Comptes Rendus Mécanique

Alice Peyraut and Martin Genet

Finite strain formulation of the discrete equilibrium gap principle: application to direct parameter estimation from large full-fields measurements

Volume 353 (2025), p. 259-308

Online since: 15 January 2025

https://doi.org/10.5802/crmeca.279

ACADÉMIE
DES SCIENCES
INSTITUT DE FRANCE

Comptes Rendus. Mécanique
2025, Vol. 353, p. 259-308
https://doi.org/10.5802/crmeca.279

Research article / *Article de recherche*

Finite strain formulation of the discrete equilibrium gap principle: application to direct parameter estimation from large full-fields measurements

Formulation en transformation finie du principe d'écart d'équilibre discret : application à l'estimation directe de paramètres à partir de mesures de champs

Alice Peyraut[*, a, b] **and Martin Genet**[a, b]

[a] Solid Mechanics Laboratory, École Polytechnique/IPP/CNRS, France
[b] MEDISIM Team, INRIA, France
E-mail: alice.peyraut@polytechnique.edu (A. Peyraut)

Abstract. The Equilibrium Gap Method (EGM) is a direct model parameter identification method, i.e., that does not require any resolution of the model. It has been extensively studied in the context of small strains but not thoroughly investigated for large strains. In this article, we propose a novel formulation of the EGM, valid in large strains, and applicable to both boundary and body forces, when full-field measurements are available. Our formulation is based on a recently proposed continuous formulation and consistent discretization of the equilibrium gap principle. Additionally, we developed an estimation pipeline to quantify the robustness of our new EGM formulation to noise, and we compared its performance to other classical estimation methods, namely the Finite Element Model Updating (FEMU) method and the Virtual Fields Method (VFM). Our robustness quantification pipeline involves generating synthetic data from a reference model through two methods: by adding noise to the reference displacement, or by generating noisy images and performing motion tracking with the Equilibrium Gap principle used as mechanical regularization. While the quality of estimation using our new EGM formulation is poor with the first data generation method, it improves drastically with the second method. Since the second method of synthetic data generation closely mimics experimental processes, the EGM, when combined with motion tracking with Equilibrium Gap regularization, demonstrates reasonable noise robustness. Thus, it is a promising option for direct parameter estimation from full-field measurements.

Résumé. La méthode de l'écart d'équilibre (EGM) est une méthode directe d'identification de paramètres de modèles, c'est-à-dire qu'elle ne nécessite aucune résolution du modèle. Elle a été largement étudiée dans le contexte des petites déformations, mais n'a pas fait l'objet d'un examen approfondi pour les grandes déformations. Dans cet article, nous proposons une nouvelle formulation de l'EGM, valable pour les grandes déformations et applicable à la fois aux forces surfaciques et aux forces volumiques, lorsque des mesures plein champ sont disponibles. Notre approche est basée sur une formulation continue et une discrétisation consistante du principe de l'écart d'équilibre récemment proposées. En outre, nous avons développé un pipeline pour quantifier la robustesse de notre nouvelle formulation EGM au bruit, et nous avons comparé ses performances à d'autres méthodes d'estimation classiques, à savoir la méthode « Finite Element Method Updating »

[*] Corresponding author

(FEMU) et la méthode des champs virtuels (VFM). Notre pipeline de quantification de la robustesse implique la génération de données synthétiques à partir d'un modèle de référence via deux méthodes potentielles : soit en ajoutant du bruit au déplacement de référence, soit en générant des images bruitées et en effectuant un suivi de mouvement avec le principe de l'écart d'équilibre utilisé comme régularisation mécanique. Alors que la qualité de l'estimation utilisant notre nouvelle formulation EGM est faible avec la première méthode de génération de données, elle s'améliore considérablement avec la seconde méthode. Étant donné que la deuxième méthode de génération de données synthétiques reproduit fidèlement les processus expérimentaux, l'EGM, lorsqu'elle est associée au suivi des mouvements avec régularisation de l'écart d'équilibre, fait preuve d'une robustesse raisonnable face au bruit. Il s'agit donc d'une option prometteuse pour l'estimation directe de paramètres à partir de mesures plein champ.

Keywords. Equilibrium Gap Method, Inverse problems, Large strains, Parameter identification, Uncertainty quantification.

Mots-clés. Méthode de l'écart à l'équilibre, Problèmes inverses, Grandes transformations, Identification, Quantification d'incertitude.

Funding. French National Research Agency (ANR) (Grant no: ANR-19-CE45-0007).

Manuscript received 26 August 2024, revised 5 December 2024, accepted 6 December 2024.

1. Introduction

Inverse problems represent a critical issue in many mechanical problems. This is especially true in biomechanics for example, where in-vivo testing is rarely feasible [1]. As a result, measurements derived from clinical imaging techniques, such as CT-scans or MRI, can be used to identify parameters such as tissue stiffness [2]. Thus, inverse problems are a valuable alternative to experimental methods for parameters identification.

Different estimation methods have been developed for inverse problems, which all present advantages and drawbacks in terms of accuracy, robustness to noise and model errors, and computation time. In the scope of this study, we will assume to have access to full-field measurements as input for the different estimation techniques. A review of the most common identification methods using full-field measurements can be found in [3] and [4]. Note however that methods such as the Finite Element Model Updating (FEMU) method, do not require full-field measurements to perform well, and can be applied to more limited data [5].

In this article we propose a new formulation of the Equilibrium Gap Method (EGM), a direct method—i.e., which does not require direct model resolution—, valid in large strains, with a consistent FE discretization. We then quantify the estimation quality of the EGM applied to large motion measurements, for data presenting high noise levels, and for boundary force and body force tests. If some estimation methods, such as the Virtual Fields Method (VFM) or FEMU, have been extensively studied in small strains [6], as well as in large strains [7, 8], the EGM has only been thoroughly studied in small strains [9, 10], but not in large strains.

If [11] already proposed a generalization of the VFM in large strains including all discrete virtual fields, which then represents an EGM formulation, we propose in this article an alternative formulation of the method, based on the novel Equilibrium Gap formulation of [12]. Our new formulation presents the advantage to be fully consistent: it converges when the FE mesh is refined.

Additionally, the EGM has mostly been studied for boundary forces, while considering that no body force was applied [9, 13]. Our new formulation allows to take into account full body fields. In order to develop a general formulation of the EGM, we therefore choose to take into account both boundary forces and body forces.

To quantify the performance of our new formulation, we study its noise robustness, and compare it to two other classical methods: the FEMU method [14, 15], which is a very robust but indirect method, i.e., a method requiring a model computation at each iteration, and the Virtual Fields Method (VFM) [16], which displays a lower noise robustness [17], but is potentially a direct, and hence less expensive method. Two families of virtual fields will be investigated for the VFM. The first family of virtual fields is generated with plane waves, which are not optimal for the simple problem considered here, but allow to study the impact of virtual field frequency on estimation performance. In this case the VFM is a direct method. The second family of virtual fields is generated based on the method proposed by [8], which was designed to apply the VFM to complex geometries. In this case, model computations are required, and the VFM is not a direct method anymore.

In this article, we therefore propose a performance quantification method, based on synthetic data generation. Two methods of data generation, based on generating noisy displacement fields and noisy images, are studied. We then apply the pipeline to our new formulation of the EGM, to the FEMU method and to the VFM, and study the impact of noise on the accuracy of the estimation.

2. Methods

2.1. *Estimation framework*

The focus of this article is the introduction of a new formulation of the Equilibrium Gap Method (EGM) for model parameter identification, and to compare quantitatively its performances to other usual estimation method. In order to formally introduce the formulation of the different methods, we start by introducing classical variables and operators.

