However, as shown later in Sections 3.2 and 3.3, the method simplifies in the case, for instance, of linear hyperbolic systems.

2.2. Infinite-Dimensional Port-Hamiltonian Systems

In this section an infinite-dimensional generalization of pHs is presented. For sake of readability, the (Stokes-) Dirac structure is first defined, and secondly, infinite-dimensional pHs are then described in both hyperbolic and parabolic cases. A more general framework can be designed, but this goes beyond the aim of this present work.

Structure operator

As to avoid functional difficulties, the analogue of the extended structure operator will not be written as in (3). More precisely, the control operator will not be included in an extended structure unbounded operator, but given apart. The Stokes-Dirac will be then obtained thanks to a structure operator related to the boundary control operator through an abstract Green formula. However, like in finite dimension, the aim is to establish a link between flow and effort variables. Most importantly, the underlying Stokes-Dirac structure must encode the power balance of the dynamical system under study.

Consider a Lipschitz domain $\Omega \subset \mathbb{R}^d$, $d \in \{1,2,3\}$, and the relation:

$$\begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix} = \begin{bmatrix} \mathbf{0} & -\mathcal{L}^* \\ \mathcal{L} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \begin{matrix} \boldsymbol{e}_1 \in H^{\mathcal{L}}, \\ \boldsymbol{e}_2 \in H^{-\mathcal{L}^*}, \end{matrix} \tag{5}$$

with:

$$H^{\mathcal{L}} : \left\{ \boldsymbol{e}_1 \in L^2(\Omega, \mathbb{A}) \,|\, \mathcal{L}\boldsymbol{e}_1 \in L^2(\Omega, \mathbb{B}) \right\},$$
$$H^{-\mathcal{L}^*} : \left\{ \boldsymbol{e}_2 \in L^2(\Omega, \mathbb{B}) \,|\, -\mathcal{L}^*\boldsymbol{e}_2 \in L^2(\Omega, \mathbb{A}) \right\}.$$

By $L^2(\Omega, \mathbb{X})$ we denote the space of square integrable $\mathbb{X}$-valued functions. Symbol $\mathbb{A}, \mathbb{B}$ denote either the space of scalars $\mathbb{R}$, vectors $\mathbb{R}^d$, symmetric tensors $\mathbb{R}^{d\times d}_{\mathrm{sym}} =: \mathbb{S}$ or a Cartesian product of those, depending on the particular example. The operator $\mathcal{L} : H^{\mathcal{L}} \to L^2(\Omega, \mathbb{B})$ is a generic differential, and therefore linear but unbounded, operator. The notation $\mathcal{L}^* : H^{-\mathcal{L}^*} \to L^2(\Omega, \mathbb{A})$ denotes the formal adjoint of $\mathcal{L}$, defined by the relation:

$$\left\langle \mathcal{L}\boldsymbol{e}_1, \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} = \left\langle \boldsymbol{e}_1, \mathcal{L}^*\boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{A})}, \quad \boldsymbol{e}_1 \in C_0^\infty(\Omega,\mathbb{A}), \boldsymbol{e}_2 \in C_0^\infty(\Omega,\mathbb{B}). \tag{6}$$

Of course, for (5) to be well-defined, constitutive relations are needed. Only physical laws will be taken into account when constructing the above relation on concrete examples. As pointed out in the introduction, PFEM aims at both preserving this relation and discretizing constitutive laws to close the system.

Remark 2. *One can be confused by the lack of evolution in time in* (5)*. However, this emphasizes an important paradigm in the proposed point of view*: *this relation translates the time-independent geometric structure of the pHs as an equation with differential operator* (*ill-posed on its own*), *while constitutive relations will bring back the time dependency of the problem. In particular*, *some flows must be the time derivative of the energy variables.*

Throughout the paper, $\langle \cdot,\cdot \rangle_X$ denotes the inner product of the Hilbert space X. Definition (6) is analogous to Definition 5.80 in [28]. In Section 3.2, the operator $\mathcal{L}$ is the gradient, denoted by grad, and its formal adjoint is minus the divergence, denoted by -div, from the so-called Green's formula (integration by parts). In Section 3.3, the operator $\mathcal{L}$ contains both grad and Grad. This latter corresponds to the symmetric part of the gradient and represents the deformation tensor in continuum mechanics:

$$\mathrm{Grad}(\boldsymbol{e}) := \frac{1}{2}\left(\nabla \boldsymbol{e} + \nabla^{\mathrm{T}}\boldsymbol{e}\right), \quad \boldsymbol{e} \in C^\infty\left(\mathbb{R}^d\right).$$

The formal adjoint of Grad is minus the tensor divergence -Div. For a tensor field $\boldsymbol{E}: \Omega \to \mathbb{M} := \mathbb{R}^{d\times d}$, with components e_{ij}, the divergence is a vector, defined columnwise as:

$$\mathrm{Div}(\boldsymbol{E}) := \left(\sum_{i=1}^{d} \partial_{x_i} e_{ij}\right)_{j=1,\cdots,d}.$$

Finally, in Section 3.4, $\mathcal{L}$ is made of the gradient and the identity operator.

Stokes-Dirac structure

Definition 1 still remains valid in infinite dimension. Nevertheless, as stated above, the structure operator in (5) is not extended to include the control operator. Hence an additional assumption has to be made for $\begin{bmatrix} \mathbf{0} & -\mathcal{L}^* \\ \mathcal{L} & \mathbf{0} \end{bmatrix}$ to define a Dirac structure in relation with a pHs coming from boundary control of partial differential equations. In other words, a Stokes-Dirac structure requires the specification of boundary variables in order to express a general power conservation property for *open* physical systems. This assumption is based on the so-called Stokes' theorem (also known as the divergence theorem, Gauss's theorem or Ostrogradsky's theorem) and its corollaries, as the Green's formula.

Assumption 1 (Abstract Green's formula) *The operator* $\mathcal{L}$ *is assumed to satisfy the abstract Green's formula*:

$$\left\langle \mathcal{L}\boldsymbol{e}_1, \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} - \left\langle \boldsymbol{e}_1, \mathcal{L}^*\boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{A})} = \left\langle \Gamma_0 \boldsymbol{e}_1, \Gamma_\perp \boldsymbol{e}_2 \right\rangle_{V_\partial,(V_\partial)'}, \quad \forall\, \boldsymbol{e}_1 \in H^{\mathcal{L}}, \boldsymbol{e}_2 \in H^{-\mathcal{L}^*}, \tag{7}$$

where the right-hand side is the duality bracket at the boundary, on a well-suited

boundary functional space V_∂ for some trace operators $\Gamma_0, \Gamma_\perp$. From now on, this duality bracket will be denoted by $\langle \cdot, \cdot \rangle_{\partial\Omega}$ with a slight abuse of notation.

Remark 3. *This abstract formula is well-known in the boundary control systems theory, see* e.g. ([29], *Chapter* 10).

Remark 4. *In practice, Equation* (7) *dictates the causalities, i.e. the possible choices for the boundary control* $\boldsymbol{u}_\partial$ *and the boundary observation* $\boldsymbol{y}_\partial$, *via the equality* $\langle \Gamma_0 \boldsymbol{e}_1, \Gamma_\perp \boldsymbol{e}_2 \rangle_{\partial\Omega} = \langle \boldsymbol{u}_\partial, \boldsymbol{y}_\partial \rangle_{\partial\Omega}$ (*with a slight abuse of notation for the right-hand side to make sense*). *Of course, the admissible causalities are also related to the well-posedness of the system under study, and in particular to the definitions of the boundary functional spaces.*

For sake of simplicity, a focus on the two following causalities will be made. Let the boundary variables associated to system (5) be defined by:

$$\boldsymbol{e}_\partial = \Gamma_\perp \boldsymbol{e}_2 \in (V_\partial)', \quad \boldsymbol{f}_\partial = -\Gamma_0 \boldsymbol{e}_1 \in V_\partial, \tag{8}$$

or the other way:

$$\boldsymbol{e}_\partial = \Gamma_0 \boldsymbol{e}_1 \in V_\partial, \quad \boldsymbol{f}_\partial = -\Gamma_\perp \boldsymbol{e}_2 \in (V_\partial)'. \tag{9}$$

In light of (7), systems:

$$\begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix} = \begin{bmatrix} \boldsymbol{0} & -\mathcal{L}^* \\ \mathcal{L} & \boldsymbol{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{e}_\partial \\ \boldsymbol{f}_\partial \end{pmatrix} = \begin{bmatrix} \boldsymbol{0} & \Gamma_\perp \\ -\Gamma_0 & \boldsymbol{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \tag{10}$$

and:

$$\begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix} = \begin{bmatrix} \boldsymbol{0} & -\mathcal{L}^* \\ \mathcal{L} & \boldsymbol{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{e}_\partial \\ \boldsymbol{f}_\partial \end{pmatrix} = \begin{bmatrix} \Gamma_0 & \boldsymbol{0} \\ \boldsymbol{0} & -\Gamma_\perp \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \tag{11}$$

define Stokes-Dirac structures with respect to the bilinear pairing:

$$\begin{aligned} &\left\langle\!\left\langle \left(\boldsymbol{f}_1^1, \boldsymbol{f}_2^1, \boldsymbol{f}_\partial^1, \boldsymbol{e}_1^1, \boldsymbol{e}_2^1, \boldsymbol{e}_\partial^1 \right), \left(\boldsymbol{f}_1^2, \boldsymbol{f}_2^2, \boldsymbol{f}_\partial^2, \boldsymbol{e}_1^2, \boldsymbol{e}_2^2, \boldsymbol{e}_\partial^2 \right) \right\rangle\!\right\rangle \\ &= \left\langle \boldsymbol{f}_1^1, \boldsymbol{e}_1^2 \right\rangle_{L^2(\Omega,\mathbb{A})} + \left\langle \boldsymbol{f}_2^1, \boldsymbol{e}_2^2 \right\rangle_{L^2(\Omega,\mathbb{B})} + \left\langle \boldsymbol{f}_1^2, \boldsymbol{e}_1^1 \right\rangle_{L^2(\Omega,\mathbb{A})} \\ &\quad + \left\langle \boldsymbol{f}_2^2, \boldsymbol{e}_2^1 \right\rangle_{L^2(\Omega,\mathbb{B})} + \left\langle \boldsymbol{f}_\partial^1, \boldsymbol{e}_\partial^2 \right\rangle_{\partial\Omega} + \left\langle \boldsymbol{f}_\partial^2, \boldsymbol{e}_\partial^1 \right\rangle_{\partial\Omega}. \end{aligned}$$

Obviously, for systems (10) and (11) to be well-defined, constitutive relations are needed.

In the remaining part of this section, only (10) will be treated in details. The other canonical causality (11), figuring in Section 3.4, straightforwardly follows with the same strategy.

Hyperbolic systems

In the hyperbolic case, both flows represent the dynamics of the independent energy variables $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2$. The Hamiltonian is a generic functional of these variables $H = H(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2)$. The co-energy variables are by definition the *variational derivatives* (see e.g. [1]) of H with respect to the energy variables:

$$\boldsymbol{f}_1 = \partial_t \boldsymbol{\alpha}_1, \quad \boldsymbol{e}_1 := \delta_{\boldsymbol{\alpha}_1} H, \quad \boldsymbol{f}_2 = \partial_t \boldsymbol{\alpha}_2, \quad \boldsymbol{e}_2 := \delta_{\boldsymbol{\alpha}_2} H. \tag{12}$$

Then system (10) possesses the equivalent state representation:

$$\begin{pmatrix} \partial_t \boldsymbol{\alpha}_1 \\ \partial_t \boldsymbol{\alpha}_2 \end{pmatrix} = \begin{bmatrix} \mathbf{0} & -\mathcal{L}^* \\ \mathcal{L} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \delta_{\boldsymbol{\alpha}_1} H \\ \delta_{\boldsymbol{\alpha}_2} H \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{u}_\partial \\ \boldsymbol{y}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & \Gamma_\perp \\ \Gamma_0 & \mathbf{0} \end{bmatrix} \begin{pmatrix} \delta_{\boldsymbol{\alpha}_1} H \\ \delta_{\boldsymbol{\alpha}_2} H \end{pmatrix}. \tag{13}$$

It holds $\boldsymbol{u}_\partial = \boldsymbol{e}_\partial$, $\boldsymbol{y}_\partial = -\boldsymbol{f}_\partial$. The power balance is naturally embedded in the Stokes-Dirac structure defined by (10):

$$\frac{\mathrm{d}}{\mathrm{d}t} H(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2) = \langle \boldsymbol{y}_\partial, \boldsymbol{u}_\partial \rangle_{\partial\Omega}. \tag{14}$$

Linear hyperbolic systems

The system is linear when the Hamiltonian has the form:

$$H(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2) = \frac{1}{2} \langle \boldsymbol{\alpha}_1, \mathcal{Q}_1 \boldsymbol{\alpha}_1 \rangle_{L^2(\Omega,\mathbb{A})} + \frac{1}{2} \langle \boldsymbol{\alpha}_2, \mathcal{Q}_2 \boldsymbol{\alpha}_2 \rangle_{L^2(\Omega,\mathbb{B})},$$

where $\mathcal{Q}_1 : L^2(\Omega, \mathbb{A}) \to L^2(\Omega, \mathbb{A})$, $\mathcal{Q}_2 : L^2(\Omega, \mathbb{B}) \to L^2(\Omega, \mathbb{B})$ are positive symmetric operators, bounded from below and above:

$$m_1 \boldsymbol{I}_\mathbb{A} \preceq \mathcal{Q}_1 \preceq M_1 \boldsymbol{I}_\mathbb{A}, \quad m_2 \boldsymbol{I}_\mathbb{B} \preceq \mathcal{Q}_2 \preceq M_2 \boldsymbol{I}_\mathbb{B},$$

$$m_1 > 0, m_2 > 0, M_1 > 0, M_2 > 0,$$

with $\boldsymbol{I}_\mathbb{A}$ and $\boldsymbol{I}_\mathbb{B}$ the identity operators in $L^2(\Omega, \mathbb{A})$ and $L^2(\Omega, \mathbb{B})$ respectively. In this case, the co-energy variables are given by:

$$\boldsymbol{e}_1 := \delta_{\alpha_1} H = \mathcal{Q}_1 \boldsymbol{\alpha}_1, \quad \boldsymbol{e}_2 := \delta_{\alpha_2} H = \mathcal{Q}_2 \boldsymbol{\alpha}_2. \tag{15}$$

Since $\mathcal{Q}_1, \mathcal{Q}_2$ are positive and bounded from below and above, it is possible to invert them to obtain:

$$\boldsymbol{\alpha}_1 = \mathcal{Q}_1^{-1} \boldsymbol{e}_1 =: \mathcal{M}_1 \boldsymbol{e}_1, \quad \boldsymbol{\alpha}_2 = \mathcal{Q}_2^{-1} \boldsymbol{e}_2 =: \mathcal{M}_2 \boldsymbol{e}_2, \tag{16}$$

giving rise to the *co-energy formulation*. The Hamiltonian is rewritten as:

$$H(\boldsymbol{e}_1, \boldsymbol{e}_2) = \frac{1}{2} \langle \boldsymbol{e}_1, \mathcal{M}_1 \boldsymbol{e}_1 \rangle_{L^2(\Omega,\mathbb{A})} + \frac{1}{2} \langle \boldsymbol{e}_2, \mathcal{M}_2 \boldsymbol{e}_2 \rangle_{L^2(\Omega,\mathbb{B})} \tag{17}$$

and a linear hyperbolic pHs (10) can be expressed as:

$$\begin{bmatrix} \mathcal{M}_1 & \mathbf{0} \\ \mathbf{0} & \mathcal{M}_2 \end{bmatrix} \begin{pmatrix} \partial_t \boldsymbol{e}_1 \\ \partial_t \boldsymbol{e}_2 \end{pmatrix} = \begin{bmatrix} \mathbf{0} & -\mathcal{L}^* \\ \mathcal{L} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{u}_\partial \\ \boldsymbol{y}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & \Gamma_\perp \\ \Gamma_0 & \mathbf{0} \end{bmatrix} \begin{pmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{pmatrix}. \tag{18}$$

In this particular case, the constitutive relations needed for system (10) to be well-defined are given by (15), and then directly included in (18). In Sections 3.2 and 3.3, it will be shown that PFEM leads directly to a finite-dimensional pHs of the form (1) with $\mathbf{R}(\mathbf{x}) = 0$. This simplification considerably facilitates the solution in time, as (1) is an Ordinary Differential Equation (ODE). Indeed, some dissipation phenomena will induce a differential-algebraic structure on the problem, for example dissipative effects due to an unbounded (e.g. elliptic) operator (see Remark 5).

Parabolic systems

In this case, the first flow $\boldsymbol{f}_1$ still represents a dynamics $\partial_t \boldsymbol{\alpha}_1$ of the energy variable $\boldsymbol{\alpha}_1$. The Hamiltonian then reads $H = H(\boldsymbol{\alpha}_1)$, and its variational derivative gives the co-energy variable $\boldsymbol{e}_1 = \delta_{\boldsymbol{\alpha}_1} H$.