Kinematics. The mapping from a material point at a position $\underline{X}$ in the reference configuration Ω (with $|\Omega|$ representing the reference volume) to a position $\underline{x}$ in a deformed configuration ω is represented by the function $\underline{\chi}$, such that

$$\underline{\chi} : \underline{X} \mapsto \underline{x}. \tag{1}$$

The displacement field from the reference to the deformed configuration $\underline{U}$ is then expressed by

$$\underline{U} := \underline{x} - \underline{X}. \tag{2}$$

In order to define the deformation gradient tensor, we classically derive it from the mapping:

$$\underline{\underline{F}} := \underline{\underline{\nabla}}\,\underline{\chi} = \underline{\underline{1}} + \underline{\underline{\nabla}}\,\underline{U}. \tag{3}$$

The volume change J is in turn defined through its relationship to the deformation gradient tensor:

$$J := \det(\underline{\underline{F}}). \tag{4}$$

We can define two additional quantities also related to the deformation tensor: the right Cauchy-Green tensor, expressed as

$$\underline{\underline{C}} := \underline{\underline{F}}^{\mathrm{T}} \cdot \underline{\underline{F}}, \tag{5}$$

and the Green–Lagrange strain tensor, expressed as

$$\underline{\underline{E}} := \tfrac{1}{2}(\underline{\underline{C}} - \underline{\underline{1}}). \tag{6}$$

We can also define the linearized strain tensor $\underline{\underline{\epsilon}}$ as

$$\underline{\underline{\epsilon}} := \tfrac{1}{2}(\underline{\underline{\nabla}}\,\underline{U} + \underline{\underline{\nabla}}\,\underline{U}^{\mathrm{T}}). \tag{7}$$

Throughout this article, we distinguish quantities defined in the deformed or the reference configuration. In particular, Γ and γ_t refer to the reference and deformed boundary where a load is applied. Similarly, we call Γ_0 and γ_{t0} the reference and deformed boundaries where a displacement is prescribed.

Additionally, the outer normal in the reference configuration is denoted $\underline{N}$, while the outer normal in the deformed configuration is denoted $\underline{n}$, which can be pulled back to the reference configuration:

$$\underline{n} = \frac{\underline{\underline{F}}^{-T}\,\underline{N}}{\|\underline{\underline{F}}^{-T}\underline{N}\|}. \tag{8}$$

Stresses. Let us also introduce various stress tensors. Both first and second Piola–Kirchhoff stress tensors, denoted as $\underline{\underline{P}}$ and $\underline{\underline{\Sigma}}$, can be related to the Cauchy stress tensor $\underline{\underline{\sigma}}$ using the operators previously introduced, omitting the mapping between the reference and the deformed configuration to simplify notations:

$$\underline{\underline{\Sigma}} := J\underline{\underline{F}}^{-1}\,\underline{\underline{\sigma}}\,\underline{\underline{F}}^{-T}, \tag{9}$$

and

$$\underline{\underline{P}} := J\,\underline{\underline{\sigma}}\,\underline{\underline{F}}^{-T}. \tag{10}$$

Loading. Lastly, let us introduce two vector fields, which both represent forces applied to the studied system. These two vector fields will be investigated in this article, and represent a particular case, which can however straightforwardly be generalized. The first vector field represents a boundary force applied to a given boundary of the system, and is denoted as $\underline{T}$ in the reference configuration and as $\underline{t}$ in the deformed configuration. Note that $\underline{T} = J\|\underline{\underline{F}}^{-T}\,\underline{N}\|\underline{t}$. The second vector field, denoted as $\underline{B}$ in the reference configuration and as $\underline{b}$ in the deformed configuration, represents a body force applied to the system. Note that $\underline{B} = J\underline{b}$. We can also define the vector field $\underline{U}_i$, which represent imposed displacements in the reference configuration.

Equilibrium. The equilibrium, which derives from the principle of conservation of momentum, can be expressed in terms of any stress measurement through the balance of internal and external virtual works:

$$\underline{\underline{\sigma}}\,/\,\underline{\underline{\Sigma}}\,/\,\underline{\underline{P}} \quad | \quad W_{\text{int}}\left(\underline{\underline{\sigma}}\,/\,\underline{\underline{\Sigma}}\,/\,\underline{\underline{P}};\underline{U}^*\right) = W_{\text{ext}}\left(\underline{U}^*\right)\ \forall\,\underline{U}^*, \tag{11}$$

where $\underline{U}^*$ represents any kinematically admissible to zero virtual field. Let us recall that when mentioning in this article that a virtual field $\underline{U}^*$ is kinematically admissible to zero, it implies that it is smooth enough on Ω, and that is satisfies homogeneous Dirichlet conditions on the boundaries where a displacement is prescribed, i.e., $\underline{U}^* = \underline{0}$ on $\underline{\Gamma}_0$.

Classically, W_{int} is expressed as:

$$\begin{aligned} W_{\text{int}}\left(\underline{\underline{\sigma}}\,/\,\underline{\underline{\Sigma}}\,/\,\underline{\underline{P}};\underline{U}^*\right) &:= \int_\omega \underline{\underline{\sigma}} : \underline{\underline{\epsilon}}(\underline{U}^*)\,\mathrm{d}\omega \\ &= \int_\Omega \underline{\underline{\Sigma}} : d_U\underline{\underline{E}}\cdot\underline{U}^*\,\mathrm{d}\Omega \\ &= \int_\Omega \underline{\underline{P}} : \underline{\underline{\nabla}}\,\underline{U}^*\,\mathrm{d}\Omega, \end{aligned} \tag{12}$$

where $d_U\underline{\underline{E}}\cdot\underline{U}^* = (\underline{\underline{F}}^{\mathrm{T}}\cdot\underline{\underline{\nabla}}\,\underline{U}^*)_{\text{sym}}$ is the first differential of the Green–Lagrange strain tensor.

Additionally, W_{ext} can be expressed as:

$$\begin{aligned} W_{\text{ext}}\left(\underline{U}^*\right) &:= \int_{\gamma_t} \underline{t}\cdot\underline{U}^*\,\mathrm{d}\gamma + \int_\omega \underline{b}\cdot\underline{U}^*\,\mathrm{d}\omega \\ &= \int_\Gamma \underline{T}\cdot\underline{U}^*\,\mathrm{d}\Gamma + \int_\Omega \underline{B}\cdot\underline{U}^*\,\mathrm{d}\Omega. \end{aligned} \tag{13}$$

Behavior. According to the second principle of thermodynamics, the second Piola–Kirchhoff stress tensor $\underline{\underline{\Sigma}}$, which depends on the state variable $\underline{\underline{E}}$, derives from the material free energy potential Ψ:

$$\underline{\underline{\Sigma}} = \frac{\partial \Psi}{\partial \underline{\underline{E}}}. \tag{14}$$

Problem formulation. Based on the equilibrium and behavior defined previously, the problem consists in solving the following equation:

$$\underline{U} \quad | \quad W_{\text{int}}\left(\underline{\underline{\Sigma}}(\underline{U}); \underline{U}^*\right) = W_{\text{ext}}\left(\underline{U}; \underline{U}^*\right) \; \forall \, \underline{U}^*. \tag{15}$$

Linearization. In this study, we will also perform analyses in the small strains setting, aiming to simplify the discussion. When considering linearized quantities, the internal virtual work is expressed as

$$W_{\text{int}}(\underline{U}, \underline{U}^*) \approx \int_\Omega \underline{\underline{\sigma}}(\underline{U}) : \underline{\underline{\epsilon}}(\underline{U}^*) \, \mathrm{d}\Omega, \tag{16}$$

with $\underline{\underline{\sigma}}(\underline{U}) = \underline{\underline{\underline{\underline{K}}}} : \underline{\underline{\epsilon}}(\underline{U})$, where $\underline{\underline{\underline{\underline{K}}}}$ is the material stiffness tensor, i.e., such that

$$\Psi = \tfrac{1}{2} \underline{\underline{\epsilon}} : \underline{\underline{\underline{\underline{K}}}} : \underline{\underline{\epsilon}}. \tag{17}$$

Similarly, the external virtual work can be expressed as:

$$W_{\text{ext}}(\underline{U}^*) \approx \int_\Omega \underline{b} \cdot \underline{U}^* \, \mathrm{d}\Omega + \int_\Gamma \underline{t} \cdot \underline{U}^* \, \mathrm{d}\Gamma. \tag{18}$$

Finite elements resolution. Using finite elements discretization, the linearized direct problem reduces to

$$\underline{\mathbb{U}} \quad | \quad \underline{\underline{\mathbb{K}}} \cdot \underline{\mathbb{U}} = \underline{\mathbb{F}}, \tag{19}$$

where $\underline{\underline{\mathbb{K}}}$ is the system stiffness matrix, defined as

$$\underline{\underline{\mathbb{K}}} := \int_\Omega \underline{\underline{\underline{\mathbb{Q}}}} : \underline{\underline{\underline{\underline{K}}}} : \underline{\underline{\underline{\mathbb{Q}}}}^{\mathrm{T}}, \tag{20}$$

with $\underline{\underline{\underline{\mathbb{Q}}}}$ the array of shape functions symmetric gradients. $\underline{\mathbb{F}}$ represents the system force vector, which can include both body and boundary forces, and is expressed as

$$\underline{\mathbb{F}} := \int_\Omega \underline{\underline{\mathbb{N}}} \cdot \underline{b} \, \mathrm{d}\Omega + \int_\Gamma \underline{\underline{\mathbb{N}}} \cdot \underline{t} \, \mathrm{d}\Gamma, \tag{21}$$

with $\underline{\underline{\mathbb{N}}}$ the array of the shape functions. Additionally, $\underline{\mathbb{U}}$ is the nodal displacement vector, defined through the approximation of the displacement field $\underline{U}$, such that

$$\underline{U} = {}^{\mathrm{t}}\underline{\underline{\mathbb{N}}} \cdot \underline{\mathbb{U}}. \tag{22}$$

We can also introduce a diagonal matrix $\underline{\underline{\mathbb{D}}}$, which selects the degrees of freedom where the force is imposed. It is particularly useful for distinguishing cases where only boundary forces or only body forces are applied. For example, if only boundary forces are applied, $\underline{\underline{\mathbb{D}}}$ is equal to 1 on the boundary degrees of freedom and 0 everywhere else.

Estimation problem formulation. In general, not all model parameters can necessarily be identified based on the available data. As a result, for inverse problems, some parameters, which are called $\underline{\theta}$, are identified, while other parameters, called $\underline{\eta}$, are chosen at reference values during the estimation process. Note that estimated parameters can be material or loading parameters.

The estimation process consists in finding the optimized parameters $\underline{\theta}^{\text{sol}}$, which minimize a cost function L. The expression of the cost function depends on the chosen estimation method. In general, the estimation problem can therefore be written as:

$$\underline{\theta}^{\text{sol}} := \underset{\underline{\theta}}{\operatorname{argmin}}\{L(\underline{\theta},\underline{\eta})\}. \tag{23}$$

This cost function implicitly includes a displacement field $\underline{U}_{\text{meas}}$, extracted from experimental measurements. Its finite element discretization is denoted as $\underline{\mathbb{U}}_{\text{meas}}$.

2.2. *Recall of existing identification methods*

This section recalls the principle and formulation of three classical identification methods, whose robustness to noise and model errors is investigated in this article.

2.2.1. *The Finite Element Model Updating (FEMU) method*

Principle. The "displacement" identification method, a.k.a. FEMU (Finite Element Model Updating), first introduced in [14], has been widely used in estimation problems, and is the most robust identification method [4, 18]. The FEMU method estimation relies on a very intuitive process, which consists in finding the parameters minimizing the distance between the displacement computed with the model, which depends on the different parameters, and the measured displacement, which is typically extracted from experimental images [3]. This method therefore requires a finite element computation at each iteration; as such, FEMU is also one of the most expensive identification methods. Note that FEMU is the only method presented in this article, which does not require full-field measurements for the estimation. It can also perform well when only partial data is available.

Cost function. The identification problem can be formalized as follows [4]:

$$\begin{aligned} L^{\text{FEMU}}(\underline{\theta},\underline{\eta}) &:= \tfrac{1}{2}\|\underline{U}(\underline{\theta},\underline{\eta}) - \underline{U}_{\text{meas}}\|^2_{L_2(\Omega)} \\ &\approx \tfrac{1}{2}\|\underline{\mathbb{U}}(\underline{\theta},\underline{\eta}) - \underline{\mathbb{U}}_{\text{meas}}\|^2_{l_2}. \end{aligned} \tag{24}$$

2.2.2. *The Virtual Fields Method (VFM)*

Principle. The FEMU method, requiring the resolution of many direct problems, is computationally expensive. Direct methods, which do not require any resolution of the considered model, are theoretically cheaper [4]. The Virtual Fields Method (VFM) is a potentially direct identification method, which has been widely used in various problems [19]. It has notably allowed to successfully identify parameters in small strains for anisotropic elasticity [20], plasticity [21] or elasto-plastic behavior [22]. The VFM has also been used for inverse problems in large strains [23]. Additionally, most identification problems use experimental data, which are frequently noisy. Being theoretically a direct method, VFM should be less expensive but also more sensitive to noise than FEMU. The sensitivity of the VFM to noise has been extensively studied in [16, 17, 21].

The VFM is based on the principle of virtual work, presented in Equation (11). This method relies on defining at least as many kinematically admissible to zero virtual displacement fields as parameters to identify. Each virtual displacement field is then plugged into Equation (11) as the virtual field $\underline{U}^*$, which leads to a system of equations to solve. Note that when as many virtual fields as parameters to identify are defined, the resolution may be explicit for simple material laws, e.g., when the stress tensor depends linearly on the parameters.

Cost function. The VFM relies on the principle of virtual work:

$$L^{\mathrm{VFM}}(\underline{\theta},\underline{\eta}) := \sum_{i=1}^{m} \left(W_{\mathrm{int}}(\underline{U}_{\mathrm{meas}};\underline{U}_i^*;\underline{\theta},\underline{\eta}) - W_{\mathrm{ext}}(\underline{U}_{\mathrm{meas}};\underline{U}_i^*;\underline{\theta},\underline{\eta}) \right)^2, \tag{25}$$

where $\underline{U}^*_{i=1,\ldots,m}$ represents the chosen kinematically admissible to zero virtual fields.

Note that at least as many kinematically admissible to zero virtual fields as parameters should be chosen to define our equation system (i.e., $m \geq \dim(\underline{\theta})$). If it is possible to take into account more virtual fields than parameters ($m > \dim(\underline{\theta})$), this however leads to an over determined system, and we will limit our study to the standard setting $m = \dim(\underline{\theta})$.

2.2.3. *The Equilibrium Gap Method (EGM)*

Principle. The Equilibrium Gap Method (EGM) is another potentially direct identification method that has been widely used for inverse problems in small strains [9]. As a direct method, EGM should be computationally more efficient, although less robust to noise and to model errors than methods such as FEMU [4]. The EGM can be seen as a generalization of the VFM, where all—in the discrete sense—virtual fields are considered [11]. The accuracy of the VFM depends on the choice of the virtual field. If the virtual field is well-adapted, the estimation is hence much more accurate than for less judicious choices of virtual fields. The EGM takes into account all possible virtual fields, the best suited as the worst ones; in average, the estimation is therefore poorer than for a "good" virtual field, but better than for a "bad" choice of virtual field. Note that the EGM has been mainly used for problems that do not include body forces.

Cost function. The EGM consists in finding the parameters that generate stresses that best respect the equilibrium. In the context of the small perturbation hypothesis, at the discrete level, the estimation has therefore been written as [4]:

$$L^{\mathrm{EGM},I}(\underline{\theta},\underline{\eta}) := \tfrac{1}{2} \| \underline{\underline{\mathbb{D}}}\,\underline{\underline{\mathbb{K}}}(\underline{\theta},\underline{\eta})\,\underline{\mathbb{U}}_{\mathrm{meas}} - \underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}(\underline{\theta},\underline{\eta}) \|_{l_2}^2, \tag{26}$$

"I" denoting here the cost function for infinitesimal strains.