The second flow $\boldsymbol{f}_2$ represents an extra flow related to the effort variable $\boldsymbol{e}_2$

appearing in the dynamics of the energy variable $\boldsymbol{\alpha}_1$. The relation is given implicitly by a mapping $\mathcal{G}$ as $\mathcal{G}(\boldsymbol{f}_2, \boldsymbol{e}_2) = 0$. Then, pHs (10) of parabolic type is expressed as:

$$\begin{pmatrix} \partial_t \boldsymbol{\alpha}_1 \\ \boldsymbol{f}_2 \end{pmatrix} = \begin{bmatrix} \mathbf{0} & -\mathcal{L}^* \\ \mathcal{L} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \delta_{\boldsymbol{\alpha}_1} H \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \begin{pmatrix} \boldsymbol{u}_\partial \\ \boldsymbol{y}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & \Gamma_\perp \\ \Gamma_0 & \mathbf{0} \end{bmatrix} \begin{pmatrix} \delta_{\boldsymbol{\alpha}_1} H \\ \boldsymbol{e}_2 \end{pmatrix}, \quad \mathcal{G}(\boldsymbol{f}_2, \boldsymbol{e}_2) = 0. \tag{19}$$

In Section 3.4, an example of a parabolic-type pHs (11) is studied. It will be shown that the PFEM structure-preserving discretization of such a system naturally leads to a finite-dimensional pHDAE. Again, the power balance is naturally embedded in the Stokes-Dirac structure defined by (10):

$$\frac{\mathrm{d}}{\mathrm{d}t} H(\boldsymbol{\alpha}_1) = -\langle \boldsymbol{f}_2, \boldsymbol{e}_2 \rangle_{L^2(\Omega)} + \langle \boldsymbol{y}_\partial, \boldsymbol{u}_\partial \rangle_{\partial\Omega}. \tag{20}$$

In practice, this becomes explicit with the constitutive relation $\mathcal{G}(\boldsymbol{f}_2, \boldsymbol{e}_2) = 0$ as it will be seen in Section 3.4 (and more generally in [9] [10]). Note that this latter relation has to be accurately discretized to ensure that the discretized power balance mimics the continuous one.

Remark 5. *By adding resistive port*(*s*), *dissipation*(*s*) *can easily be taken into account* (*both internal or at the boundary*), *as done in the finite-dimensional case via* $\mathbf{R}(\mathbf{x})$ *playing the role of a output feedback gain matrix. In this case, the system becomes a parabolic system, the dissipative constitutive relation being represented by* $\mathcal{G}$. *See* [30] [31] *for a detailed discussion about structure-preserving discretization of dissipative systems.*

3. The Partitioned Finite Element Method (PFEM)

We are now in a position to introduce a general methodology to discretize infinite-dimensional pHs in a structure-preserving manner. The main contribution in this section is the application of PFEM to a general abstract class of pHs, unifying the previously published results. This generality is notably of particular interest for the development of a well-structured software for the numerical simulations of physics-based models. The power balances (14), (20) are deeply linked to a linear underlying Stokes-Dirac. The main idea of PFEM is to mimic this structure, in order to obtain a discretized copy of these power balances as (2). This systematically translates the Stokes-Dirac structure into a finite-dimensional Dirac structure. The compatible discretization, with respect to this Dirac structure, of the constitutive relations allows to mimic the continuous power-balance. This method goes under the name Partitioned Finite Element Method (PFEM), and was originally presented in [8]. The procedure is a natural extension of MFEM to pHs and boils down to these three simple steps:

1) System (5) is written in weak form;

2) The integration by parts (7) is carried out on a partition of the system (5) to make the appropriate boundary control appear;

3) A Galerkin method is employed to obtain a finite-dimensional system. For the approximation basis, the finite element method is used here but spectral

methods can be chosen as well.

This strategy of structured discretization in order to mimic the continuous power balance at the discrete level has been addressed for closed abstract linear hyperbolic systems in [21]. This pioneering work already proposed the key point of PFEM: the integration by parts on a partition of the weak formulation of the system. The author called the obtained systems *primal-dual* or *dual-primal* formulation, depending on which line is integrated by parts. In the port-Hamiltonian formalism, systems are opened with control and observation. It appears that [21] admits PFEM as a generalization for structure-preserving space discretization. The choice of a control in the pHs community is called a *causality*, and *primal-dual* or *dual-primal* correspond in this work to the *canonical* causalities (10) and (11) respectively.

3.1. General Strategy

Consider smooth test functions $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ and the weak form of (5):

$$\begin{aligned} \left\langle \boldsymbol{v}_1, \boldsymbol{f}_1 \right\rangle_{L^2(\Omega,\mathbb{A})} &= \left\langle \boldsymbol{v}_1, -\mathcal{L}^* \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{A})}, \\ \left\langle \boldsymbol{v}_2, \boldsymbol{f}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} &= \left\langle \boldsymbol{v}_2, \mathcal{L} \boldsymbol{e}_1 \right\rangle_{L^2(\Omega,\mathbb{B})}. \end{aligned} \tag{21}$$

Next the integration by parts is performed either on the first or on the second line (the system is *partitioned*), depending on the causality.

Integration by parts of the term $\left\langle \boldsymbol{v}_1, -\mathcal{L}^* \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{A})}$

In this case case, using (7), it is obtained:

$$-\left\langle \boldsymbol{v}_1, \mathcal{L}^* \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{A})} = -\left\langle \mathcal{L} \boldsymbol{v}_1, \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} + \left\langle \Gamma_0 \boldsymbol{v}_1, \Gamma_\perp \boldsymbol{e}_2 \right\rangle_{\partial\Omega}.$$

The boundary variable $\boldsymbol{e}_\partial \coloneqq \Gamma_\perp \boldsymbol{e}_2$ in (10) explicitly appears. Then the equation defining the corresponding $\boldsymbol{f}_\partial \coloneqq \Gamma_0 \boldsymbol{e}_1$ is put into weak form to obtain the final system for all smooth test functions $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, and $\boldsymbol{v}_\partial$:

$$\begin{aligned} \left\langle \boldsymbol{v}_1, \boldsymbol{f}_1 \right\rangle_{L^2(\Omega,\mathbb{A})} &= -\left\langle \mathcal{L} \boldsymbol{v}_1, \boldsymbol{e}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} + \left\langle \Gamma_0 \boldsymbol{v}_1, \boldsymbol{e}_\partial \right\rangle_{\partial\Omega}, \\ \left\langle \boldsymbol{v}_2, \boldsymbol{f}_2 \right\rangle_{L^2(\Omega,\mathbb{B})} &= \left\langle \boldsymbol{v}_2, \mathcal{L} \boldsymbol{e}_1 \right\rangle_{L^2(\Omega,\mathbb{B})}, \\ \left\langle \boldsymbol{v}_\partial, \boldsymbol{f}_\partial \right\rangle_{\partial\Omega} &= -\left\langle \boldsymbol{v}_\partial, \Gamma_0 \boldsymbol{e}_1 \right\rangle_{\partial\Omega}. \end{aligned} \tag{22}$$

Now, a Galerkin discretization is introduced. Test, energy and co-energy functions with the same subscript are discretized using the same basis, for all $t \geq 0$:

$$\begin{aligned} \square_1(t,\boldsymbol{x}) &\approx \square_1^d(t,\boldsymbol{x}) \coloneqq \sum_{i=1}^{N_1} \boldsymbol{\varphi}_1^i(\boldsymbol{x})\, \square_1^i(t), \quad \forall \boldsymbol{x} \in \Omega, \\ \square_2(t,\boldsymbol{x}) &\approx \square_2^d(t,\boldsymbol{x}) \coloneqq \sum_{i=1}^{N_2} \boldsymbol{\varphi}_2^i(\boldsymbol{x})\, \square_2^i(t), \quad \forall \boldsymbol{x} \in \Omega, \\ \square_\partial(t,\boldsymbol{s}) &\approx \square_\partial^d(t,\boldsymbol{s}) \coloneqq \sum_{i=1}^{N_\partial} \boldsymbol{\psi}_\partial^i(\boldsymbol{s})\, \square_\partial^i(t), \quad \forall \boldsymbol{s} \in \partial\Omega, \end{aligned} \tag{23}$$

where $\square$ stands for v, f, and e and $\boldsymbol{\varphi}_1^i \in H^{\mathcal{L}}$, $\boldsymbol{\varphi}_2^i \in L^2(\Omega,\mathbb{B})$, and $\boldsymbol{\psi}_\partial^i \in L^2(\partial\Omega,\mathbb{R}^m)$.

Remark 6. *In general, a discretization in the same basis of either* $(\mathbf{f}_1, \mathbf{e}_1)$ *and* $(\mathbf{f}_2, \mathbf{e}_2)$ *or* $(\mathbf{f}_1, \mathbf{f}_2)$ *and* $(\mathbf{e}_1, \mathbf{e}_2)$ *(as done in* [16]*) must be performed. The former is our choice since it directly leads to square mass matrices, while the latter may be more appropriate when dealing for instance with Maxwell's equations for electromagnetics, see* [32] *and references therein for details on the difficulties that may then occur.*

Then plugging the approximations into (22), it is computed:

$$\begin{bmatrix} \mathbf{M}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M}_\partial \end{bmatrix} \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{D}_{\mathcal{L}}^{\mathrm{T}} & \mathbf{B}_\perp \\ \mathbf{D}_{\mathcal{L}} & \mathbf{0} & \mathbf{0} \\ -\mathbf{B}_\perp^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_\partial \end{pmatrix}, \tag{24}$$

where vectors $\mathbf{f}_1$, $\mathbf{f}_2$, $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{f}_\partial$, and $\mathbf{e}_\partial$ are given by the column-wise concatenation of the respective degrees of freedom of $\boldsymbol{f}_1^d$, $\boldsymbol{f}_2^d$, $\boldsymbol{e}_1^d$, $\boldsymbol{e}_2^d$, $\boldsymbol{f}_\partial^d$, and $\boldsymbol{e}_\partial^d$, and where the matrices are defined as follows:

$$\begin{aligned} M_1^{ij} &= \left\langle \boldsymbol{\varphi}_1^i, \boldsymbol{\varphi}_1^j \right\rangle_{L^2(\Omega,\mathbb{A})}, & D_{\mathcal{L}}^{mi} &= \left\langle \boldsymbol{\varphi}_2^m, \mathcal{L}\boldsymbol{\varphi}_1^i \right\rangle_{L^2(\Omega,\mathbb{B})}, \\ M_2^{mn} &= \left\langle \boldsymbol{\varphi}_2^m, \boldsymbol{\varphi}_2^n \right\rangle_{L^2(\Omega,\mathbb{B})}, & B_\perp^{ik} &= \left\langle \Gamma_0 \boldsymbol{\varphi}_1^i, \boldsymbol{\psi}_\partial^k \right\rangle_{\partial\Omega}, \\ M_\partial^{lk} &= \left\langle \boldsymbol{\psi}_\partial^l, \boldsymbol{\psi}_\partial^k \right\rangle_{\partial\Omega}, \end{aligned} \tag{25}$$

where $1 \le i, j \le N_1$, $1 \le m, n \le N_2$, and $1 \le l, k \le N_\partial$. System (24) is a kernel representation of a Dirac structure as in (3) (see Remark 1).

Remark 7. *Note that matrices* $\mathbf{D}_{\mathcal{L}}$ *and* $\mathbf{B}_\perp$ *are not square.*

The discrete Hamiltonian is naturally defined as the continuous one evaluated in the discrete energy variables. As done in Section 2, it is easier to distinguish the linear hyperbolic from the parabolic case.

Hyperbolic case

In this setting, the flows $\boldsymbol{f}_i$, $i = 1, 2$, are given by the time derivative of the energy variables $\boldsymbol{\alpha}_i$. Hence, the discretization of these energy variables is given by:

$$\boldsymbol{\alpha}_i^d(t, \boldsymbol{x}) = \boldsymbol{\alpha}_i^d(0, \boldsymbol{x}) + \int_0^t \boldsymbol{f}_i^d(s, \boldsymbol{x}) \mathrm{d}s, \quad i = 1, 2.$$

The discrete Hamiltonian is then defined by $H^d(\underline{\alpha}_1, \underline{\alpha}_2) := H(\boldsymbol{\alpha}_1^d, \boldsymbol{\alpha}_2^d)$, where $\underline{\alpha}_1$ and $\underline{\alpha}_2$ are the column-wise concatenation of the time varying coefficients of $\boldsymbol{\alpha}_1^d$ and $\boldsymbol{\alpha}_2^d$ in their respective basis.

Definition 2. *The discretization of the constitutive relations is said to be compatible if and only if:*

$$\nabla H^d = \begin{pmatrix} \nabla_{\underline{\alpha}_1} H^d \\ \nabla_{\underline{\alpha}_2} H^d \end{pmatrix} = \begin{bmatrix} \mathbf{M}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 \end{bmatrix} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{pmatrix}.$$

Proposition 1. *If the discretisation of the constitutive relations is compatible, the discrete power balance reads at the discrete level:*

$$\frac{\mathrm{d}}{\mathrm{d}t} H^d = -(\mathbf{e}_\partial)^{\mathrm{T}} \mathbf{M}_\partial \mathbf{f}_\partial,$$

which perfectly mimics the continuous identity.

Proof 1. *A straightforward computation gives*:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}H^d &= \left(\nabla_{\underline{\alpha}_1}H^d\right)^{\mathrm{T}}\dot{\underline{\alpha}}_1 + \left(\nabla_{\underline{\alpha}_2}H^d\right)^{\mathrm{T}}\dot{\underline{\alpha}}_2\\ &= \left(\mathbf{M}_1\mathbf{e}_1\right)^{\mathrm{T}}\mathbf{f}_1 + \left(\mathbf{M}_2\mathbf{e}_2\right)^{\mathrm{T}}\mathbf{f}_2\\ &= \left(\mathbf{e}_1\right)^{\mathrm{T}}\mathbf{M}_1\mathbf{f}_1 + \left(\mathbf{e}_2\right)^{\mathrm{T}}\mathbf{M}_2\mathbf{f}_2\\ &= -\left(\mathbf{e}_\partial\right)^{\mathrm{T}}\mathbf{M}_\partial\mathbf{f}_\partial,\end{aligned}$$

where the symmetry of the mass matrices and the Dirac structure have been used.

Remark 8. *In the special case of linear hyperbolic systems, it has been seen that the co-energy formulation allows to take the constitutive relations into account directly in the differential equations. Applying PFEM to* (18) *then leads to an* ODE, *and the constitutive relations are then automatically discretized in a compatible manner.*

Parabolic case

In this setting, only the flow $\boldsymbol{f}_1$ is the time derivative of the energy variable $\boldsymbol{\alpha}_1$. This energy variable is discretized as in the hyperbolic case. The discrete Hamiltonian is then defined by $H^d\left(\underline{\alpha}_1\right) := H\left(\boldsymbol{\alpha}_1^d\right)$.

Definition 3. *The discretization of the constitutive relation is said to be compatible if and only if*: $\nabla H^d = \mathbf{M}_1\mathbf{e}_1$.

Proposition 2. *If the discretisation of the constitutive relations is compatible, the discrete power balance reads at the discrete level*:

$$\frac{\mathrm{d}}{\mathrm{d}t}H^d = -\mathbf{e}_2^{\mathrm{T}}\mathbf{M}_2\mathbf{f}_2 - \mathbf{e}_\partial^{\mathrm{T}}\mathbf{M}_\partial\mathbf{f}_\partial,$$

which perfectly mimics the continuous identity.

Proof 2. *The proof can be derived similarly as in the hyperbolic case.*

Remark 9. *Of course, an accurate discretization of the implicit constitutive relation* $\mathcal{G}\left(\boldsymbol{f}_2, \boldsymbol{e}_2\right) = 0$ *is also required to conclude. This will be illustrated in Section* 3.4.