2.2.4. *Brief review of other existing methods*

For the sake of simplicity, only the methods described previously will be used for comparison with our new formulation of the EGM. There are however many other methods, which are used for inverse problems. In particular, [24] proposed the Constitutive Relation Error (CRE) method, from which other methods, such as the modified Constitutive Relation Error (mCRE) method [25, 26] or the Constitutive Equation Gap Method [3] derive. Additionally, since the EGM usually displays a very high noise sensitivity [4], other formulations, based on the EGM, were developed to address this problem and decrease the noise sensitivity of the method. In particular, the Reconditioned Gap Method was developed in [4, 10], and [27] proposed the Reciprocity Gap Method (RGM) as estimation method. However, these methods require finite element computations, and are hence not direct. In the scope of this article, we will hence only take into account a model updating method (FEMU), and a potentially direct method (VFM) for comparison with our new EGM formulation, which is detailed hereafter. Note that other methods, such as the Reconditioned Gap Method are also very interesting, as they are faster than FEMU for example, and more robust than raw EGM. For the sake of simplicity, however, we only focus on the three specific methods previously mentionned.

2.3. *General formulation of the Equilibrium Gap Method*

Principle. If the EGM has been widely studied for small strains, there has not been yet to our knowledge a formulation in large strains, for hyperelastic problems, including both surface and body forces. Boddapati et al. [11] has proposed a generalization of VFM in large strains, by including all discrete virtual fields, which then represents an EGM formulation. We propose in this article an alternative formulation of the EGM in large strains, based on a novel Equilibrium Gap formulation introduced in [12] for motion tracking regularization. Our formulation is fully consistent, in the sense that it generalizes to any number of degrees of freedom, as it derives from a proper discretization of a continuous term. Note also that our formulation linearizes into Equation (26), with an additional mass matrix that makes the term consistent.

As for the classical formulation of the EGM, our proposed new expression in large strains consists in finding the parameters $\underline{\theta}$, which generate stresses that best verify the equilibrium. Notably, in the general case, the parameters $\underline{\theta}$ should verify the equilibrium of body and boundary forces.

Formulation. In the scope of our problem, we will only consider body and boundary forces. Note that imposed displacements are not considered; it would however be trivial to take them into account [4]. As a reminder, the strong form of the equilibrium, when applied to the measured displacement field U_{meas}, is:

$$\begin{cases} \underline{\nabla}\cdot\underline{\underline{P}}(\underline{U}_{\text{meas}})+\underline{B}=0 & \text{in } \Omega \\ \underline{\underline{P}}(\underline{U}_{\text{meas}})\cdot\underline{N}=\underline{T} & \text{on } \Gamma. \end{cases} \tag{27}$$

First, we want to quantify the non-verification of the first Equation of (27) by integrating the norm of its residual over the mesh. However, the divergence of $\underline{\underline{P}}$ is not defined for usual finite element discretizations on edges in 2D and on faces in 3D. Hence, [12] proposed an approach to consider the divergence of a tensor in the finite element formulation. Following this approach, let us introduce a new vector $\underline{\Pi}$, which corresponds to the projection of $\underline{\nabla}\cdot\underline{\underline{P}}+\underline{B}$ onto a space of square integrable functions V. As a result, we can write $\underline{\Pi}$ as:

$$\underline{\Pi} \quad | \quad \int_\Omega \underline{\Pi}\cdot\underline{\Pi}^*\,\mathrm{d}\Omega = -\int_\Omega \left(\underline{\nabla}\cdot\underline{\underline{P}}+\underline{B}\right)\cdot\underline{\Pi}^*\,\mathrm{d}\Omega \quad \forall\underline{\Pi}^*\in V, \tag{28}$$

where $\underline{\Pi}^*$ is any kinematically admissible field to zero belonging to V. Integrating by parts this expression yields:

$$\underline{\Pi} \quad | \quad \int_\Omega \underline{\Pi}\cdot\underline{\Pi}^*\,\mathrm{d}\Omega = \int_\Omega \underline{\underline{P}}:\underline{\underline{\nabla\Pi}}^* - \int_\Omega \underline{B}\cdot\underline{\Pi}^*\,\mathrm{d}\Omega \quad \forall\underline{\Pi}^*\in V. \tag{29}$$

Once discretized, this equation simply leads to the following linear system:

$$\underline{\underline{\mathbb{M}}}\cdot\underline{\mathbb{\Pi}}=\underline{\mathbb{R}}, \tag{30}$$

where $\underline{\underline{\mathbb{M}}}=\int_\Omega\underline{\underline{\mathbb{N}}}\cdot\underline{\underline{\mathbb{N}}}^{\mathrm{T}}$, $\underline{\mathbb{\Pi}}$ is such that $\underline{\Pi}=\underline{\underline{\mathbb{N}}}^{\mathrm{T}}\cdot\underline{\mathbb{\Pi}}$, and $\underline{\mathbb{R}}$ is defined as:

$$\mathbb{R}_i := \begin{cases} \displaystyle\int_\Omega \underline{\underline{P}}:\underline{\underline{\nabla N}}_i\,\mathrm{d}\Omega - \int_\Omega \underline{B}\cdot\underline{N}_i\,\mathrm{d}\Omega & \text{if } i \text{ is a body d.o.f} \\ 0 & \text{if } i \text{ is a boundary d.o.f,} \end{cases} \tag{31}$$

with $\underline{N}_i$ the shape function associated to the ith degree of freedom.

We can therefore define the contribution of the volume force to the cost function $L_\Omega^{\text{EGM},F}$—"F" denoting the cost function for finite strains—as

$$\begin{aligned} L_\Omega^{\text{EGM},F} &:= \tfrac{1}{2}\underline{\mathbb{\Pi}}^{\mathrm{T}}\cdot\underline{\underline{\mathbb{M}}}\cdot\underline{\mathbb{\Pi}} \\ &= \tfrac{1}{2}\underline{\mathbb{R}}^{\mathrm{T}}\cdot\underline{\underline{\mathbb{M}}}^{-1}\cdot\underline{\mathbb{R}}. \end{aligned} \tag{32}$$

Secondly, let us now consider the boundary force. In this case, we simply need to quantify the non-verification of the second Equation of (27):

$$L_\Gamma^{\text{EGM},F} := \tfrac{1}{2}\int_\Gamma (\underline{\underline{P}}\cdot\underline{N}-\underline{T})^2\,\mathrm{d}\Gamma. \tag{33}$$

This term corresponds to the contribution of the boundary force to the cost function.

Cost function. Based on the developments previously presented, the cost function can be written as:

$$L^{\mathrm{EGM},F}(\underline{\theta},\underline{\eta}) := L_{\Omega}^{\mathrm{EGM},F}(\underline{\mathbb{U}}_{\mathrm{meas}},\underline{\theta},\underline{\eta}) + L_{\Gamma}^{\mathrm{EGM},F}(\underline{\mathbb{U}}_{\mathrm{meas}},\underline{\theta},\underline{\eta}). \tag{34}$$

As a reminder, "F" denotes the cost function for finite strains. Note that the cost function $L^{\mathrm{EGM},F}$ depends on $\underline{\theta}$ and on $\underline{\eta}$ through the material law—included in $\underline{\underline{P}}$—and through the loading constants. As mentioned previously, note also that our formulation linearizes into Equation (26), with an additional mass matrix that makes the term consistent.

In the following sections of this article, any reference to our EGM will pertain to the new formulation. The FEMU method and the VFM will be employed as benchmarks to compare estimation performances of our new formulation.

2.4. *Performance quantification method*

All estimation methods presented previously, excepted FEMU, are only applicable for problems for which full-field displacement measurements are available. The estimation relies on finding the parameters $\underline{\theta}$ minimizing the cost functions defined in the previous section, which all take in argument (directly, e.g., in FEMU, or indirectly, e.g., in EGM) the displacement field measured from experimental images and $\underline{\theta}$. Note that for the VFM, when as many virtual fields as parameters are used, and for particular cases (e.g., elastic or simple hyperelastic law), the expression of the identified parameters can be explicit, and the method hence does not require any optimization process.

In general, all parameters cannot be identified. As reminded in Section 2.1, some parameters, $\underline{\theta}$, are hence estimated, while all other parameters, $\underline{\eta}$, are fixed at reference values $\underline{\eta}_0$. The optimized cost function then allows to access the estimated parameters $\underline{\theta}^{\mathrm{sol}}$. The estimation process is described Figure 1. However, when using experimental data, this process does not give any information on the error committed during the estimation, since the ground-truth parameters are unknown.

In order to quantify the reliability of the estimated parameters, we would like to investigate the impact of noise on the estimation. To do so, since the ground-truth values of the estimated parameters are unknown for real experimental images, we generated synthetic—and potentially noisy and biased data. Two different methods of data generation are studied here. The first approach involves computing a displacement field with our finite element model, using parameters set to benchmark values, to which noise is possibly added. The second approach consists in generating images with the model, and adding noise to these images, following the approach of [12]. Both approaches are detailed hereafter.

Estimation with noise added to the displacement fields. In order to create our synthetic data, a first—very classical [13]—option consists in using a displacement field computed with our model. Noise is then added to the displacement field in order to be closer to experimental data, which always contains noise. To do so, we first choose the parameters to identify $\underline{\theta}$ at reference values $\underline{\theta}_0$. We also fix the parameters that are not identified $\underline{\eta}$ at reference values $\underline{\eta}_0$. The associated displacement field $\underline{U}_0 = \underline{U}(\underline{\theta}_0,\underline{\eta}_0)$ is then computed with our finite element model.

The noise added to the displacement field corresponds to a random Gaussian field $\underline{\mathbb{G}}$, defined at the discrete level. We can also define a continuous random Gaussian field $\underline{g}$. For each degree of freedom, a random value is generated. This random value is drawn from a Gaussian distribution with a null mean and a standard deviation depending on the desired noise level, defined through the Signal-to-Noise-Ratio (SNR). The noisy displacement field corresponds to our synthetic measurement $\underline{\mathbb{U}}_{\mathrm{synth}} = \underline{\mathbb{U}}_0 + \underline{\mathbb{G}}$.

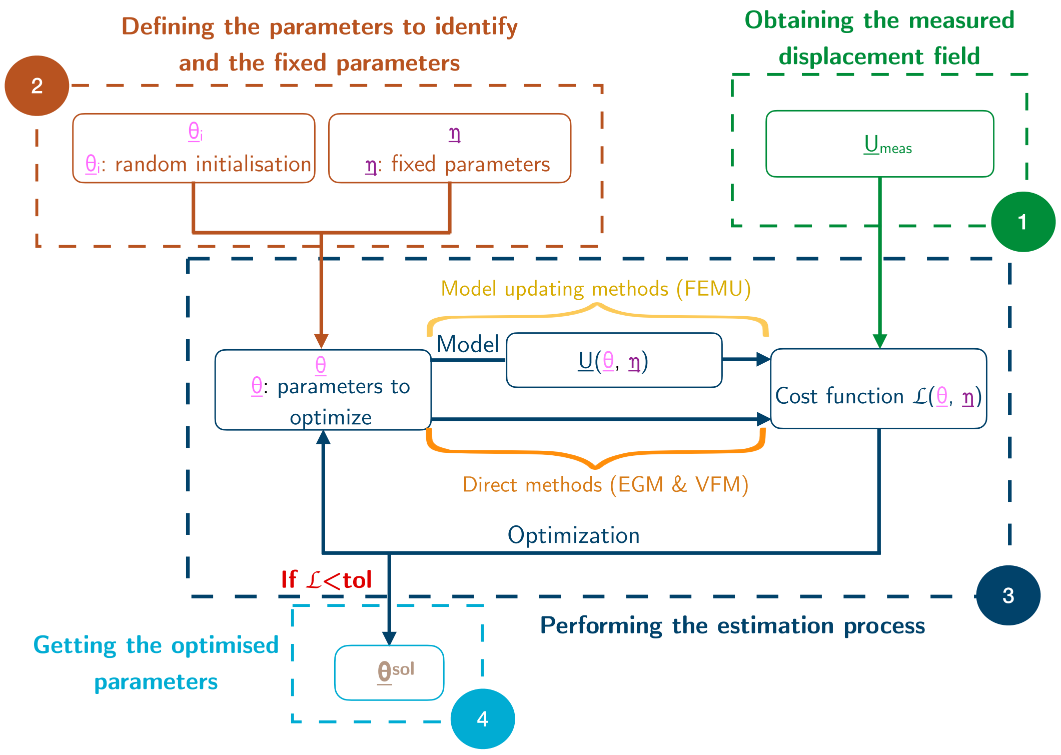

Defining the parameters to identify and the fixed parameters
2
θ_i
θ_i: random initialisation
η
η: fixed parameters
Obtaining the measured displacement field
U_meas
1
Model updating methods (FEMU)
Model
U(θ, η)
θ
θ: parameters to optimize
Cost function ℒ(θ, η)
Direct methods (EGM & VFM)
Optimization
If ℒ<tol
3
Performing the estimation process
Getting the optimised parameters
θ_sol
4

Figure 1. Method for identifying the parameters $\underline{\theta}$. "tol" is the tolerance for the minimization. Note that for the VFM, if there are as many virtual fields chosen as parameters to identify, and if W_{int} and W_{ext} are linear in $\underline{\theta}$, identifying $\underline{\theta}$ is equivalent to solving a linear system.

Additionally, when using real experimental data, the parameters $\underline{\eta}$ are fixed at reference values during the estimation process, but these reference values are very likely different from the ground-truth values $\underline{\eta}_0$. This is especially true in biomechanics, for example, where the material parameters are strongly patient-dependent [28]. Fixing $\underline{\eta}$ at a reference value will therefore very likely introduce errors into our model.

To study the impact of such model errors on the estimation, we can use biased values of $\underline{\eta}$ instead of the ground-truth values $\underline{\eta}_0$ in the estimation. We propose here to alter $\underline{\eta}_0$ with a bias $\underline{w}$. To do so, the synthetic measurement is generated with $\underline{\eta} = \underline{\eta}_0$. We then define a new parameter $\underline{\eta}_w$, by adding a bias $\underline{w}$ to $\underline{\eta}_0$. $\underline{\eta}_0$—the parameter used for building the synthetic measure—is then replaced by the parameter $\underline{\eta}_w = \underline{\eta}_0 + \underline{w}$ during the estimation. This method allows to investigate the impact of model errors on the estimation, and is illustrated in Figure 2.

Estimation with noise added to the images. Generating a noisy synthetic measurement by adding a random Gaussian field directly to the displacement field is classical, although not very realistic. This method indeed leads to a completely unstructured noise, which might be non-physical. As a result, we subsequently chose to investigate a second method of synthetic data generation closer to experimental processes. As such, let us first describe classical methods used to retrieve experimental displacement fields from experimental data.

The first step of this method consists in, similarly to the first method, creating a synthetic displacement field $\underline{U}_0 = \underline{U}(\underline{\theta}_0, \underline{\eta}_0)$, computed with the model. Images of the reference, I_0, and of the deformed, I_t, configurations are then created. To create images, different textures can be

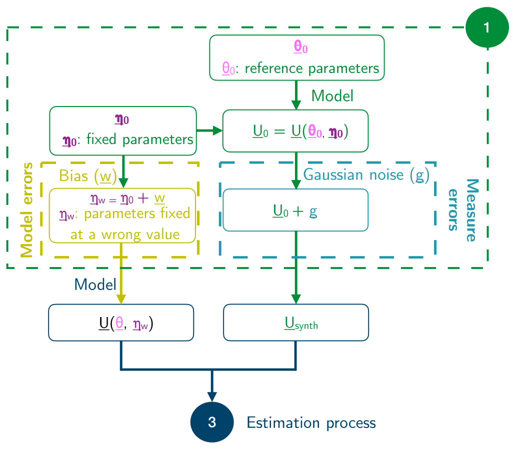
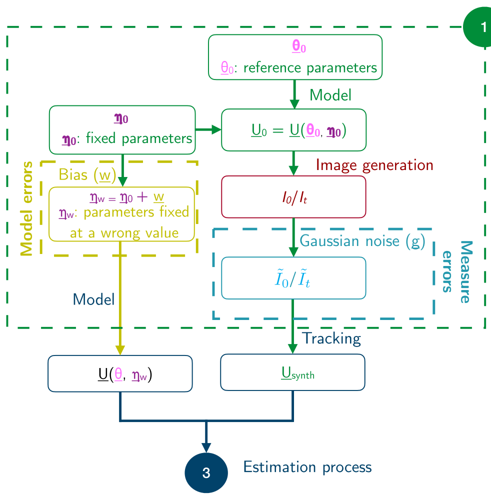
Generating a synthetic measurement by adding noise to displacements	Generating a synthetic measurement by adding noise to images
	