Integration by parts of the term $\left\langle \boldsymbol{v}_2, \mathcal{L}\boldsymbol{e}_1\right\rangle_{L^2(\Omega,\mathbb{B})}$

Using (7), it comes:

$$\left\langle \boldsymbol{v}_2, \mathcal{L}\boldsymbol{e}_1\right\rangle_{L^2(\Omega,\mathbb{B})} = \left\langle \mathcal{L}^*\boldsymbol{v}_2, \boldsymbol{e}_1\right\rangle_{L^2(\Omega,\mathbb{A})} + \left\langle \Gamma_\perp\boldsymbol{v}_2, \Gamma_0\boldsymbol{e}_1\right\rangle_{\partial\Omega}.$$

Now the boundary variable $\boldsymbol{e}_\partial := \Gamma_0\boldsymbol{e}_1$ explicitly appears, *i.e.* the causality considered in (11). The weak formulation then reads:

$$\begin{aligned}&\left\langle \boldsymbol{v}_1, \boldsymbol{f}_1\right\rangle_{L^2(\Omega,\mathbb{A})} = -\left\langle \boldsymbol{v}_1, \mathcal{L}^*\boldsymbol{e}_2\right\rangle_{L^2(\Omega,\mathbb{A})},\\ &\left\langle \boldsymbol{v}_2, \boldsymbol{f}_2\right\rangle_{L^2(\Omega,\mathbb{B})} = \left\langle \mathcal{L}^*\boldsymbol{v}_2, \boldsymbol{e}_1\right\rangle_{L^2(\Omega,\mathbb{A})} - \left\langle \Gamma_\perp\boldsymbol{v}_2, \boldsymbol{e}_\partial\right\rangle_{\partial\Omega},\\ &\left\langle \boldsymbol{v}_\partial, \boldsymbol{f}_\partial\right\rangle_{\partial\Omega} = \left\langle \boldsymbol{v}_\partial, \Gamma_\perp\boldsymbol{e}_2\right\rangle_{\partial\Omega}.\end{aligned} \tag{26}$$

Plugging the approximations (23) into (26), this time with $\boldsymbol{\varphi}_1^i \in L^2\left(\Omega, \mathbb{A}\right)$, $\boldsymbol{\varphi}_2^i \in H^{-\mathcal{L}^*}$, and $\boldsymbol{\psi}_\partial^i \in L^2\left(\partial\Omega, \mathbb{R}^m\right)$, gives the following kernel representation of a finite-dimensional Dirac structure:

$$\begin{bmatrix} \mathbf{M}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M}_\partial \end{bmatrix} \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{D}_{-\mathcal{L}^*} & \mathbf{0} \\ -\mathbf{D}_{-\mathcal{L}^*}^{\mathrm{T}} & \mathbf{0} & \mathbf{B}_0 \\ \mathbf{0} & -\mathbf{B}_0^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_\partial \end{pmatrix}, \tag{27}$$

where the matrices $\mathbf{D}_{-\mathcal{L}^*}$ and $\mathbf{B}_0$ are defined by:

$$D_{-\mathcal{L}^*}^{im} = \left\langle \boldsymbol{\varphi}_1^i, -\mathcal{L}^* \boldsymbol{\varphi}_2^m \right\rangle_{L^2(\Omega,\mathbb{A})}, \quad B_0^{mk} = \left\langle \Gamma_\perp \boldsymbol{\varphi}_2^m, \boldsymbol{\psi}_\partial^k \right\rangle_{\partial\Omega}. \tag{28}$$

The power balances proven above still hold true with this causality, where the role played by $\boldsymbol{e}_\partial$ and $\boldsymbol{f}_\partial$ have been switched.

In the sequel, this methodology is applied to the wave equation, the Mindlin-Reissner plate model and the heat equation. These models have been chosen to demonstrate the versatility of our methodology. The wave equation is the prototype of linear hyperbolic systems, and the first example treated by PFEM [8]. The Mindlin model combines wave dynamics and plane elastodynamics, and requires the introduction of tensor-valued variables. Finally, the heat equation is the prototype of parabolic systems, and leads to a pHs with intrinsic algebraic constraint, namely, to a pHDAE.

3.2. The Wave Equation

The wave equation is a well-known model, used as the first example of linear hyperbolic systems in many lecture notes and books. This work is no exception to the rule. However, to account for more realistic physics, let us consider the heterogeneous and anisotropic multidimensional wave equation. The equation reads (see [33]):

$$\rho \frac{\partial^2 w}{\partial t^2} = \operatorname{div}\left(\mathcal{T} \operatorname{grad}(w)\right), \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_{\text{end}}], \quad \Omega \subset \mathbb{R}^N, \tag{29}$$

where ρ is the mass density (bounded from above and below), $\mathcal{T}$ is the *tensorial* Young modulus (symmetric and positive definite) and w is the deflection from the equilibrium.

Let us denote $\alpha_p \coloneqq \rho \dfrac{\partial w}{\partial t}$ the linear momentum and $\boldsymbol{\alpha}_q \coloneqq \operatorname{grad}(w)$ the strain, as energy variables. Hence the Hamiltonian is given as the total energy (summing kinetic and potential energies) by:

$$H = \frac{1}{2} \int_\Omega \left\{ \frac{\alpha_p^2}{\rho} + \boldsymbol{\alpha}_q \cdot \left(\mathcal{T} \boldsymbol{\alpha}_q\right) \right\} \mathrm{d}\Omega. \tag{30}$$

The co-energy variables are by definition the variational derivatives of H with respect to the energy variables, *i.e.*:

$$e_p \coloneqq \delta_{\alpha_p} H = \frac{\alpha_p}{\rho} = wt, \quad \boldsymbol{e}_q \coloneqq \delta_{\boldsymbol{\alpha}_q} H = \mathcal{T} \boldsymbol{\alpha}_q = \mathcal{T} \operatorname{grad}(w), \tag{31}$$

the velocity and stress. With these notations, Equation (29) rewrites:

$$\frac{\partial}{\partial t} \begin{pmatrix} \alpha_p \\ \boldsymbol{\alpha}_q \end{pmatrix} = \begin{bmatrix} 0 & \operatorname{div} \\ \operatorname{grad} & 0 \end{bmatrix} \begin{pmatrix} e_p \\ \boldsymbol{e}_q \end{pmatrix}, \tag{32}$$

together with the constitutive relations given in (31).

Let us denote:

$$\boldsymbol{e}_1 \coloneqq e_p,\ \boldsymbol{e}_2 \coloneqq \boldsymbol{e}_q,\ \mathcal{M}_1 \coloneqq \rho,\ \mathcal{M}_2 \coloneqq \boldsymbol{\mathcal{T}}^{-1},$$
$$\mathcal{L} \coloneqq \text{grad},\ \Gamma_\perp \coloneqq \gamma_\perp = \gamma_0 \cdot \boldsymbol{n},\ \Gamma_0 \coloneqq \gamma_0,$$

where γ_0 is the Dirichlet trace operator. Then the Hamiltonian (30) rewrites as (17). The wave equation (29) with Neumann boundary control $\boldsymbol{u}_\partial = \gamma_\perp\left(\boldsymbol{e}_q\right)$ is given by (18).

The application of PFEM directly gives:

$$\begin{aligned}
&\begin{bmatrix} \mathbf{M}_\rho & 0 \\ 0 & \mathbf{M}_{\boldsymbol{\mathcal{T}}^{-1}} \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{pmatrix} = \begin{bmatrix} 0 & -\mathbf{D}_{\text{grad}}^{\mathrm{T}} \\ \mathbf{D}_{\text{grad}} & 0 \end{bmatrix}\begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{pmatrix} + \begin{bmatrix} \mathbf{B}_\perp \\ 0 \end{bmatrix}\mathbf{u}_\partial \\
&\mathbf{M}_\partial \mathbf{y}_\partial = \begin{bmatrix} \mathbf{B}_\perp^{\mathrm{T}} & 0 \end{bmatrix}\begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{pmatrix},
\end{aligned} \tag{33}$$

where:

$$M_\rho^{ij} \coloneqq \left\langle \boldsymbol{\varphi}_1^i, \rho\boldsymbol{\varphi}_1^j \right\rangle_{L^2(\Omega,\mathbb{R})},\quad M_{\boldsymbol{\mathcal{T}}^{-1}}^{kl} \coloneqq \left\langle \boldsymbol{\varphi}_2^k, \boldsymbol{\mathcal{T}}^{-1}\boldsymbol{\varphi}_2^l \right\rangle_{L^2\left(\Omega,\mathbb{R}^N\right)},$$

are the discretizations of the operators $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively.

3.3. The Mindlin Plate Model

The Mindlin model is a generalization to the 2D case of the Timoshenko beam model and is expressed by a system of two coupled PDEs (see [34]):

$$\begin{cases} \rho b\dfrac{\partial^2 w}{\partial t^2} = \text{div}\left(\boldsymbol{q}\right) + f, \quad (\boldsymbol{x},t) \in \Omega \times [0, t_{\text{end}}], \quad \Omega \subset \mathbb{R}^2, \\ \dfrac{\rho b^3}{12}\dfrac{\partial^2 \boldsymbol{\theta}}{\partial t^2} = \boldsymbol{q} + \text{Div}\left(\boldsymbol{M}\right) + \boldsymbol{t}, \end{cases} \tag{34}$$

where ρ is the mass density, b the plate thickness, w the vertical displacement, $\boldsymbol{\theta} = \left(\theta_x, \theta_y\right)^{\mathrm{T}}$ collects the deflection of the cross section along axes x and y respectively. The fields $f, \boldsymbol{\tau}$ represent distributed forces and torques. Variables $\boldsymbol{M}, \boldsymbol{q}$ represent the momenta tensor and the shear stress. Hooke's law relates those to the curvature tensor and shear deformation vector:

$$\begin{aligned} \boldsymbol{M} &\coloneqq \boldsymbol{\mathcal{D}}_b \boldsymbol{K} \in \mathbb{S}, \quad & \boldsymbol{K} &\coloneqq \text{Grad}\left(\boldsymbol{\theta}\right) \in \mathbb{S}, \\ \boldsymbol{q} &\coloneqq D_s \boldsymbol{\gamma}, & \boldsymbol{\gamma} &\coloneqq \text{grad}\left(w\right) - \boldsymbol{\theta}. \end{aligned}$$

$D_s \coloneqq \dfrac{E_Y b K_{\text{sh}}}{2(1+\nu)}$ is the shear rigidity coefficient, where E_Y is the Young modulus, ν is the Poisson modulus, K_{sh} is the shear correction factor. Tensor $\boldsymbol{\mathcal{D}}_b$ is the bending stiffness:

$$\boldsymbol{\mathcal{D}}_b\left(\cdot\right) = \frac{E_Y b^3}{12\left(1-\nu^2\right)}\left[\left(1-\nu\right)\left(\cdot\right) + \nu\,\text{Tr}\left(\cdot\right)\right]. \tag{35}$$

An appropriate selection of the energy variables is the following [7] [35]:

$$\alpha_w = \rho b \partial_t w, \quad \boldsymbol{\alpha}_\theta \coloneqq \frac{\rho b^3}{12}\partial_t \boldsymbol{\theta}, \quad \boldsymbol{A}_\kappa \coloneqq \boldsymbol{K}, \quad \boldsymbol{\alpha}_\gamma \coloneqq \boldsymbol{\gamma}. \tag{36}$$

The Hamiltonian H (total energy) is expressed in terms of energy variables as:

$$H = \frac{1}{2}\int_\Omega \left\{ \frac{1}{\rho b}\alpha_w^2 + \frac{12}{\rho b^3}\left\|\boldsymbol{\alpha}_\theta\right\|^2 + \boldsymbol{A}_\kappa : \left(\mathcal{D}_b \boldsymbol{A}_\kappa\right) + D_s \left\|\boldsymbol{\alpha}_\gamma\right\|^2 \right\} \Omega, \tag{37}$$

where $\boldsymbol{A}:\boldsymbol{B}$ denotes the tensor contraction. The co-energy variables are defined as follows:

$$\begin{aligned} e_w &:= \delta_{\alpha_w} H = \partial_t w, \quad \boldsymbol{e}_\theta := \delta_{\boldsymbol{\alpha}_\theta} H = \partial_t \boldsymbol{\theta}, \\ \boldsymbol{E}_\kappa &:= \delta_{\boldsymbol{A}_\kappa} H = \boldsymbol{M}, \quad \boldsymbol{e}_\gamma := \delta_{\boldsymbol{\alpha}_\gamma} H = \boldsymbol{q}. \end{aligned} \tag{38}$$

System (34) is then expressed in port-Hamiltonian form as [7] (forces and torques have been omitted for simplicity):

$$\frac{\partial}{\partial t}\begin{pmatrix} \alpha_w \\ \boldsymbol{\alpha}_\theta \\ \boldsymbol{A}_\kappa \\ \boldsymbol{\alpha}_\gamma \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & \mathrm{div} \\ \boldsymbol{0} & \boldsymbol{0} & \mathrm{Div} & \boldsymbol{I}_{2\times 2} \\ \boldsymbol{0} & \mathrm{Grad} & \boldsymbol{0} & \boldsymbol{0} \\ \mathrm{grad} & -\boldsymbol{I}_{2\times 2} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{pmatrix} e_w \\ \boldsymbol{e}_\theta \\ \boldsymbol{E}_\kappa \\ \boldsymbol{e}_\gamma \end{pmatrix}. \tag{39}$$

By applying the divergence theorem, the energy rate is expressed as the duality product of the boundary variables:

$$\begin{aligned} \frac{\mathrm{d}H}{\mathrm{d}t} &= \left\langle \partial_t \alpha_w, e_w \right\rangle_{L^2(\Omega,\mathbb{R})} + \left\langle \partial_t \boldsymbol{\alpha}_\theta, \boldsymbol{e}_\theta \right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)} \\ &\quad + \left\langle \partial_t \boldsymbol{A}_\kappa, \boldsymbol{E}_\kappa \right\rangle_{L^2(\Omega,\mathbb{S})} + \left\langle \partial_t \boldsymbol{\alpha}_\gamma, \boldsymbol{e}_\gamma \right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)}, \\ &= \left\langle \gamma_0 e_w, \gamma_\perp \boldsymbol{e}_\gamma \right\rangle_{\partial\Omega} + \left\langle \boldsymbol{\gamma}_0 \boldsymbol{e}_\theta, \boldsymbol{\gamma}_\perp \boldsymbol{E}_\kappa \right\rangle_{\partial\Omega} \\ &= \left\langle \boldsymbol{y}_{\partial,1}, \boldsymbol{u}_{\partial,1} \right\rangle_{\partial\Omega} + \left\langle \boldsymbol{y}_{\partial,2}, \boldsymbol{u}_{\partial,2} \right\rangle_{\partial\Omega}, \end{aligned} \tag{40}$$

where:

$$\begin{aligned} \boldsymbol{u}_{\partial,1} &= \gamma_\perp \boldsymbol{e}_\gamma, \quad \boldsymbol{y}_{\partial,1} = \gamma_0 e_w, \\ \boldsymbol{u}_{\partial,2} &= \boldsymbol{\gamma}_\perp \boldsymbol{E}_\kappa, \quad \boldsymbol{y}_{\partial,2} = \boldsymbol{\gamma}_0 \boldsymbol{e}_\theta. \end{aligned}$$

The traces $\boldsymbol{\gamma}_0 \boldsymbol{u} = \boldsymbol{u}\big|_{\partial\Omega}$, $\boldsymbol{\gamma}_\perp \boldsymbol{U} = \boldsymbol{U}\cdot\boldsymbol{n}\big|_{\partial\Omega}$ correspond to the Dirichlet trace for $\mathbb{R}^d$ vectors and to the normal trace for $\mathbb{R}^{d\times d}$ tensors. The mass operators are given by:

$$\mathcal{M}_1 = \begin{bmatrix} \rho b & 0 \\ \boldsymbol{0} & \dfrac{\rho b^3}{12} \end{bmatrix}, \quad \mathcal{M}_2 = \begin{bmatrix} \mathcal{D}_b^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & D_s^{-1} \end{bmatrix}. \tag{41}$$

The $\mathcal{L}$, Γ_0, and $\Gamma_\perp$ operators are:

$$\mathcal{L} = \begin{bmatrix} \boldsymbol{0} & \mathrm{Grad} \\ \mathrm{grad} & -\boldsymbol{I}_{2\times 2} \end{bmatrix}, \quad \Gamma_0 = \begin{bmatrix} \gamma_0 & 0 \\ 0 & \boldsymbol{\gamma}_0 \end{bmatrix}, \quad \Gamma_\perp = \begin{bmatrix} 0 & \gamma_\perp \\ \boldsymbol{\gamma}_\perp & 0 \end{bmatrix}. \tag{42}$$

Introducing the approximations for the test and co-energy variables:

$$\begin{aligned} \Delta_w &= \sum_{i=1}^{N_w} \phi_w^i \Delta_w^i, \quad \Delta_\theta = \sum_{i=1}^{N_\theta} \boldsymbol{\phi}_\theta^i \Delta_\theta^i, \\ \Delta_\kappa &= \sum_{i=1}^{N_\kappa} \boldsymbol{\Phi}_\kappa^i \Delta_\kappa^i, \quad \Delta_\gamma = \sum_{i=1}^{N_\gamma} \boldsymbol{\phi}_\gamma^i \Delta_\gamma^i, \end{aligned} \tag{43}$$

where $\Delta = \{v, e\}$, and for the boundary controls:

$$\begin{aligned}&\square_{\partial,1}=\sum_{i=1}^{N_{\partial,1}}\psi_{\partial,1}^{i}\square_{\partial,1}^{i},\ \square_{\partial,2}=\sum_{i=1}^{N_{\partial,2}}\boldsymbol{\psi}_{\partial,2}^{i}\square_{\partial,2}^{i},\\&\square=\{v,u,y\},\ \psi_{\partial,1}^{i}\in\mathbb{R},\ \boldsymbol{\psi}_{\partial,2}^{i}\in\mathbb{R}^{2},\end{aligned}\tag{44}$$

PFEM can be applied to obtain:

$$\begin{aligned}\operatorname{Diag}\begin{bmatrix}\mathbf{M}_{\rho h}\\ \mathbf{M}_{I_\theta}\\ \mathbf{M}_{\mathcal{D}_b^{-1}}\\ \mathbf{M}_{D_s^{-1}}\end{bmatrix}\begin{pmatrix}\dot{\mathbf{e}}_w\\ \dot{\mathbf{e}}_\theta\\ \dot{\mathbf{e}}_\kappa\\ \dot{\mathbf{e}}_\gamma\end{pmatrix}&=\begin{bmatrix}\mathbf{0}&\mathbf{0}&\mathbf{0}&-\mathbf{D}_{\text{grad}}^{\mathrm{T}}\\ \mathbf{0}&\mathbf{0}&-\mathbf{D}_{\text{Grad}}^{\mathrm{T}}&-\mathbf{D}_0^{\mathrm{T}}\\ \mathbf{0}&\mathbf{D}_{\text{Grad}}&\mathbf{0}&\mathbf{0}\\ \mathbf{D}_{\text{grad}}&\mathbf{D}_0&\mathbf{0}&\mathbf{0}\end{bmatrix}\begin{pmatrix}\mathbf{e}_w\\ \mathbf{e}_\theta\\ \mathbf{e}_\kappa\\ \mathbf{e}_\gamma\end{pmatrix}\\&+\begin{bmatrix}\mathbf{B}_{\perp,w}&\mathbf{0}\\ \mathbf{0}&\mathbf{B}_{\perp,\theta}\\ \mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}\end{bmatrix}\begin{pmatrix}\mathbf{u}_{\partial,1}\\ \mathbf{u}_{\partial,2}\end{pmatrix},\\ \operatorname{Diag}\begin{bmatrix}\mathbf{M}_{\partial,1}\\ \mathbf{M}_{\partial,2}\end{bmatrix}\begin{pmatrix}\mathbf{y}_{\partial,1}\\ \mathbf{y}_{\partial,2}\end{pmatrix}&=\begin{bmatrix}\mathbf{B}_{\perp,w}^{\mathrm{T}}&\mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{B}_{\perp,\theta}^{\mathrm{T}}&\mathbf{0}&\mathbf{0}\end{bmatrix}\begin{pmatrix}\mathbf{e}_w\\ \mathbf{e}_\theta\\ \mathbf{e}_\kappa\\ \mathbf{e}_\gamma\end{pmatrix}.\end{aligned}\tag{45}$$

The notation Diag denotes a block diagonal matrix. The mass matrices $\mathbf{M}_{\rho h},\mathbf{M}_{I_\theta},\mathbf{M}_{\mathcal{C}_b},\mathbf{M}_{C_s}$ are computed as:

$$\begin{aligned}M_{\rho h}^{ij}&=\left\langle\phi_w^i,\rho h\phi_w^j\right\rangle_{L^2(\Omega)},&M_{\mathcal{D}_b^{-1}}^{pq}&=\left\langle\boldsymbol{\Phi}_\kappa^p,\mathcal{D}_b^{-1}\boldsymbol{\Phi}_\kappa^q\right\rangle_{L^2\left(\Omega,\mathbb{R}_{\text{sym}}^{2\times2}\right)},\\M_{I_\theta}^{mn}&=\left\langle\boldsymbol{\phi}_\kappa^m,I_\theta\boldsymbol{\phi}_\kappa^n\right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)},&M_{D_s^{-1}}^{rs}&=\left\langle\boldsymbol{\phi}_\gamma^r,D_s^{-1}\boldsymbol{\phi}_\gamma^s\right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)},\end{aligned}\tag{46}$$

where $i,j\in\{1,N_w\}$, $m,n\in\{1,N_\theta\}$, $p,q\in\{1,N_\kappa\}$, $r,s\in\{1,N_\gamma\}$. Matrices $\mathbf{D}_{\text{grad}},\mathbf{D}_{\text{Grad}},\mathbf{D}_0$ assume the form:

$$\begin{aligned}D_{\text{grad}}^{rj}&=\left\langle\boldsymbol{\phi}_\gamma^r,\text{grad}\phi_w^j\right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)},\\D_{\text{Grad}}^{pn}&=\left\langle\boldsymbol{\Phi}_\kappa^p,\text{Grad}\boldsymbol{\phi}_\theta^n\right\rangle_{L^2\left(\Omega,\mathbb{R}_{\text{sym}}^{2\times2}\right)},\end{aligned}\qquad D_0^{rn}=-\left\langle\boldsymbol{\phi}_\gamma^r,\boldsymbol{\phi}_\theta^n\right\rangle_{L^2\left(\Omega,\mathbb{R}^2\right)}.\tag{47}$$

Matrices $\mathbf{B}_w,\mathbf{B}_\theta,\mathbf{M}_{\partial,1},\mathbf{M}_{\partial,2}$ are computed as:

$$\begin{aligned}\mathbf{B}_{\perp,w}^{ig}&=\left\langle\gamma_0\phi_w^i,\psi_{\partial,1}^g\right\rangle_{\partial\Omega},&M_{\partial,1}^{fg}&=\left\langle\psi_{\partial,1}^f,\psi_{\partial,1}^g\right\rangle_{L^2\left(\partial\Omega,\mathbb{R}^m\right)},&f,g\in\{1,N_{\partial,1}\},\\\mathbf{B}_{\perp,\theta}^{mg}&=\left\langle\boldsymbol{\gamma}_0\boldsymbol{\phi}_\theta^m,\boldsymbol{\psi}_{\partial,2}^g\right\rangle_{\partial\Omega},&M_{\partial}^{lk}&=\left\langle\boldsymbol{\psi}_{\partial,2}^l,\boldsymbol{\psi}_{\partial,2}^k\right\rangle_{L^2\left(\partial\Omega,\mathbb{R}^m\right)},&l,k\in\{1,N_{\partial,2}\}\end{aligned}\tag{48}$$

The discrete Hamiltonian is then computed as:

$$H_d=\frac{1}{2}\mathbf{e}_w^{\mathrm{T}}\mathbf{M}_{\rho h}\mathbf{e}_w+\frac{1}{2}\mathbf{e}_\theta^{\mathrm{T}}\mathbf{M}_{I_\theta}\mathbf{e}_\theta+\frac{1}{2}\mathbf{e}_\kappa^{\mathrm{T}}\mathbf{M}_{\mathcal{D}_b^{-1}}\mathbf{e}_\kappa+\frac{1}{2}\mathbf{e}_\gamma^{\mathrm{T}}\mathbf{M}_{D_s^{-1}}\mathbf{e}_\gamma.\tag{49}$$

From system (45) the discrete energy rate is readily obtained:

$$\frac{\mathrm{d}H_d}{\mathrm{d}t}=\mathbf{y}_{\partial,1}^{\mathrm{T}}\mathbf{M}_{\partial,1}\mathbf{u}_{\partial,1}+\mathbf{y}_{\partial,2}^{\mathrm{T}}\mathbf{M}_{\partial,2}\mathbf{u}_{\partial,2}.\tag{50}$$

The discrete energy rate then mimics its infinite-dimensional counterpart.

Remark 10. *Equivalently a purely mixed formulation can be obtained by integrating by parts the third and fourth lines of* (39)*. In this case, the system of*

equations gathers together a plane elasticity problem [36] *and a wave equation in mixed form. Conforming finite elements for the plane elasticity system on simplicial meshes have been constructed in* [37]. *The simpler PEERS elements based on a weak symmetry formulation have been proposed in* [38]. *The PEERS elements have been used in* [39] *to construct a stable locking-free mixed formulation for the static Mindlin problem.*

3.4. The Heat Equation

The heat equation is the simplest example of parabolic system. Instead of rewriting the well-known PDE as a pHs, a direct pHs modelling is presented, as done in [9] [10]. The model is constructed in order to keep apart thermodynamical principles from equations of state. Indeed, the pHs formalism allows to modify the latter, by keeping the structure of the former.

Let $\Omega \subset \mathbb{R}^N$ be a bounded open connected set. Assume that this domain models a rigid body: its volume does not change over time and no chemical reaction is to be found. Let us denote: ρ the mass density, u the internal energy density, $\boldsymbol{J}_Q$ the heat flux, T the local temperature, $\beta := \frac{1}{T}$ the reciprocal temperature, s the entropy density, $\boldsymbol{J}_S := \beta \boldsymbol{J}_Q$ the entropy flux, $C_V := \left(\frac{\mathrm{d}u}{\mathrm{d}T}\right)_V$ the isochoric heat capacity.

The first law of thermodynamic reads:

$$\rho \frac{\partial u}{\partial t} = -\mathrm{div}\left(\boldsymbol{J}_Q\right). \tag{51}$$

Under the hypothesis of an inert rigid solid, Gibbs formula reads $\mathrm{d}u = T\mathrm{d}s$, giving:

$$\frac{\partial u}{\partial t} = T \frac{\partial s}{\partial t}. \tag{52}$$

Defining $\sigma := \mathrm{grad}\left(\beta\right)\boldsymbol{J}_Q$, and seeing u as a function of the entropy density s, Gibbs formula (52) gives:

$$\rho \frac{\partial s}{\partial t} = -\mathrm{div}\left(\boldsymbol{J}_S\right) + \sigma. \tag{53}$$

Then σ is the irreversible entropy production.

In this work, the following constitutive equations of state will be assumed:

- The rigid body is at room temperature: the Dulong-Petit model is supposed to be satisfied, *i.e.* $u = C_V T$, with time-invariant C_V;
- The thermal conduction is given by Fourier's law, with a symmetric positive tensor $\boldsymbol{\lambda}$: $\boldsymbol{J}_Q = -\boldsymbol{\lambda}\mathrm{grad}\left(T\right)$.

Thanks to (51) and the equations of state, we easily recover the classical PDE for the temperature T: $\rho C_V \frac{\partial T}{\partial t} = \mathrm{div}\left(\boldsymbol{\lambda}\mathrm{grad}\left(T\right)\right)$.

The "L^2-energy" $\left(\int_\Omega \rho C_V T^2 \mathrm{d}\Omega\right)^{\frac{1}{2}}$ is commonly used as Hamiltonian for the

heat equation. However, it lacks of a thermodynamical meaning. The internal energy would be more accurate for this physical problem, even though it will rise some difficulties. Nevertheless, the pHs formalism allows dealing with it, and PFEM proves to be powerful enough to discretize the system in a structure-preserving manner even for this choice of Hamiltonian.

Let the internal energy be seen as a functional of the local entropy as energy variable: $\alpha_s := \rho s$, then:

$$H := \int_\Omega \rho u(\alpha_s)\mathrm{d}\Omega.$$

The co-energy variable is given by $e_s := \delta_{\alpha_s} H = \dfrac{\mathrm{d}\rho u}{\mathrm{d}\rho s} = T$, the local temperature. Denoting $\boldsymbol{e}_S := \boldsymbol{J}_S$, one can introduce a new flow variable $\boldsymbol{f}_S$ such that:

$$\begin{pmatrix} \dfrac{\partial \alpha_s}{\partial t} \\ \boldsymbol{f}_S \end{pmatrix} = \begin{bmatrix} 0 & -\mathrm{div} \\ -\mathrm{grad} & 0 \end{bmatrix} \begin{pmatrix} e_s \\ \boldsymbol{e}_S \end{pmatrix} + \begin{pmatrix} \sigma \\ 0 \end{pmatrix}.$$

Obviously, $\boldsymbol{f}_S = -\mathrm{grad}(e_s)$. In order to get a formally skew-symmetric operator, let us also introduce an *entropy* port (f_σ, e_σ), such that $e_\sigma = -\sigma$. Then:

$$\begin{pmatrix} \dfrac{\partial \alpha_s}{\partial t} \\ \boldsymbol{f}_S \\ f_\sigma \end{pmatrix} = \begin{bmatrix} 0 & -\mathrm{div} & -1 \\ -\mathrm{grad} & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{pmatrix} e_s \\ \boldsymbol{e}_S \\ e_\sigma \end{pmatrix}. \tag{54}$$

Remark 11. *As surprising as it can be, in this setting, Fourier's law appears to be stated in a nonlinear way*: $e_s \boldsymbol{e}_S - \boldsymbol{\lambda} \boldsymbol{f}_S = 0$. *This comes from the necessity to express the constitutive relations in function of the flows and efforts appearing in the equation defining the Stokes-Dirac structure.*

Remark 12. *Two variables have been added to obtain* (54), *but only one equation naturally appears*: $f_\sigma = e_s$. *Thus, another equation is needed to close the system*: $\mathcal{G}(\boldsymbol{f}_2, \boldsymbol{e}_2) = 0$. *Here, it is given by the definition of the irreversible entropy production* $\sigma := \mathrm{grad}(\beta)\boldsymbol{J}_Q$, *rewritten in the flows and efforts variables. This leads to the nonlinear constitutive relation*: $\boldsymbol{f}_S \boldsymbol{e}_S + f_\sigma e_\sigma = 0$.

Remark 13. *Usually the system energy is taken to be* $\left(\int_\Omega \rho C_V T^2 \mathrm{d}\Omega\right)^{\frac{1}{2}}$, *that gives rise to the well-known linear diffusive system. At first glance, our approach may be surprising since it leads to a lossless nonlinear differential-algebraic system. There is indeed a major advantage in doing so, in view of the modelling and discretization of complex systems by interconnection of several pHs. For instance, if one wants to interconnect both thermal and mechanical processes, for the energy exchanges to be consistent, physics must be coherent. When dissipation occurs, through* e.g. *friction or viscosity, the kinetic energy is converted into internal energy. Hence, for a physically meaningful system, the internal energy is indeed the one to consider.*

Let us define:

$$\boldsymbol{f}_1 \coloneqq \frac{\partial \alpha_s}{\partial t}, \quad \boldsymbol{f}_2 \coloneqq \begin{pmatrix} \boldsymbol{f}_S \\ f_\sigma \end{pmatrix}, \quad \boldsymbol{e}_1 \coloneqq e_s, \quad \boldsymbol{e}_2 \coloneqq \begin{pmatrix} \boldsymbol{e}_S \\ e_\sigma \end{pmatrix},$$

and:

$$\mathcal{L} \coloneqq \begin{pmatrix} -\mathrm{grad} \\ 1 \end{pmatrix}, \quad \Gamma_\perp \coloneqq \begin{pmatrix} -\gamma_\perp & 0 \end{pmatrix}, \quad \Gamma_0 \coloneqq \gamma_0.$$

Then, the heat Equation (54) with boundary control $\boldsymbol{e}_\partial \coloneqq \boldsymbol{u}_\partial = \Gamma_0 \boldsymbol{e}_1 = \gamma_0(e_s)$ and boundary observation $\boldsymbol{f}_\partial \coloneqq -\boldsymbol{y}_\partial = \Gamma_\perp \boldsymbol{e}_2 = \gamma_\perp(\boldsymbol{e}_S)$ rewrites under the form (11). Thus PFEM will be applied with an *integration by parts* on the second line in this strategy, leading to (27), which rewrites with the current variables:

$$\mathrm{Diag}\begin{bmatrix} \mathbf{M}_s \\ \mathbf{M}_S \\ \mathbf{M}_\sigma \\ \mathbf{M}_\partial \end{bmatrix}\begin{pmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\underline{\alpha}_s \\ \mathbf{f}_S \\ \mathbf{f}_\sigma \\ -\mathbf{y}_\partial \end{pmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{D}_{-\mathrm{div}} & -\mathbf{M}_\sigma & \mathbf{0} \\ -\mathbf{D}_{-\mathrm{div}}^{\mathrm{T}} & \mathbf{0} & \mathbf{0} & \mathbf{B}_0 \\ \mathbf{M}_\sigma & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{B}_0^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \end{bmatrix}\begin{pmatrix} \mathbf{e}_s \\ \mathbf{e}_S \\ \mathbf{e}_\sigma \\ \mathbf{u}_\partial \end{pmatrix}, \tag{55}$$

where:

$$M_s^{ij} \coloneqq \left\langle \boldsymbol{\varphi}_1^i, \boldsymbol{\varphi}_1^j \right\rangle_{L^2(\partial\Omega,\mathbb{R})}, \qquad M_S^{kl} \coloneqq \left\langle \boldsymbol{\varphi}_S^k, \boldsymbol{\varphi}_S^l \right\rangle_{L^2\left(\Omega,\mathbb{R}^N\right)},$$

$$M_\sigma^{\tilde{k}\tilde{l}} \coloneqq \left\langle \boldsymbol{\varphi}_\sigma^{\tilde{k}}, \boldsymbol{\varphi}_\sigma^{\tilde{l}} \right\rangle_{L^2\left(\Omega,\mathbb{R}^N\right)}, \quad B_0^{km} \coloneqq -\left\langle \Gamma_\perp \boldsymbol{\varphi}_S^k, \boldsymbol{\psi}_\partial^m \right\rangle_{L^2(\partial\Omega,\mathbb{R})},$$

with $\boldsymbol{\varphi}_2^k \coloneqq \begin{pmatrix} \boldsymbol{\varphi}_S^k \\ \mathbf{0} \end{pmatrix}$ if $1 \le k \le N_S$, and $\boldsymbol{\varphi}_2^k \coloneqq \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\varphi}_\sigma^{k-N_S} \end{pmatrix}$ if $N_S + 1 \le k \le N_S + N_\sigma$.