Figure 2. Pipeline for generating a synthetic measure, by adding random Gaussian noise to a reference displacement field (left) and to a reference image (right). When adding noise to a displacement field, a displacement field is computed with the model for reference parameter values. A random Gaussian field is then added to the displacement field. When adding noise to images, a reference displacement field is computed with the model by fixing the parameters at reference values. Associated reference and deformed images—I_0 and I_t—are then computed. A random Gaussian field is then generated to alter the pixel of both images, leading to noisy images $\tilde{I}_0$ and $\tilde{I}_t$. The synthetic displacement field is then retrieved with tracking.

used, such as no texture, MR-tagging-like texture, or sinusoidal patterns [29]. I_0 will be defined later. The deformed image is defined as

$$I_t(\underline{x}) = I_0(\underline{\chi}^{-1}(\underline{x})), \tag{35}$$

where $\underline{\chi}$ corresponds to the mapping associated to the synthetic displacement $\underline{U}_0$.

The next step consists in adding noise to the images: the pixels values of the images are altered with a random Gaussian noise, with null mean and a standard deviation chosen depending on the desired noise level. This leads to noisy images, denoted as $\tilde{I}_0$ and $\tilde{I}_t$. The synthetic "measured" displacement field $\underline{U}_{\text{synth}}$ is then retrieved by applying a tracking process, described hereafter.

In general, displacement fields are extracted from images, which contain noise, through motion tracking. The tracking process applied between two images can be formulated as finding the smooth mapping $\underline{\chi}$, or the smooth displacement $\underline{U}$—which is equivalent—, between the material points of the deformed image to the reference image, through the minimization problem [30, 31]:

$$\underline{U}_{\text{meas}} = \underset{\underline{U}}{\text{argmin}} \left\{ (1-\beta)\Xi^{\text{ima}}(\underline{U}) + \beta\Xi^{\text{reg}}(\underline{U}) \right\}, \tag{36}$$

with Ξ^{ima} the image correlation energy, Ξ^{reg} the regularization energy, and β the regularization strength. The regularization is added to implement physical constraints in the resolution, and hence provide structure to the measurements errors. [32] and [33] indeed showed that implementing regularization allowed to decrease the displacement resolution space, and to find a solution that follows a mechanical law. This method hence allows to filter high frequencies, as thoroughly explained in Section 2.2 (correlation procedure) of [34], which explains that the Equilibrium Gap Regularization acts as a fourth-order low-pass filter.

Even though more complex correlation energy terms, designed to decrease the impact of artifacts during tracking, have been proposed in [35, 36], we will consider here the following correlation energy:

$$\Xi^{\text{ima}}(\underline{U}) := \frac{1}{2}\int_{\Omega} (\tilde{I}_t(\underline{X}+\underline{U}(\underline{X})) - \tilde{I}_0(\underline{X}))^2 \, d\Omega. \tag{37}$$

This approach is commonly used in image intensity-based approaches [10, 35], and presents the advantage of being easily differentiable. Additionally, it is particularly adapted to our problem, as this expression represents the maximal likelihood (in the sense of the least squares) for Gaussian noise [4]. This tracking process is therefore applied, with different regularization levels. Note that in the scope of this article, we will only use the equilibrium gap regularization, introduced in large deformation by [30, 37], and later on improved in [12]. The formulation is thoroughly described in the latter article.

This regularization states that the displacement field obtained should be as close as possible to a problem with arbitrary boundary traction values. Creating a synthetic measurement with our second method relies on this tracking process. This synthetic measurement can then be used to study the impact of noise, but also the impact of model errors—by fixing the parameter $\underline{\eta}$ at an arbitrary value, which is different from the ground truth value $\underline{\eta}_0$—on the estimation. Note that if the regularization term for the body force introduced in [12] might seem similar to the term implemented in the cost function of our new formulation of the EGM, the two are in fact very different, as they serve different purposes (tracking and estimation respectively) and take different variables as argument.

For all methods, the noise added to the displacement field and to the images is a random Gaussian field. For a given noise level, the estimation might strongly depend on the random field generated. This is especially true for high noise levels, which can potentially lead to either overestimate or underestimate the identified parameter for some estimation methods. The estimation is therefore performed for many realizations of a given noise level, until convergence of the estimated parameters distributions.

2.5. *Illustrative framework*

This article aims at comparing the performance of our new formulation of the EGM with those of two other classical estimation methods—FEMU and VFM. We choose to perform the estimation on a specific problem, fairly simple for illustration purposes, described hereafter.

2.5.1. *Geometry*

The estimation is performed in plane strains on a square of characteristic length of 0.6 mm, meshed with triangular elements. The estimation is performed on a coarse mesh of 144 elements, as well as on a finer mesh of 556 elements. This simple geometry was chosen to perform a proof of concept of our method and validate its performances; it is indeed not straightforward that the method will work in large strains, for high noise levels and fine meshes. To illustrate that the approach can be extended to more complex geometries, we also propose in Appendix C the application of the method on a 3D problem.

2.5.2. *Constitutive law*

Since we would like to study inverse problems in large hyperelastic deformations, we choose a Neohookean law to describe the behavior of our square. In this case, the second Piola–Kirchhoff stress tensor $\underline{\underline{\Sigma}}$ can be written as

$$\underline{\underline{\Sigma}} = \frac{1}{2}\frac{E\nu}{(1+\nu)(1-2\nu)}(J^2-1)\underline{\underline{C}}^{-1} + \frac{E}{2(1+\nu)}(\underline{\underline{1}} - \underline{\underline{C}}^{-1}), \tag{38}$$

where E corresponds to the Young modulus, and ν corresponds to the Poisson ratio.

The inverse problem solved in this article consists in trying to identify the Young modulus, whose ground truth value is fixed at 1 kPa. The ground truth value of the Poisson ratio is chosen at 0.3. Note that for such a simple model, the stress tensor is linear in the Young modulus.

2.5.3. *Loading*

For the EGM, the estimation with a boundary force has been widely studied, which is not the case for body forces. Since boundary and body forces might have a different impact on the estimation—notably, the cost function differs greatly between the two cases—, we choose to consider two decoupled problems. In the first problem, the only force considered is a boundary force, applied to the right boundary of the square. This boundary force is a compression pressure $\underline{t} = -\tau\underline{n}$, with $\tau = 0.3$ kPa. In the second problem, only a body force $\underline{b} = \mu\underline{e}_x$, with $\mu = 0.3$ mN/mm^3 is applied. In both problems, the left boundary of the cube is clamped.

2.5.4. *Choice of virtual fields for the VFM*

Finding a kinematically admissible virtual field is rarely straightforward. Any virtual field chosen should indeed be smooth enough on the whole domain, but also respect the imposed displacement fields, i.e., be equal to zero anywhere a displacement is imposed. In our example, our virtual fields should be null on the left boundary of the square to respect the imposed boundary conditions. Additionally, $\underline{U}^*$ should not cancel W_{ext}, in order to avoid an estimation that would necessarily be wrong. In the scope of our problem, we studied two different virtual fields.

Plane waves as virtual fields. The first virtual field family corresponds to plane waves, written as

$$\underline{U}^* = \underline{d}\cdot\sin(\underline{k}\cdot(\underline{X}-\underline{X}_0)), \tag{39}$$

where $\underline{d}$ is the plane wave direction, chosen here equal to the wave propagation direction $\underline{e}$ for the sake of simplicity. This wave corresponds to a tension-compression wave. $\underline{k}$ represents the wave vector, and is defined as:

$$\underline{k} := \frac{2\pi}{\Delta}\underline{e}, \tag{40}$$

with Δ the period of the wave.

This choice was possible given the simplicity of our geometry. It was indeed easy to find a continuous and differentiable function, null on the left boundary. Note that this virtual field was chosen as it is easy to define, and as it includes a parameter (the frequency), which can be easily varied, and that we study in this article. However, a more simple virtual field could be defined, whose perfomances are studied in Appendix D. It is however much more difficult to find kinematically admissible displacement fields for more complex geometries.

Virtual fields obtained by solving the model. This is the reason, which motivated [8] to propose a method to build virtual displacement fields, applicable to more complex problems. This method consists in particular in building a virtual field from an intermediate position, close to the deformed position, with parameters close to the deformed configuration as well. We adapted the code provided in [38] to our problem, and used this virtual displacement field in our study for comparison with our virtual field built as a plane wave.

Note that if this method can be applied to any geometry, the construction of the virtual field requires the computation of a finite element problem (to compute the intermediate configuration). As a result, with this choice of virtual field, the VFM is not a direct method anymore. Increasing the complexity of the geometry does therefore not always allow to build directly a kinematically admissible virtual field to zero. Note also that this virtual field is suited to investigate more complex problems [8] and hence presents a complexity which is not required for our simple problem. However, it remains important to characterize its performance with respect to simpler virtual fields.

Identification in the scope of our hyperelastic problem. In the scope of the problem solved in this article, the second Piola–Kirchhoff stress tensor depends linearly on the Young modulus E. For the VFM, the estimation is direct, and consists solely in solving Equation (25): the right-hand side of the equation is divided by the left-hand side of the equation—as a reminder, the expression of $\underline{\underline{\Sigma}}$ is detailed in Equation (38). Both sides are then multiplied by E.

In the case of a boundary force problem, the Young modulus is denoted as E^{Γ}, and E^{Ω} in the case a body force problem. Based on the previous observations, E^{Γ} and E^{Ω} can directly be expressed as:

$$E^{\Gamma/\Omega} = \frac{W_{\text{ext}}^{\Gamma/\Omega}}{W_{\text{int}}(E = E_0)}, \tag{41}$$

with E_0 the ground-truth value of the Young modulus.

The last step for identifying E is to define a kinematically admissible virtual displacement field to zero.

2.5.5. *Synthetic measurements generation*

For the generation of synthetic measurements, the displacement fields are first computed with the model with the parameters' ground truth values, i.e., a Young modulus of 1 kPa and a Poisson ratio of 0.3. The noise added to the displacement field and to the images consists in a Gaussian random field with a null mean, and with SNRs of $+\infty$—i.e., no noise—, 10, 5 and 3.3 respectively. Examples of meshes associated to the noisy measurements, obtained by adding noise to the displacement fields are presented Figure 3.

As for the generation of noisy measurements, obtained by adding noise on images, the tracking is done with a regularization using a Neohookean law, with a Young modulus of 1 kPa and a Poisson ratio of 0.3. Additionally, the images are generated using the following intensity field for the image I_0:

$$I_0(\underline{X}) = \begin{cases} 0 & \text{if } \underline{X} \notin \Omega \\ \sqrt{\sin\left(\frac{\pi \underline{X}_0}{s}\right) \sin\left(\frac{\pi \underline{X}_1}{s}\right)} & \text{if } \underline{X} \in \Omega, \end{cases} \tag{42}$$

where s corresponds to the tagging period, chosen as $s = 0.1$ in the scope of our illustrative framework. This expression ensures that only the body is textured. The images created are presented in Figure 4.

For the same noise level, the deformation depends strongly on the regularization level applied. Examples of the impact of the regularization levels on the displacement field extracted from images are presented Figure 5.

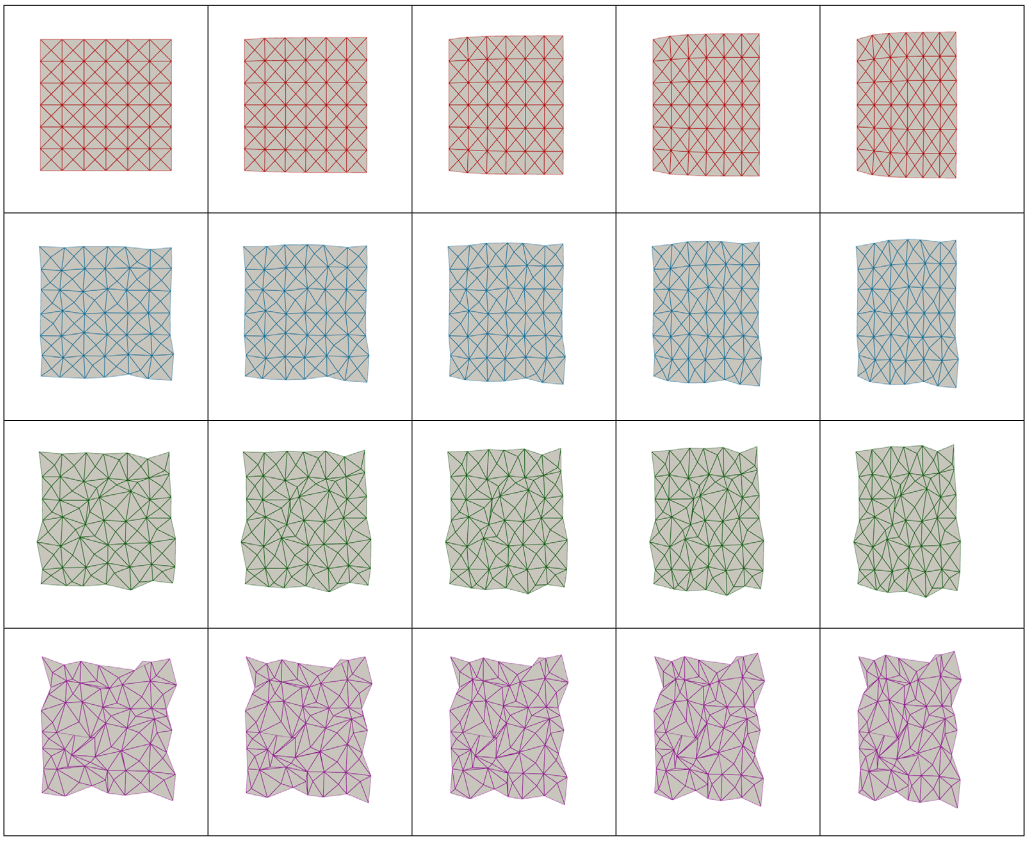


Figure 3. Deformed configuration for different SNRs—synthetic measure—: $+\infty$ (red), 10 (blue), 5 (green), 3.3 (purple), with an element size of 0.1 mm for the boundary force test, at different steps of the deformation.

2.6. *Implementation*

The results generated in this article were obtained through the use of different libraries and open-access code. In particular, the implementation of our methods relies heavily on the libraries "dolfin_mech" [39] and "dolfin_warp" [40], which are based on FEniCS [41, 42] and VTK [43].

Additionally, we studied the noise robustness of the VFM, using the virutal fields proposed by [8]. To do so, we adapted the code available in [38]. Note that this virtual field requires the computation of the problem and is hence close to FEMU in terms of computation costs.

The code used to generate the figures presented in the Results section is freely available. In particular, the different estimation methods are implemented in the library dolfin_estim [44]. The code to generate the figures is reproduced in the appendix of the paper, and is currently available interactively at https://apeyraut.gitlabpages.inria.fr/identification-methods-paper-demos.

3. Results

This section presents the estimation results for the different methods presented in Section 2. The obtained results allow us to compare the performance of our new formulation of the EGM with another direct method, the VFM with a plane wave, and two non-direct method, the VFM using a virtual field created from [8], and the FEMU method.