To be compatible, the discretizations of the constitutive relations are given as follows:

- Dulong-Petit model reads: $\mathbf{M}_s \underline{\alpha}_s = \mathbf{M}_{\rho C_V} \mathbf{e}_s$, with $M_{\rho C_V}^{ij} \coloneqq \left\langle \boldsymbol{\varphi}_1^i, \rho C_V \boldsymbol{\varphi}_1^j \right\rangle_{L^2(\Omega,\mathbb{R})}$;
- Following Remark 11, Fourier's law reads: $\boldsymbol{\Lambda}\mathbf{f}_S = \mathbf{M}_{e_s}\mathbf{e}_S$ with $\boldsymbol{\Lambda}^{ij} \coloneqq \left\langle \boldsymbol{\varphi}_S^i, \boldsymbol{\lambda}\boldsymbol{\varphi}_S^j \right\rangle_{L^2\left(\Omega,\mathbb{R}^N\right)}$ and $M_{e_s}^{ij} \coloneqq \left\langle \boldsymbol{\varphi}_S^i, e_s \boldsymbol{\varphi}_S^j \right\rangle_{L^2\left(\Omega,\mathbb{R}^N\right)}$;
- The constitutive law coming from the introduction of the irreversible entropy production, as explained in Remark 12, is taken into account by:

$$\left(\mathbf{e}_S\right)^{\mathrm{T}} \mathbf{M}_S \mathbf{f}_S + \left(\mathbf{e}_\sigma\right)^{\mathrm{T}} \mathbf{M}_\sigma \mathbf{f}_\sigma = 0. \tag{56}$$

Remark 14. *In Fourier's law, the mass matrix* $\mathbf{M}_{e_s}$ *depends on the co-energy variable* e_s. *This will rise difficulties for the numerical solution in time.*

To conclude, the structure-preserving property can be appreciated in the following result.

Proposition 3. *Let* $H^d(\underline{\alpha}_s) \coloneqq H(\alpha_s^d)$ *be the discrete Hamiltonian, where* α_s^d *is the discretization of the energy variable in the basis* $\boldsymbol{\varphi}_1$. *It holds*:

$$\frac{\mathrm{d}}{\mathrm{d}t} H^d = \mathbf{u}_\partial^{\mathrm{T}} \mathbf{M}_\partial \mathbf{y}_\partial,$$

that is the first law of thermodynamics at the discrete level.

Proof 3. *Thanks to the compatible discretization of the Dulong-Petit model,*

Proposition 2 *gives*:

$$\frac{\mathrm{d}}{\mathrm{d}t}H^d = -\mathbf{e}_2^{\mathrm{T}}\mathbf{M}_2\mathbf{f}_2 - \mathbf{e}_\partial^{\mathrm{T}}\mathbf{M}_\partial\mathbf{f}_\partial.$$

By definition of $\mathbf{f}_2$, $\mathbf{e}_2$, $\mathbf{M}_2$, $\mathbf{e}_\partial$, and $\mathbf{f}_\partial$ one computes:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}H^d &= -\mathbf{e}_2^{\mathrm{T}}\mathbf{M}_2\mathbf{f}_2 - \mathbf{e}_\partial^{\mathrm{T}}\mathbf{M}_\partial\mathbf{f}_\partial,\\ &= -\begin{pmatrix}\mathbf{e}_S\\ \mathbf{e}_\sigma\end{pmatrix}^{\mathrm{T}}\begin{bmatrix}\mathbf{M}_S & \mathbf{0}\\ \mathbf{0} & \mathbf{M}_\sigma\end{bmatrix}\begin{pmatrix}\mathbf{f}_S\\ \mathbf{f}_\sigma\end{pmatrix} + \mathbf{u}_\partial^{\mathrm{T}}\mathbf{M}_\partial\mathbf{y}_\partial,\\ &= \mathbf{u}_\partial^{\mathrm{T}}\mathbf{M}_\partial\mathbf{y}_\partial,\end{aligned}$$

thanks to the constitutive relation (56) coming from the irreversible entropy production.

Remark 15. *Fourier's law does not contribute to the power balance of the internal energy. Nevertheless, such a constitutive relation is needed for the problem to be well-defined.*

Remark 16. *The methodology detailed so far is certainly not limited to the previous three examples. Indeed higher-order differential* [40], *curl operator for Maxwell's equations* [32], *nonlinear system* [41], *and different Hamiltonian choices can be handled as well. For instance, in the case of the heat equation, the entropy or the classical L^2-norm of the temperature can be alternatively considered as Hamiltonian functional* [9] [10]. *In addition, mixed boundary conditions can be incorporated either by introducing Lagrange multipliers or by employing a virtual domain decomposition method* [42]. *As already mentioned, dissipation* (*both in the domain and on the boundary*) *can also be considered in this strategy* [30] [31]. *Hence a very wide class of* (*nonlinear*) *multiphysics systems can be discretized in a structure-preserving manner* (*with well-represented exchanges of energy between the subsystems*).

In the next section we present an ongoing project which has been initiated to prove the efficiency of the PFEM methodology, leveraging well-established and robust software tools for the finite element discretization of partial differential equations and time integration.

4. SCRIMP: Simulation and Control of Interactions in Multi-Physics

In this section the main features related to the numerical simulation of pHs in the framework of the ongoing project named SCRIMP (Simulation and Control of Interactions in Multi-Physics) are detailed. The aim is to provide a flexible prototype Python code for the numerical simulation of pHs both for research and educational purposes. In addition to numerical experiments proposed later in Section 5, the reader is referred to interactive companion Jupyter notebooks [6] to learn how to numerically solve the model problems introduced in Section 3 with SCRIMP. In the following, the key ideas behind SCRIMP are mentioned and then a specific emphasis on both space and time discretizations is given.

4.1. Key Ideas Behind SCRIMP

In short, the key ideas related to the design of SCRIMP are provided:

- The Python dynamic programming language has been selected due to its expressiveness and the availability of high-level interfaces to scientific computing software libraries [43];
- SCRIMP assumes to rely on open-source, external software for the finite-dimensional discretization of partial differential equations;
- SCRIMP encapsulates the finite-dimensional objects related to the finite element discretization in space (e.g. matrices) to deduce the resulting linear or nonlinear pHs in a generic pHODE/pHDAE form as proposed in [26];
- For multiphysics problems, this design offers the advantage that discretization in space may be handled by different software components depending on the discipline or on the modelling. The modularity and the object-oriented nature of Python thus offer the flexibility to easily combine the different pHs to deduce the global interconnected system. This is much in line with the mathematical theory of pHs [24]. Furthermore we note that interconnections of different systems (with e.g. the transformer or gyrator transformations [24]) can be easily incorporated.

The design of SCRIMP is based on procedural and object-oriented paradigms and thus follows the standard ideas governing most of the numerical PDE software. Whereas a detailed exposition of the design patterns of SCRIMP and its performance will be published elsewhere, concrete illustrations of most of these key ideas can be found in the companion Jupyter notebooks [6]. The description of the current numerical methods related to space and time discretizations available in SCRIMP is given.

4.2. Semi-Discretization in Space

As outlined in Section 3, PFEM relies on an abstract variational formulation written in appropriate finite element spaces.

To perform the semi-discretization in space, we rely on FEniCS [23], an open-source C++ scientific software library that provides a high-level Python interface. The FEniCS Project is mainly based on a collection of software components targeting the automated solution of partial differential equations via the finite element method. Its core components notably include the Unified Form Language (UFL) [44], the FEniCS Form Compiler (FFC) [45] and the finite element library DOLFIN [46], which contains various types of conforming finite element methods, e.g., nodal Lagrangian finite elements for grad-conforming approximations or non non-nodal finite elements (e.g., Raviart-Thomas spaces for div-conforming approximations) as well. These families of finite elements are notably required to tackle the discretization in space of our core problems.

A key point to facilitate the generic implementation of PFEM is the use of UFL. UFL is indeed an expressive domain-specific language for abstractly representing (finite element) variational formulations of differential equations. In par-

ticular, this language defines a syntax for the integration of variational forms over various domains. This simply leads to an expressive implementation that is close to the abstract mathematical formulations presented in Section 3. The FEniCS Form Compiler FFC then generates specialized C++ code from the symbolic UFL representation of variational forms and finite element spaces. The combination of these core elements makes FEniCS a versatile and efficient software for the finite element approximation of partial differential equations as outlined in [47]. Additionally, FEniCS also provides an interface for state-of-the-art linear solvers and preconditioners from freely available third-party libraries such as PETSc [48]. This last feature may be especially useful to handle the numerical simulation of large-scale pHs.

4.3. Time Integration Methods

As outlined in Section 3, the semi-discretization in space of the resulting pHs leads to systems of either ordinary differential equations (ODE) or differential algebraic equations (DAE). Hence reliable and accurate time integration methods must be provided.

To offer a large panel of numerical methods, a high-level interface to well-established time integration libraries is provided in SCRIMP. Concerning the numerical solution of ODEs, we provide light interfaces to the Assimulo library [49] and to the SciPy time integration method scipy.integrate.solve_ivp[1] that both include standard multistep and one-step methods for stiff and non-stiff ordinary differential equations given in explicit form $y' = f(t, y)$ with $y(t_0) = y_0$ where t_0 and y_0 denote the initial time and initial condition, respectively. This formulation requires the solution of linear systems of equations involving sparse finite element mass matrices. State-of-the-art sparse direct solvers based on Gaussian factorization are used for that purpose. Through Assimulo, the popular CVODE[2] solver from Sundials [50] is also accessible. For nonstiff problems, CVODE relies on the Adams-Moulton formulas, with the order varying between 1 and 12. For stiff problems, CVODE includes schemes based on Backward Differentiation Formulas (BDFs) with order varying between 1 and 5. In addition, we also propose standalone implementations of symplectic time integration methods such as the second order accurate Stormer-Verlet method.

The interface to Assimulo also allows one to handle the numerical solution of linear DAEs through the use of the Sundials IDA solver[3]. IDA is a package for the solution of differential algebraic equation systems written in the form $F(t, y, y') = 0$ with $y(t_0) = y_0$. The integration method in IDA is based on variable-order, variable-coefficient BDF in fixed-leading-coefficient form, where the method order varies between 1 and 5. We note that setting the initial conditions properly is of utmost importance for a DAE solver. To do so, we rely on

[1]https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html#scipy.integrate.solve_ivp
[2]https://computing.llnl.gov/sites/default/files/cv_guide-5.7.0.pdf
[3]https://computing.llnl.gov/projects/sundials/ida

the IDA_YA_YDP_INIT method to find consistent initial conditions for the time integration. We refer the reader to ([50], Section 2.3] for additional details on IDA. As standalone methods, we have considered the second-order accurate Stormer-Verlet and fourth-order accurate Runge-Kutta (RK4) methods for the solution of linear DAEs.

To the best of our knowledge, open-source libraries for the solution of general nonlinear differential algebraic equations with high-level Python interfaces are not yet available. Hence a simple forward in time integration method for the solution of the nonlinear pHDAE related to the energy formulation of the heat equation problem has been provided; see [51] for illustrations and discussion. As a future direction, we plan to investigate the potential of the PETSc's time stepping library TS [52] to be able to tackle the solution of large-scale pHDAE systems.

4.4. Model Reduction of Port-Hamiltonian Systems

Structure-preserving model reduction is of significant importance for stability analysis, optimization or control of problems related to pHs. Hence structure-preserving model reduction algorithms have been implemented in SCRIMP. In particular, the structure-preserving model reduction algorithm (Algorithm 1) proposed in [53] has been selected in the pHODE case. We refer the reader to [6] for an illustration, where the model reduction of the pHs related to the wave equation problem is considered. While for linear pHDAE systems consolidated methodologies have been proposed (see, e.g., [54]), structure-preserving model reduction for general nonlinear differential algebraic systems remains to be explored, to the best of our knowledge. This is a significant research direction to be considered within SCRIMP in a near future.

5. Numerical Simulations

In this section, PFEM is applied to the pHs presented in Section 3. We specifically learn how to define and solve those problems with SCRIMP. These tutorials introduce the methodology step-by-step and are supposed to be self-contained and independent from the others. We refer the reader to the companion Jupyter notebooks [6] for additional information.

5.1. Anisotropic Heterogeneous Wave Equation

We first recall the continuous problem related to the anisotropic heterogeneous wave equation, enriched with internal and boundary damping, and tackle the semi-discretization in space of the port-Hamiltonian system through the PFEM methodology. This discretization leads to a pHODE formulation as explained in Section 3.2. After time discretization, we perform a numerical simulation to obtain an approximation of the space-time solution.

5.1.1. Problem Statement

We consider the two-dimensional heterogeneous anisotropic wave equation

with impedance boundary condition defined for all ($t \geq 0$) as:

$$\rho(\boldsymbol{x})\frac{\partial^2}{\partial t^2}w(t,\boldsymbol{x}) - \operatorname{div}\left(\mathcal{T}(\boldsymbol{x})\cdot\mathbf{grad}\,w(t,\boldsymbol{x})\right) = -\varepsilon(\boldsymbol{x})\partial_t w(t,\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega,$$

$$Z(\boldsymbol{x})\left(\mathcal{T}(\boldsymbol{x})\cdot\mathbf{grad}\,w(t,\boldsymbol{x})\right)\cdot\mathbf{n} + \partial_t w(t,\boldsymbol{x}) = 0, \quad \boldsymbol{x}\in\partial\Omega,$$

$$w(0,\boldsymbol{x}) = w_0(x), \quad \boldsymbol{x}\in\Omega, t=0$$

$$\partial_t w(0,\boldsymbol{x}) = w_1(x), \quad \boldsymbol{x}\in\Omega, t=0,$$

with $\Omega\subset\mathbb{R}^2$ an open bounded spatial domain with Lipschitz-continuous boundary $\partial\Omega$. We consider here a rectangular shaped domain for Ω. $w(t,\boldsymbol{x})$ denotes the deflection from the equilibrium position at point $\boldsymbol{x}\in\Omega$ and time *t*. $\rho\in C^\infty(\Omega)$ (positive and bounded from below) denotes the mass density, $\mathcal{T}\in C^\infty(\Omega)^{2\times 2}$ (symmetric and coercive) the Young's elasticity modulus, ε a positive viscous damping parameter and $Z(\boldsymbol{x})$ is the positive impedance function defined on $\partial\Omega$, respectively.

5.1.2. Setup

We initialize here the Python object related to the Wave_2D class of SCRIMP. This object will be used throughout this section.

```
W = SCRIMP.Wave_2D()
```

5.1.3. Constants

We define the constants related to the rectangular domain Ω. The coordinates of the bottom left (x_0, y_0) and top right (x_L, y_L) corners of the rectangle are required.

```
x0, xL, y0, yL = 0., 2., 0., 1.
```

We then define the time interval related to the time discretization. t_i, t_f denote the initial and final time instants respectively.

```
ti, tf  = 0., 5.
```

We specify that we choose the Assimulo external library to be used later for the time integration of the resulting ODE and provide the value of the time step. This should be considered as a reference value since adaptative methods in time can be used later.

```
dt           = 1.e-3
ode_library  = 'ODE:Assimulo'
```

5.1.4. FEniCS Expressions Definition

For the finite element discretization of the pHs, the FEniCS library is used in the Wave_2D class of SCRIMP. Hence to properly use FEniCS expression definition, we provide the definition of the different variables in C++ code given in strings. We first specify the mass density as a function depending on the space coordinates. Hence in this expression, $x[0]$ corresponds to the first spatial variable

and $x[1]$ to the second one, respectively.

```
Rho    = 'x[0]*x[0] * (2.-x[0])+ 1'
```

We specify the Young's elasticity modulus tensor. Three components are only required due to the symmetry property of this tensor.

```
T11    = 'x[0]*x[0]+1'
T12    = 'x[1]'
T22    = 'x[0]+2'
```

We finally set the impedance function Z defined on the boundary of the domain. Here a constant value is used on $\partial\Omega$. We also provide the viscous damping parameter (eps).

```
Z   = '0.1'
eps = '25 * x[0] * (xL-x[0]) * x[1] * (yL-x[1])'
```

Finally we specify the initial conditions of the problem related to the energy variables and to the deflection.

```
Aq_0_1= '0'
Aq_0_2= '0'
Ap_0  = '0'
W_0   = '0'
```

5.1.5. Problem at the Continuous Level

We are now able to completely define the problem at the continuous level. We start by specifying that the computational domain Ω is of rectangular shape. To do so, we provide the coordinates of the bottom left and top right corners to the Wave_2D object.

```
W.Set_Rectangular_Domain(x0, xL, y0, yL);
```

Remark 17. *General* Gmsh *meshes can be imported by the user. However, for the time being, the library does not allow the treatment of mixed boundary conditions on generic meshes.*

We provide next the time integration interval.

```
W.Set_Initial_Final_Time(ti, tf);
```

We then provide the physical parameters related to the wave equation: the mass density, the Young's elasticity modulus tensor and the impendance function, respectively.

```
W.Set_Physical_Parameters(Rho, T11, T12, T22);
```

We then specify the complete modelling for the damping and thus provide information related to the impedance function and viscous damping parameter, respectively.

```
W.Set_Damping(damp=['impedance', 'fluid'], Z=Z, eps=eps);
```

The user has to provide the temporal and spatial parts of the boundary control function (Ub_tm0 and Ub_sp0, respectively).

```
W.Set_Boundary_Control(Ub_tm0=lambda t:  np.sin( 2 * 2*pi/tf␣
↪*t) * 25 , Ub_sp0='x[0] * x[1] * (1-x[1])');
```

Finally we provide the initial conditions for the ODE.

```
W.Set_Initial_Data(Aq_0_1='0', Aq_0_2='0', Ap_0='0',␣
↪W_0='0');
```

5.1.6. Problem at the Discrete Level in Space and Time

We start by selecting the computational mesh which is generated with Gmsh[4] and saved as a.xml file. Here the parameter *rfn_num* corresponds to a mesh refinement parameter.

```
W.Set_Gmsh_Mesh('rectangle.xml', rfn_num=2);
```

To perform the discretization in space, we must first specify the conforming finite element approximation spaces to be used (see [20]). Concerning the energy variables associated with the strain, we select the Raviart-Thomas finite element family known as RT_k consisting of vector functions with a continuous normal component across the interfaces between the elements of a mesh. For the energy variables associated with the linear momentum and the boundary variables, we choose the classical P_k finite element approximation. The given combination of parameters rt_order=0, p_order=1, b_order=1 corresponds to the $RT_0P_1P_1$ family.

```
W.Set_Finite_Element_Spaces(family_q='RT', family_p='P',␣
↪family_b='P',rq=0, rp=1, rb=1);
```

We then perform the semi-discretization in space of the weak formulation with PFEM. At the end of this stage, the complete formulation of the pHODE is obtained. The different matrices related to the pHODE system are constructed in the Assembly method of the Wave_2D class of SCRIMP and are directly accessible through the object of the Wave_2D class. The finite element assembly relies on the variational formulation of PFEM and exploits the level of abstraction provided by the UFL used in FEniCS, leading to a code that is close to the mathematical formulation. The divergence based formulation is selected leading to a pHODE system. In other words, the integration by parts will be performed on the second line of (32).

```
W.Assembly(formulation='Div');
```

To perform the time integration of the pHODE, we first need to interpolate both the control function on the boundary and the initial data on the appropriate finite element spaces.

```
W.Project_Boundary_Control()
W.Project_Initial_Data();
```

[4]https://gmsh.info/

Then we specify the parameters related to the time discretization.

```
W.Set_Time_Setting(dt);
```

5.1.7. Numerical Approximation of the Space-Time Solution

We are now able to perform the time integration of the resulting pHODE system and deduce the behaviour of both the energy variables and the Hamiltonian with respect to the time and space variables, respectively. Detailed information from the Assimulo library is included after time integration.

```
A, Hamiltonian = W.Time_Integration(ode_library)
```

```
ODE Integration using assimulo built-in functions:
Final Run Statistics: ---

 Number of steps                                : 614
 Number of function evaluations                 : 800
 Number of Jacobian vector evaluations          : 2977
 Number of function eval. due to Jacobian eval. : 0
 Number of error test failures                  : 0
 Number of nonlinear iterations                 : 797
 Number of nonlinear convergence failures       : 51

Solver options:

 Solver                   : CVode
 Linear multistep method  : BDF
 Nonlinear solver         : Newton
 Linear solver type       : SPGMR
 Maximal order            : 3
 Tolerances (absolute)    : 1e-05
 Tolerances (relative)    : 1e-05

Simulation interval    : 0.0 - 5.0 seconds.
Elapsed simulation time: 0.9727537930002654 seconds.
```

5.1.8. Post-Processing

We represent the two-dimensional mesh with corresponding degrees of freedom for each variable in **Figure 1**.

```
W.notebook = True
W.Plot_Mesh_with_DOFs()
```

We plot the Hamiltonian function versus time in **Figure 2**. Here *tspan* represents the collection of discrete times due to the possibly adaptative time procedure used in the time integration library.

```
W.Plot_Hamiltonian(W.tspan,Hamiltonian, marker='o')
```

The behaviour of the deflection is graphically represented at a given time. Here we simply plot the deflection at the final time of the simulation in **Figure 3**.

```
W.Plot_3D(W.Get_Deflection(A), tf, 'Deflection at␣
↪t='+str(tf))
```

The related Jupyter notebook [6] further illustrates how to obtain a structure-

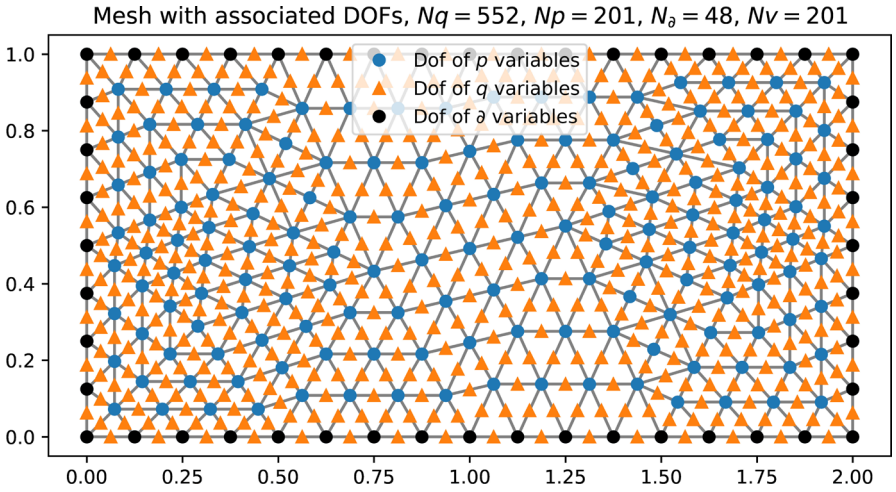

Figure 1. Two-dimensional triangular mesh with corresponding degrees of freedom for each variable for the anisotropic wave problem with damping.

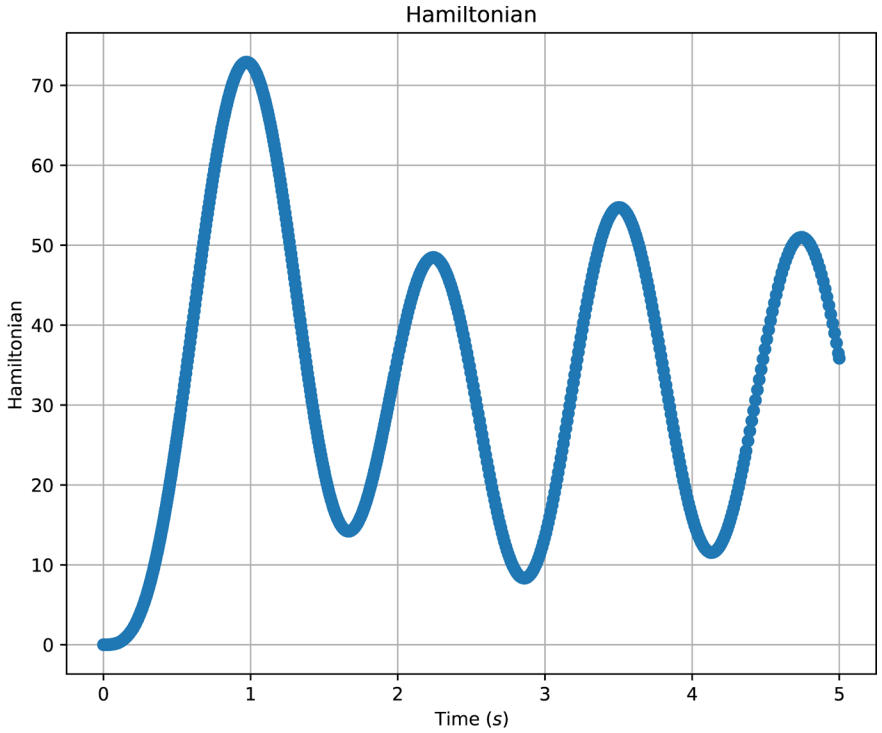

Figure 2. Hamiltonian function versus time for the anisotropic, heterogeneous and boundary controlled wave problem with damping.

preserving reduced model of this port-Hamiltonian system. After application of the model reduction algorithm proposed in [53], a pHODE of reduced size has to be integrated to obtain an approximate solution of the wave propagation problem. This is further illustrated on the simple application detailed in this section. In addition, a supplementary notebook illustrates the numerical simulation of the wave equation problem, when mixed boundary conditions (*i.e.* Dirichlet and Neumann conditions) on the boundary control function are imposed by Lagrange multipliers [42].

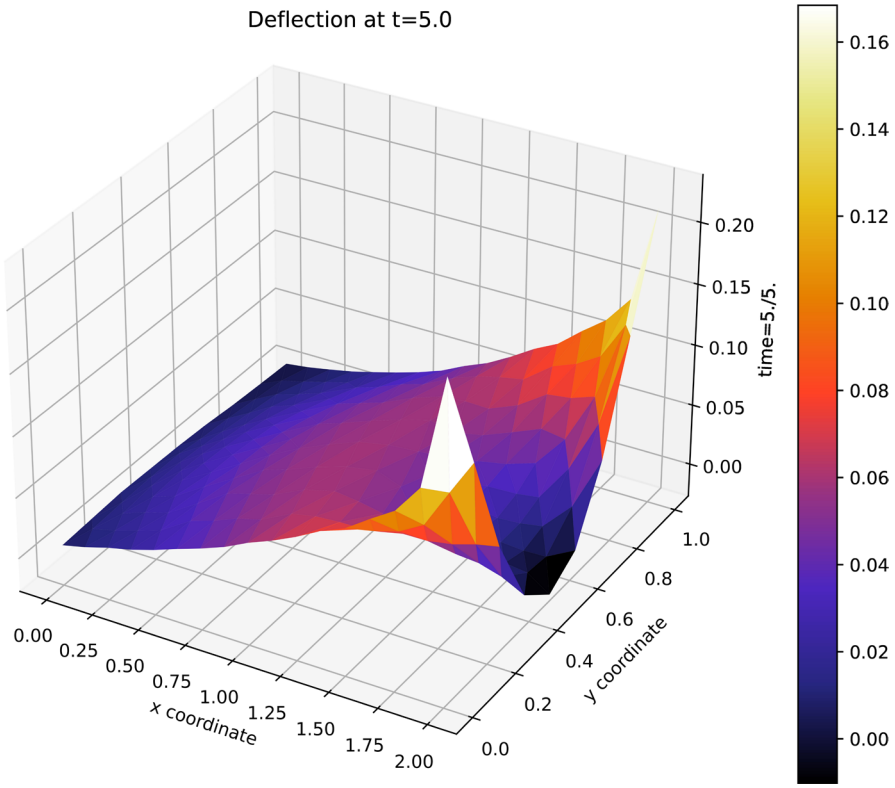

Figure 3. Deflection at the final time for the anisotropic, heterogeneous and boundary controlled wave problem with damping.

5.2. The Mindlin Plate Problem

We first recall the considered continuous problem related to the Mindlin plate and tackle the semi-discretization in space of the pHs by PFEM. After transformation and time discretization, we perform a numerical simulation to obtain an approximation of the space-time solution. As in Section 5.1, the procedure is described step-by-step and detailed explanations and numerical illustrations are provided.

5.2.1. Problem Statement

Consider the Mindlin plate problem defined for all $t \geq 0$ as:

$$\begin{cases} \rho b \partial_{tt} w = \operatorname{div}(\boldsymbol{q}), & \boldsymbol{x} \in \Omega = \{0, L_x\} \times \{0, L_y\}, \\ \dfrac{\rho b^3}{12} \partial_{tt} \boldsymbol{\theta} = \boldsymbol{q} + \operatorname{Div}(\boldsymbol{M}) \end{cases} \tag{57}$$

with initial conditions:

$$\begin{aligned} w(0, \boldsymbol{x}) &= 0, & \boldsymbol{\theta}(0, \boldsymbol{x}) &= \boldsymbol{0}, & \boldsymbol{x} \in \Omega, t = 0, \\ \partial_t w(0, \boldsymbol{x}) &= w_1(x), & \partial_t \boldsymbol{\theta}(0, \boldsymbol{x}) &= \boldsymbol{0}, & \boldsymbol{x} \in \Omega, t = 0, \end{aligned} \tag{58}$$

and boundary conditions:

$$\begin{aligned} w|_{\Gamma_D} &= u_D(t), & \boldsymbol{\theta}|_{\Gamma_D} &= 0, & \Gamma_D &= \{x = 0\}, \\ \boldsymbol{q} \cdot \boldsymbol{n}|_{\Gamma_N} &= u_N(t), & \boldsymbol{M} \cdot \boldsymbol{n}|_{\Gamma_N} &= 0, & \Gamma_N &= \{x = L_x, y = 0, y = L_y\}. \end{aligned} \tag{59}$$

Mixed boundary conditions are considered in this example. The subsets Γ_N, Γ_D represent the subsets of the boundary where Neumann and Dirichlet conditions hold respectively. The Dirichlet conditions are enforced using Lagrange multipliers. The PFEM discretization then leads to a pHDAE, as explained in [42] for the wave equation. The following expressions have been considered for the initial and boundary conditions:

$$\begin{aligned} w_1 &= xy, \quad u_D = 0.01\left(\cos\left(2\pi\frac{t}{t_f}\right)-1\right), \\ u_N &= 10^5 \sin\left(2\pi\frac{x}{L_x}\right)\left(1-\exp^{-10\frac{t}{t_f}}\right). \end{aligned} \tag{60}$$

5.2.2. Setup

We initialize here the Python object related to the Mindlin class of SCRIMP. This object will be used throughout this section.

```
Min = SCRIMP.Mindlin()
```

5.2.3. Constants

We define the constants related to the rectangular domain Ω. The coordinates of the bottom left $(x_0, y_0) = (0,0)$ and the top right $(x_L, y_L) = (L_x, L_y)$ corners of the rectangle are required.

```
x0, xL, y0, yL = 0., 2., 0., 1.
```

As in the previous example, the time interval related to the time discretization is defined as follows:

```
ti, tf  = 0., 0.01
```

A Runge-Kutta method for the time integration of the system is prescribed. This method is conditionally stable, so the time-step has to be set accurately to avoid numerical instabilities.

```
dt           = 1.e-6
dae_library  = 'DAE:RK4_Augmented'
```

5.2.4. FEniCS Expressions Definition

The FEniCS library is also used in the Mindlin class of SCRIMP. The coefficients related to the physical parameters of the isotropic plate can be provided as either real numbers or FEniCS expressions.

```
E   = 7*10**10
rho = 2700
nu  = 0.3
h   = 0.1
k   = 5/6
```

Similarly the initial vertical condition w_1 can be defined as a string. It

represents a C++ code that will be compiled by the Dolfin library of FEniCS.

```
ew_0 = 'x[0]*x[1]'
```

This means that the initial velocity satisfies $w_1 = xy$. Note that the initial condition has to be compatible with the boundary conditions. The other initial conditions will be set to zero.

5.2.5. Problem at the Continuous Level

We are now able to completely define the problem at the continuous level. We start by specifying that the computational domain Ω is of rectangular shape. To define Ω, we provide the coordinates of the bottom left and top right corners to the Mindlin object.

```
Min.Set_Rectangular_Domain(x0, xL, y0, yL);
```

The time integration interval is then given.

```
Min.Set_Initial_Final_Time(ti, tf);
```

The physical parameters related to the Mindlin plate are set.

```
Min.Set_Physical_Parameters(rho, h, E, nu, k,␣
↪init_by_value=True);
```

Finally the initial conditions in terms of co-energy variables are also set.

```
Min.Set_Initial_Data(W_0='0', Th1_0='0', Th2_0='0',\
                     ew_0=ew_0, eth1_0='0', eth2_0='0',\
                     Ekap11_0='0', Ekap12_0='0',␣
↪Ekap22_0='0',\
                     egam1_0='0', egam2_0='0');
```

5.2.6. Problem at the Discrete Level in Space and Time

We start by selecting the computational mesh which is generated with FEniCS inner mesh utilities. The first parameter corresponds to a mesh refinement parameter.

```
Min.Generate_Mesh(10, structured_mesh=True);
```

To perform the discretization in space, the conforming finite element approximation spaces to be used has to be specified. The finite element for the linear and angular velocity are Lagrange polynomials of order r. The momenta tensor and shear stress are chosen as Discontinuous Galerkin elements of order $r-1$ [35]. This choice of finite elements is similar to the one proposed in [55], but a simplicial mesh is used instead of a quadrilateral one. By default, the boundary variables are approximated as Lagrange polynomials of order 1. This can be easily changed through the option family_b for the family, and rb for the degree.

```
Min.Set_Finite_Elements_Spaces(r=1);
```

We then perform the semi-discretization in space of the weak formulation with PFEM. At the end of this stage, the complete formulation of the pHDAE is obtained. The different matrices related to the pHDAE system are constructed in the Assembly_Mixed_BC method of the Mindlin class of SCRIMP and are directly accessible through the object of the Mindlin class. The subsets named G1, G2, G3, G4, denote the left, bottom, right and top sides of the rectangle, respectively.

In SCRIMP the boundary control $\boldsymbol{u}_\partial$ is assumed to take the form:

```
Ub_tm0(t) * Ub_sp0(x) + Ub_tm1(t) + Ub_sp1(x)
```

Its derivative $\dot{\boldsymbol{u}}_\partial$ is expressed as:

```
Ub_tm0_dir(t) * Ub_sp0(x) + Ub_tm1_dir(t)
```

To integrate in time we need to provide the derivative of the boundary condition. This information is provided by the variables Ub_tm0_dir, Ub_tm1_dir, respectively. This is needed to reduce the resulting DAE of index 2 to index 1.

```
Min.Set_Mixed_Boundaries(Dir=['G1'], Nor=['G2', 'G3', 'G4'])
Min.Assembly_Mixed_BC()
Min.Set_Mixed_BC_Normal(Ub_tm0=lambda t: np.array([(1 - np.
↪exp(-t/tf) ),0,0]) ,\
                        Ub_sp0=('100000*sin(2*pi/xL*x[0])',␣
↪'0.', '0.'))
amp   = 0.01
omega = 2*pi/tf
Min.Set_Mixed_BC_Dirichlet(Ub_tm0=lambda t : np.
↪array([-amp*omega*np.sin(omega*t),0,0]), Ub_sp0=('1.', '0.
↪','0.'),\
                           Ub_tm0_dir=lambda t : np.
↪array([-amp*omega**2*np.cos(omega*t),0,0]));
```

To perform the time integration of the pHDAE, we first need to interpolate the boundary control function and the initial data on the appropriate finite element spaces.

```
Min.Project_Boundary_Control()
Min.Project_Initial_Data();
```

Finally the specification of the parameters related to the time discretization is made.

```
Min.Set_Time_Setting(dt);
```

5.2.7. Numerical Approximation of the Space-Time Solution

For the numerical approximation of the solution of the pHDAE system, the algebraic condition is differentiated. The integrator 'DAE:RK4_Augmented' takes as input a pHDAE. Then, it exploits a projection method to express the Lagrange multiplier in terms of the unknown [56], thus reducing the original DAE system into a purely ODE one. This allows employing standard ODE solvers for the time integration, as discussed in Section 5.2.3.

```
A, Hamiltonian = Min.Time_Integration(dae_library)
```

5.2.8. Post-Processing

Post-processing is performed similarly as in Section 5.1.8. Hence we omit the related Python lines of code for sake of brevity. In Figure 4 the evolution of the Hamiltonian function is shown versus time. We note that the Dirichlet condition causes an increase in energy. In Figure 5 snapshots of the vertical deflection at different instants are shown. We remark that the Neumann boundary condition causes the plate to bend asymmetrically.

5.3. Anisotropic Heterogeneous Heat Equation

This third tutorial aims at illustrating PFEM to discretize the pHs presented in Section 3.4, modelling the heat equation. We specifically learn how to define and solve this problem with SCRIMP. We first define the continuous problem by using a specific class of SCRIMP related to the heat equation in two dimensions. Then we tackle the discretization in space of the pHs through PFEM. The discretization of the energy formulation leads to a *nonlinear* pHDAE formulation. After time discretization, we perform a numerical simulation to obtain an approximation of the space-time solution. Finally a simple post-processing is provided.

5.3.1. Problem Statement

We consider the two-dimensional heterogeneous anisotropic heat equation defined for all $t \geq 0$ as

$$\rho(\boldsymbol{x}) C_V(\boldsymbol{x}) \frac{\partial}{\partial t} T(t, \boldsymbol{x}) = \operatorname{div}\left(\boldsymbol{\lambda}(\boldsymbol{x}) \cdot \mathbf{grad} T(t, \boldsymbol{x})\right), \quad \boldsymbol{x} \in \Omega,$$

$$T(t, \boldsymbol{x}) = v_\partial(t, \boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega,$$

$$T(0, \boldsymbol{x}) = T_0(x), \quad \boldsymbol{x} \in \Omega, t = 0$$

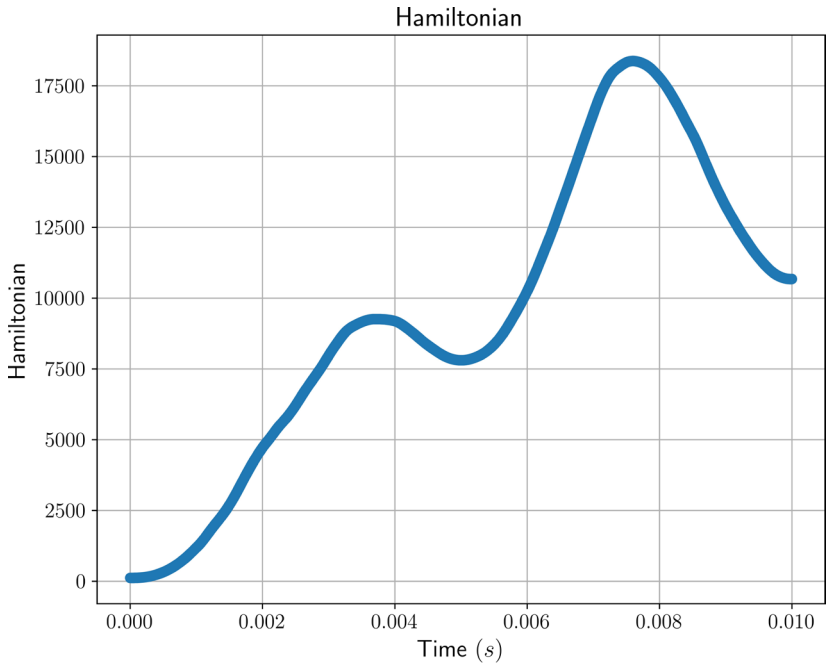

Figure 4. Hamiltonian versus time for the Mindlin plate problem.

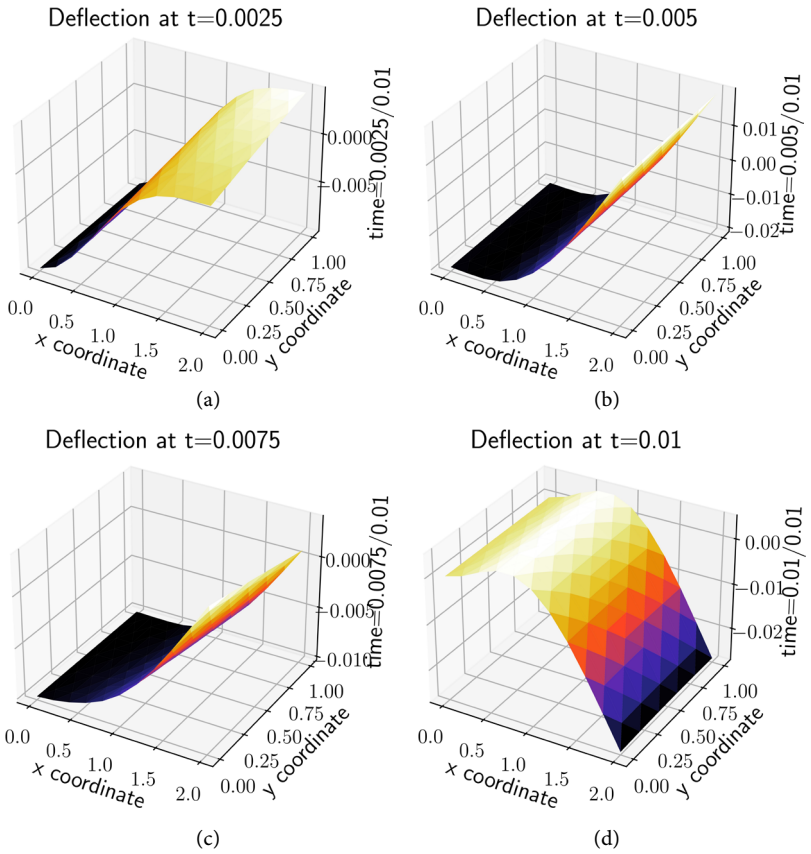

Figure 5. Snapshots of the vertical deflection of the Mindlin plate at different instants. (a) 0.35 Deflection w at $t = t_f/4$; (b) 0.35 Deflection w at $t = t_f/2$; (c) 0.35 Deflection w at $t = 3t_f/4$; (d) 0.35 Deflection w at $t = t_f$.

with $\Omega \subset \mathbb{R}^2$ an open bounded spatial domain with Lipschitz-continuous boundary $\partial\Omega$. $v_\partial(t, \boldsymbol{x})$ represents the boundary control function on the temperature (the notation u is kept for the internal energy density). $T(t, \boldsymbol{x})$ denotes the temperature at point $\boldsymbol{x} \in \Omega$ and time t. $\rho \in C^\infty(\Omega)$ (positive and bounded from below) denotes the mass density, $\lambda \in C^\infty(\Omega)^{2\times 2}$ (symmetric and coercive) the thermal conductivity. In the following Ω is assumed to be of rectangular shape.

5.3.2. Port-Hamiltonian Formulation

We refer to [9] [10] for the modeling and discretization of various port-Hamiltonian formulations of this problem. The authors consider quadratic Lyapunov functional, entropy or internal energy as Hamiltonian, respectively. We will consider the PFEM discretization of the internal energy functional formulation as proposed in Section 3.4, which will lead to a nonlinear pHDAE. Our goal in this tutorial is to show how a pHDAE system can be formulated and solved with SCRIMP.

5.3.3. Setup

We initialize here the Python object related to the energy formulation of the Heat_2D class of SCRIMP, that is assumed to be imported. This object will be

used throughout this tutorial.

```
H = Energy()
```

Energy corresponds to a class inherited from the Heat_2D base class. This base class contains implementations of the Lyapunov and entropy formulations as well.

5.3.4. Constants

The same lines of code as for the Wave_2D and Mindlin classes are used to define the constants related to the rectangular mesh.

```
x0, xL, y0, yL = 0., 2., 0., 1.
```

The time interval related to the time discretization is specified similarly.

```
ti, tf  = 0., 5.
```

We provide the time step for the time discretization of the pHDAE as well.

```
dt = 1.e-3
```

5.3.5. FEniCS Expressions Definition

Using FEniCS expressions, the physical parameters related to our model problem are defined.

```
rho    = 'x[0]*(xL-x[0]) + x[1]*(yL-x[1]) + 2'
Lambda11 = '5. + x[0]*x[1]'
Lambda12 = '(x[0]-x[1])*(x[0]-x[1])'
Lambda22 = '3.+x[1]/(x[0]+1)'
CV      = '3.'
```

The initial conditions of the problem related to the temperature and to the flow and effort variables are then given. The temperature follows a Gaussian behaviour for which we specify related parameters.

```
ampl, sX, sY, X0, Y0    = 1000, xL/6, yL/6, xL/2, yL/2
au_0 = '3000'
eu_0 = ' ampl * exp(- pow( (x[0]-X0)/sX, 2) - pow( (x[1]-Y0)/
 ↪sY, 2) ) + 1000'
```

The spatial part of the boundary control function is defined next.

```
Ub_sp0  = '''
           ( abs(x[0]) <= DOLFIN_EPS ?  1. * (yL-x[1])*x[1] :
 ↪ 0 )
        + ( abs(x[1])  <= DOLFIN_EPS ?  1. * (xL-x[0])*x[0] :
 ↪ 0 )
        + ( abs(xL - x[0]) <= DOLFIN_EPS ?  -15. *␣
 ↪(yL-x[1])*x[1] : 0 )
        + ( abs(yL - x[1]) <= DOLFIN_EPS  ? 1. *␣
 ↪(xL-x[0])*x[0] : 0 )
        '''
```

Finally we define the time-dependent part of the boundary control as a pure Python function. The whole boundary control function is then given as the

product of the two quantities (Ub_sp0 and Ub_tm0, respectively).

```
def Ub_tm0(t):
    if t<=2:
        return 500 * sin(2 * pi * t)
    else: return 0
```

5.3.6. Problem at the Continuous Level

We are now able to completely define the problem at the continuous level.

```
H.Set_Rectangular_Domain(x0, xL, y0, yL);
H.Set_Initial_Final_Time(initial_time=ti, final_time=tf);
H.Set_Physical_Parameters(rho=rho, Lambda11=Lambda11,␣
 ↪Lambda12=Lambda12, Lambda22=Lambda22, CV=CV);
```

5.3.7. Problem at the Discrete Level in Space and Time

The structure-preserving discretization of the infinite-dimensional pHs with PFEM is described in detail in [10]. This leads to the pHDAE given in (55). The definition of the system at the discrete level follows the same steps as for the two previous examples.

```
H.Set_Gmsh_Mesh(xmlfile='rectangle.xml', rfn_num=1);
H.Set_Finite_Element_Spaces(family_q='RT', family_p='P',␣
 ↪family_b='P',rq=0, rp=1, rb=1);
H.Assembly();
```

To perform the time integration of the pHDAE, we first need to set and interpolate the initial data and the boundary control function on the appropriate finite element spaces. Then, the time step is specified.

```
H.Set_Initial_Data(au_0=au_0, eu_0=eu_0, ampl=ampl, sX=sX,␣
 ↪sY=sY, X0=X0, Y0=Y0);
H.Project_Initial_Data();
H.Set_Boundary_Control(Ub_tm0=Ub_tm0, Ub_sp0=Ub_sp0,␣
 ↪Ub_tm1=lambda t :0, Ub_sp1='1000');
H.Project_Boundary_Control();
H.Set_Time_Setting(dt);
```

5.3.8. Numerical Approximation of the Space-Time Solution

Now we perform the time integration of the resulting pHDAE system and deduce the behaviour of the energy variables, the Hamiltonian with respect to the time and space variables, respectively. For the time discretization, we employ a fully explicit scheme, presented in [51] (Algorithm 2 of Section 4.4) as a first attempt.

```
H.Set_Formulation('div')
alpha_s, fS, fsig, es, eS, esig, Hamiltonian = H.
 ↪Integration_DAE();
```

5.3.9. Post-Processing

As an illustration, we plot the Hamiltonian function (*i.e.* the internal energy) versus time. The Hamiltonian function is constant after 2 seconds, when the boundary control is switched off, as expected by the first law of thermodynamics.

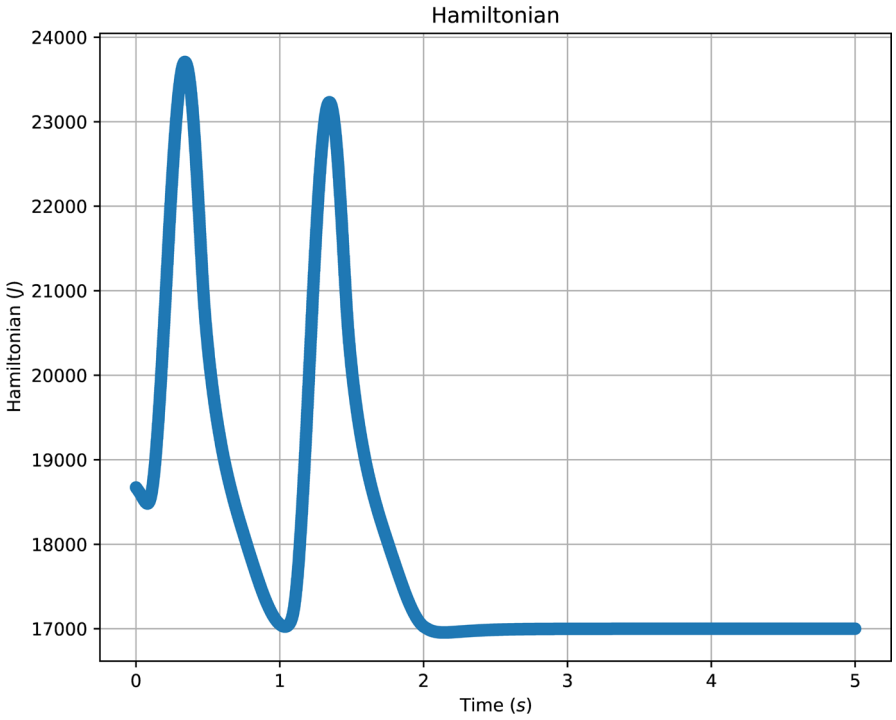

Figure 6. Hamiltonian (internal energy) versus time for the heat equation.

6. Conclusions and Perspectives

We have provided a general structure for the theoretical and numerical solution of infinite-dimensional port-Hamiltonian systems. This structure is particularly appealing since PFEM straightforwardly applies. Concerning the numerical solution, PFEM offers the advantage to leverage robust software components for the discretization of boundary controlled PDEs and time integration.

We have applied this strategy to abstract multidimensional linear hyperbolic and parabolic boundary controlled systems. We have notably shown that model problems based on the wave equation, Mindlin equation and heat equation fit within this unified theoretical framework. Numerical simulations of infinite-dimensional pHs have been performed with the ongoing software project SCRIMP that has been briefly introduced. Finally, we have illustrated how to solve three case studies within this framework by carefully explaining the methodology, and have provided companion interactive Jupyter notebooks.

Besides the generalization of the classes related to the heat and wave equation to the three-dimensional case, we plan to propose in SCRIMP more advanced model problems based on the two-dimensional Shallow Water Equation (SWE) [57] [41], the Kirchhoff model for thin plates [40] and Maxwell's equations [32]. Furthermore, we will investigate both time integration methods that allow structure-preserving time discretization [58] of finite-dimensional pHs and more accurate time integrators for nonlinear pHDAE. In addition, we plan to enrich the panel of structure-preserving model reduction algorithms to facilitate the simulation of large-scale port-Hamiltonian systems. This is an essential prerequisite

before first attempts related to control design. Further developments foresee the comparisons with well-established algorithms for multi-physics problems leading to coupled systems of PDEs.

Acknowledgements

This work is supported by the project ANR-16-CE92-0028, entitled *Interconnected Infinite-Dimensional systems for Heterogeneous Media*, INFIDHEM, financed by the French National Research Agency (ANR) and the Deutsche Forschungsgemeinschaft (DFG). Further information is available at https://websites.isae-supaero.fr/infidhem/the-project.

Moreover the authors would like to thank Michel Salaün and Denis Matignon for the fruitful and insightful discussions.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] van der Schaft, A.J. and Maschke, B. (2002) Hamiltonian Formulation of Distributed-Parameter Systems with Boundary Energy Flow. *Journal of Geometry and Physics*, **42**, 166-194. https://doi.org/10.1016/S0393-0440(01)00083-3

[2] Le Gorrec, Y. and Matignon, D. (2013) Coupling between Hyperbolic and Diffusive Systems: A Port-Hamiltonian Formulation. *European Journal of Control*, **19**, 505-512. https://doi.org/10.1016/j.ejcon.2013.09.003

[3] van der Schaft, A.J. and Maschke, B. (2018) Geometry of Thermodynamic Processes. *Entropy*, **20**, 1-23. https://doi.org/10.3390/e20120925

[4] Vu, N.M.T., Lefèvre, L. and Maschke, B. (2016) A Structured Control Model for the Thermo-Magneto-Hydrodynamics of Plasmas in Tokamaks. *Mathematical and Computer Modelling of Dynamical Systems*, **3954**, 1-26.

[5] Rashad, R., Califano, F., Van der Schaft, A.J. and Stramigioli, S. (2020) Twenty Years of Distributed Port-Hamiltonian Systems: A Literature Review. *IMA Journal of Mathematical Control and Information*, **37**, 1400-1422. https://doi.org/10.1093/imamci/dnaa018

[6] Brugnoli, A., Haine, G., Serhani, A. and Vasseur, X. (2020) Supplementary Material for "Numerical Approximation of Port-Hamiltonian Systems for Hyperbolic or Parabolic PDEs with Boundary Control". Dataset on Zenodo.

[7] Brugnoli, A., Alazard, D., Pommier-Budinger, V. and Matignon, D. (2019) Port-Hamiltonian Formulation and Symplectic Discretization of Plate Models Part I: Mindlin Model for Thick Plates. *Applied Mathematical Modelling*, **75**, 940-960. https://doi.org/10.1016/j.apm.2019.04.035

[8] Cardoso-Ribeiro, F.L., Matignon, D. and Lefèvre, L. (2018) A Structure-Preserving Partitioned Finite Element Method for the 2D Wave Equation. *IFAC-PapersOnLine*, **51**, 119-124. https://doi.org/10.1016/j.ifacol.2018.06.033

[9] Serhani, A., Haine, G. and Matignon, D. (2019) Anisotropic Heterogeneous n-D Heat Equation with Boundary Control and Observation: I. Modeling as Port-Hamiltonian System. *IFAC-PapersOnLine*, **52**, 51-56. https://doi.org/10.1016/j.ifacol.2019.07.009

[10] Serhani, A., Haine, G. and Matignon, D. (2019) Anisotropic Heterogeneous n-D Heat Equation with Boundary Control and Observation: II. Structure-Preserving Discretization. *IFAC-PapersOnLine*, **52**, 57-62. https://doi.org/10.1016/j.ifacol.2019.07.010

[11] Toledo, J., Wu, Y., Ramírez, H. and Le Gorrec, Y. (2020) Observer-Based Boundary Control of Distributed Port-Hamiltonian Systems. *Automatica*, **120**, Article ID: 109130. https://doi.org/10.1016/j.automatica.2020.109130

[12] Krug, R., Mehrmann, V. and Schmidt, M. (2020) Nonlinear Optimization of District Heating Networks. *Optimization and Engineering*. https://doi.org/10.1007/s11081-020-09549-0

[13] Golo, G., Talasila, V., Van der Schaft, A.J. and Maschke, B. (2004) Hamiltonian Discretization of Boundary Control Systems. *Automatica*, **40**, 757-771. https://doi.org/10.1016/j.automatica.2003.12.017

[14] Moulla, R., Lefèvre, L. and Maschke, B. (2012) Pseudo-Spectral Methods for the Spatial Symplectic Reduction of Open Systems of Conservation Laws. *Journal of Computational Physics*, **231**, 1272-1292. https://doi.org/10.1016/j.jcp.2011.10.008

[15] Trenchant, V., Ramírez, H., Le Gorrec, Y. and Kotyczka, P. (2018) Finite Differences on Staggered Grids Preserving the Port-Hamiltonian Structure with Application to an Acoustic Duct. *Journal of Computational Physics*, **373**, 673-697. https://doi.org/10.1016/j.jcp.2018.06.051

[16] Kotyczka, P., Maschke, B. and Lefèvre, L. (2018) Weak Form of Stokes-Dirac Structures and Geometric Discretization of Port-Hamiltonian Systems. *Journal of Computational Physics*, **361**, 442-476. https://doi.org/10.1016/j.jcp.2018.02.006

[17] Kotyczka, P. (2019) Numerical Methods for Distributed Parameter Port-Hamiltonian Systems. TUM University Press, Munich.

[18] Kirby, R.C. and Kieu, T.T. (2015) Symplectic-Mixed Finite Element Approximation of Linear Acoustic Wave Equations. *Numerische Mathematik*, **130**, 257-291. https://doi.org/10.1007/s00211-014-0667-4

[19] Altmann, R. and Schulze, P. (2017) A Port-Hamiltonian Formulation of the Navier-Stokes Equations for Reactive Flows. *Systems & Control Letters*, **100**, 51-55. https://doi.org/10.1016/j.sysconle.2016.12.005

[20] Haine, G., Matignon, D. and Serhani, A. (2020) Numerical Analysis of a Structure-Preserving Space-Discretization for an Anisotropic and Heterogeneous Boundary Controlled N-Dimensional Wave Equation as Port-Hamiltonian System.

[21] Joly, P. (2003) Variational Methods for Time-Dependent Wave Propagation Problems. In: Ainsworth, M., Davies, P., Duncan, D., Rynne, B. and Martin, P., Eds., *Topics in Computational Wave Propagation: Direct and Inverse Problems*, Volume 31 of Lecture Notes in Computational Science and Engineering, Springer, Berlin, 201-264.

[22] Boffi, D., Brezzi, F. and Fortin, M. (2013) Mixed Finite Element Methods and Applications. Volume 44 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin. https://doi.org/10.1007/978-3-642-36519-5

[23] Alnæs, M.S., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M.E. and Wells, G.N. (2015) The FEniCS Project Version 1.5. *Archive of Numerical Software*, **3**, 9-23.

[24] Van der Schaft, A. and Jeltsema, D. (2014) Port-Hamiltonian Systems Theory: An Introductory Overview. *Foundations and Trends® in Systems and Control*, **1**, 173-378. https://doi.org/10.1561/2600000002

[25] Courant, T.J. (1990) Dirac Manifolds. *Transactions of the American Mathematical Society*, **319**, 631-661. https://doi.org/10.1090/S0002-9947-1990-0998124-1

[26] Beattie, C., Mehrmann, V., Xu, H. and Zwart, H. (2018) Linear Port-Hamiltonian Descriptor Systems. *Mathematics of Control, Signals, and Systems*, **30**, 17. https://doi.org/10.1007/s00498-018-0223-3

[27] Mehrmann, V. and Morandin, R. (2019) Structure-Preserving Discretization for Port-Hamiltonian Descriptor Systems. 2019 *IEEE* 58*th Conference on Decision and Control* (*CDC*), Nice, 11-13 December 2019, 6863-6868. https://doi.org/10.1109/CDC40024.2019.9030180

[28] Renardy, M. and Rogers, R.C. (2004) An Introduction to Partial Differential Equations. Number 13 in Texts in Applied Mathematics. 2nd Edition, Springer-Verlag, New York.

[29] Tucsnak, M. and Weiss, G. (2009) Observation and Control for Operator Semi-Groups. Birkhäuser Advanced Texts: Basler Lehrbücher. Birkhäuser Verlag, Basel.

[30] Serhani, A., Matignon, D. and Haine, G. (2019) A Partitioned Finite Element Method for the Structure-Preserving Discretization of Damped Infinite-Dimensional Port-Hamiltonian Systems with Boundary Control. In: Nielsen, F. and Barbaresco, F., Eds., *Geometric Science of Information*, Volume 11712 of Lecture Notes in Computer Science, Springer, Cham, 549-558. https://doi.org/10.1007/978-3-030-26980-7_57

[31] Serhani, A., Matignon, D. and Haine, G. (2019) Partitioned Finite Element Method for Port-Hamiltonian Systems with Boundary Damping: Anisotropic Heterogeneous 2-D Wave Equations. *IFAC-PapersOnLine*, **52**, 96-101. https://doi.org/10.1016/j.ifacol.2019.08.017

[32] Payen, G., Matignon, D. and Haine, G. (2020) Modelling and Structure-Preserving Discretization of Maxwell's Equations as Port-Hamiltonian System. *Proceedings of the* 21*st IFAC World Congress*, Volume 53, 7671-7676. https://doi.org/10.1016/j.ifacol.2020.12.1355

[33] Kurula, M. and Zwart, H. (2015) Linear Wave Systems on n-D Spatial Domains. *International Journal of Control*, **88**, 1063-1077.

[34] Timoshenko, S. and Woinowsky-Krieger, S. (1959) Theory of Plates and Shells. Engineering Societies Monographs. McGraw-Hill, New York.

[35] Brugnoli, A. (2020) A Port-Hamiltonian Formulation of Flexible Structures. Modelling and Structure-Preserving Finite Element Discretization. PhD Thesis, Université de Toulouse, ISAE-SUPAERO, Toulouse.

[36] Arnold, D. and Lee, J. (2014) Mixed Methods for Elastodynamics with Weak Symmetry. *SIAM Journal on Numerical Analysis*, **52**, 2743-2769. https://doi.org/10.1137/13095032X

[37] Arnold, D. and Winther, R. (2002) Mixed Finite Elements for Elasticity. *Numerische Mathematik*, **92**, 401-419. https://doi.org/10.1007/s002110100348

[38] Arnold, D., Brezzi, F. and Douglas, J. (1984) Peers: A New Mixed Finite Element for Plane Elasticity. *Japan Journal of Applied Mathematics*, **1**, 347. https://doi.org/10.1007/BF03167064

[39] Beirão da Veiga, L., Mora, D. and Rodríguez, R. (2013) Numerical Analysis of a Locking-Free Mixed Finite Element Method for a Bending Moment Formulation of Reissner-Mindlin Plate Model. *Numerical Methods for Partial Differential Equations*, **29**, 40-63. https://doi.org/10.1002/num.21698

[40] Brugnoli, A., Alazard, D., Pommier-Budinger, V. and Matignon, D. (2019) Port-

Hamiltonian Formulation and Symplectic Discretization of Plate Models Part II: Kirchhoff Model for Thin Plates. *Applied Mathematical Modelling*, **75**, 961-981. https://doi.org/10.1016/j.apm.2019.04.036

[41] Cardoso-Ribeiro, F.L., Matignon, D. and Lefèvre, L. (2020) A Partitioned Finite Element Method for Power-Preserving Discretization of Open Systems of Conservation Laws. *IMA Journal of Mathematical Control and Information*, 1-41.

[42] Brugnoli, A., Cardoso-Ribeiro, F.L., Haine, G. and Kotyczka, P. (2020) Partitioned Finite Element Method for Power-Preserving Structured Discretization with Mixed Boundary Conditions. *Proceedings of the* 21*st IFAC World Congress*, Volume 53, 7647-7652. https://doi.org/10.1016/j.ifacol.2020.12.1351

[43] Linge, S. and Langtangen, H.P. (2020) Programming for Computations Python. Springer, Berlin. https://doi.org/10.1007/978-3-030-16877-3

[44] Alnæs, M.S., Logg, A., Ølgaard, K.B., Rognes, M.E. and Wells, G.N. (2014) Unified Form Language: A Domain-Specific Language for Weak Formulations of Partial Differential Equations. *ACM Transactions on Mathematical Software*, **40**, Article No. 9. https://doi.org/10.1145/2566630

[45] Kirby, R.C. and Logg, A. (2007) Efficient Compilation of a Class of Variational Forms. *ACM Transactions on Mathematical Software*, **33**, 17-es. https://doi.org/10.1145/1268769.1268771

[46] Logg, A. and Wells, G.N. (2010) DOLFIN: Automated Finite Element Computing. *ACM Transactions on Mathematical Software*, **37**, 20. https://doi.org/10.1145/1731022.1731030

[47] Logg, A., Mardal, K.A., Wells, G.N., *et al.* (2012) Automated Solution of Differential Equations by the Finite Element Method. Springer, Berlin. https://doi.org/10.1007/978-3-642-23099-8

[48] Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W.D., Karpeyev, D., Kaushik, D., Knepley, M.G., May, D.A., McInnes, L.C., Mills, R.T., Munson, T., Rupp, K., Sanan, P., Smith, B.F., Zampini, S., Zhang, H. and Zhang, H. (2020) PETSc Users Manual. Technical Report ANL-95/11 Revision 3.13, Argonne National Laboratory.

[49] Andersson, C., Führer, C. and Åkesson, J. (2015) Assimulo: A Unified Framework for ODE Solvers. *Mathematics and Computers in Simulation*, **116**, 26-43. https://doi.org/10.1016/j.matcom.2015.04.007

[50] Hindmarsh, A.C., Brown, P., Grant, K.E., Lee, S., Serban, R., Shumaker, D. and Woodward, C. (2005) SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers. *ACM Transactions on Mathematical Software* (*TOMS*), **31**, 363-396. https://doi.org/10.1145/1089014.1089020

[51] Serhani, A. (2020) Systèmes couplés d'EDPs, vus comme des systèmes Hamiltoniens à ports avec dissipation: Analyse théorique et simulation numérique. PhD Thesis, Université de Toulouse, ISAE-SUPAERO, Toulouse.

[52] Abhyankar, S., Brown, J., Constantinescu, E., Ghosh, D., Smith, B. and Zhang, H. (2018) PETSc/TS: A Modern Scalable ODE/DAE Solver Library.

[53] Chaturantabut, S., Beattie, C. and Gugercin, S. (2016) Structure-Preserving Model Reduction for Nonlinear Port-Hamiltonian Systems. *SIAM Journal on Scientific Computing*, **38**, B837-B865. https://doi.org/10.1137/15M1055085

[54] Egger, H., Kugler, T., Liljegren-Sailer, B., Marheineke, N. and Mehrmann, V. (2018) On Structure-Preserving Model Reduction for Damped Wave Propagation in Transport Networks. *SIAM Journal on Scientific Computing*, **40**, A331-A365.

https://doi.org/10.1137/17M1125303

[55] Cohen, G. and Grob, P. (2007) Mixed Higher Order Spectral Finite Elements for Reissner-Mindlin Equations. *SIAM Journal on Scientific Computing*, **29**, 986-1005. https://doi.org/10.1137/050642332

[56] Benner, P. and Heiland, J. (2015) Time-Dependent Dirichlet Conditions in Finite Element Discretizations. ScienceOpen Research. https://doi.org/10.14293/S2199-1006.1.SOR-MATH.AV2JW3.v1

[57] Cardoso-Ribeiro, F., Brugnoli, A., Matignon, D. and Lefèvre, L. (2019) Port-Hamiltonian Modeling, Discretization and Feedback Control of a Circular Water Tank. 2019 *IEEE* 58*th Conference on Decision and Control* (*CDC*), Nice, 11-13 December 2019, 6881-6886. https://doi.org/10.1109/CDC40024.2019.9030007

[58] Kotyczka, P. and Lefèvre, L. (2018) Discrete-Time Port-Hamiltonian Systems: A Definition Based on Symplectic Integration. *Systems and Control Letters*, **133**, Article ID: 104530. https://doi.org/10.1016/j.sysconle.2019.104530