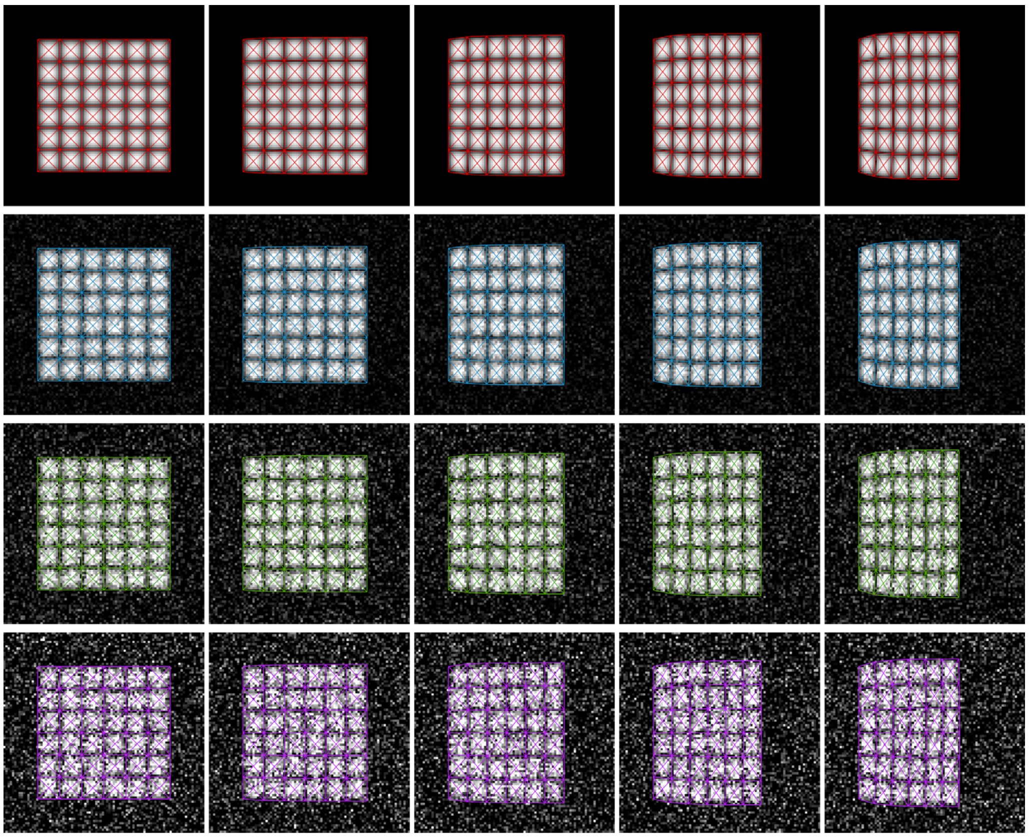


Figure 4. Images generated for different SNRs, with superimposed meshes of the ground-truth solution: $+\infty$ (red), 10 (blue), 5 (green), 3.3 (purple), at different steps of the deformation when the load applied is a boundary force.

Two different problems are considered: a boundary force problem and a body force problem. Within these two problems, two cases are also studied: identification using the synthetic measurements generated by altering directly the displacement field computed with the model, and identification using the synthetic measurements generated by adding noise to the images generated with the model. Additionally, the estimation is performed for two different mesh sizes: a mesh with characteristic element size of 0.1 mm, and a mesh with characteristic element size of 0.05 mm.

Note that for the case where the identification is conducted on the images altered with noise, different regularization levels were investigated. However, for the sake of simplicity, only two regularization levels are presented in this section. The first regularization level is 0, to illustrate the poor identification results obtained when no regularization is added. The second case corresponds to a regularization level of 0.2, for which the identification is the best for all tests performed—except for the last figure, which will be detailed later in the article.

3.1. *Estimation performance in the case of boundary forces*

3.1.1. *Error distribution on the coarser mesh*

This section compares the estimation performance of our proposed EGM with the performances of the FEMU method and the VFM, when the synthetic measurements are created by adding noise to the displacements, and by adding noise to the images—for a regularization level

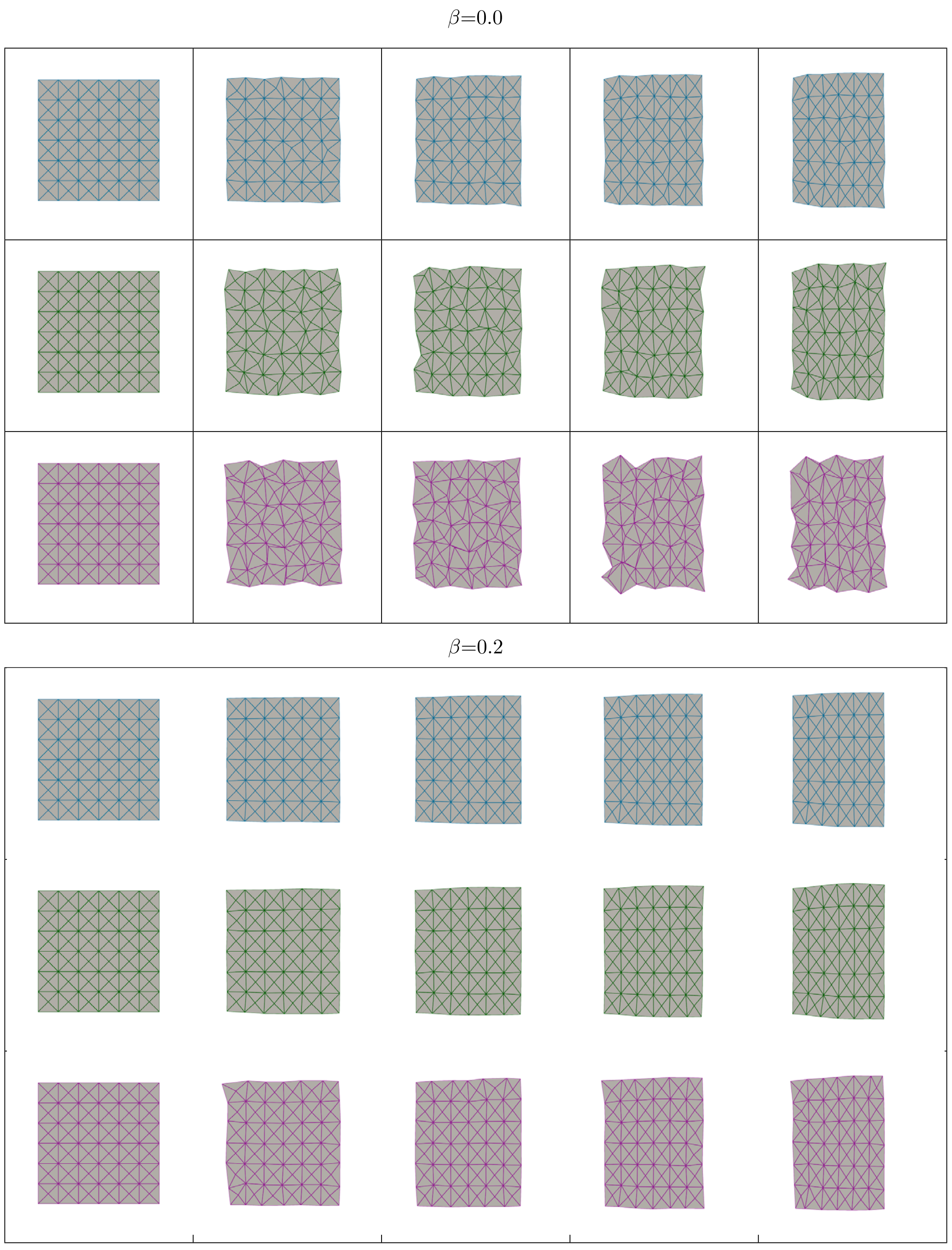


Figure 5. Tracking solutions for different regularization levels (regularization = 0 on top, regularization = 0.2 at the bottom) for SNRs of 10 (blue), 5 (green), and 3.3 (purple), at different steps of the deformation when the load applied is a boundary force.

of 0.0 and 0.2. The estimation was conducted using a mesh with a characteristic element length of 0.1 mm. The load considered here is a boundary force. The obtained results are presented in Figure 6.

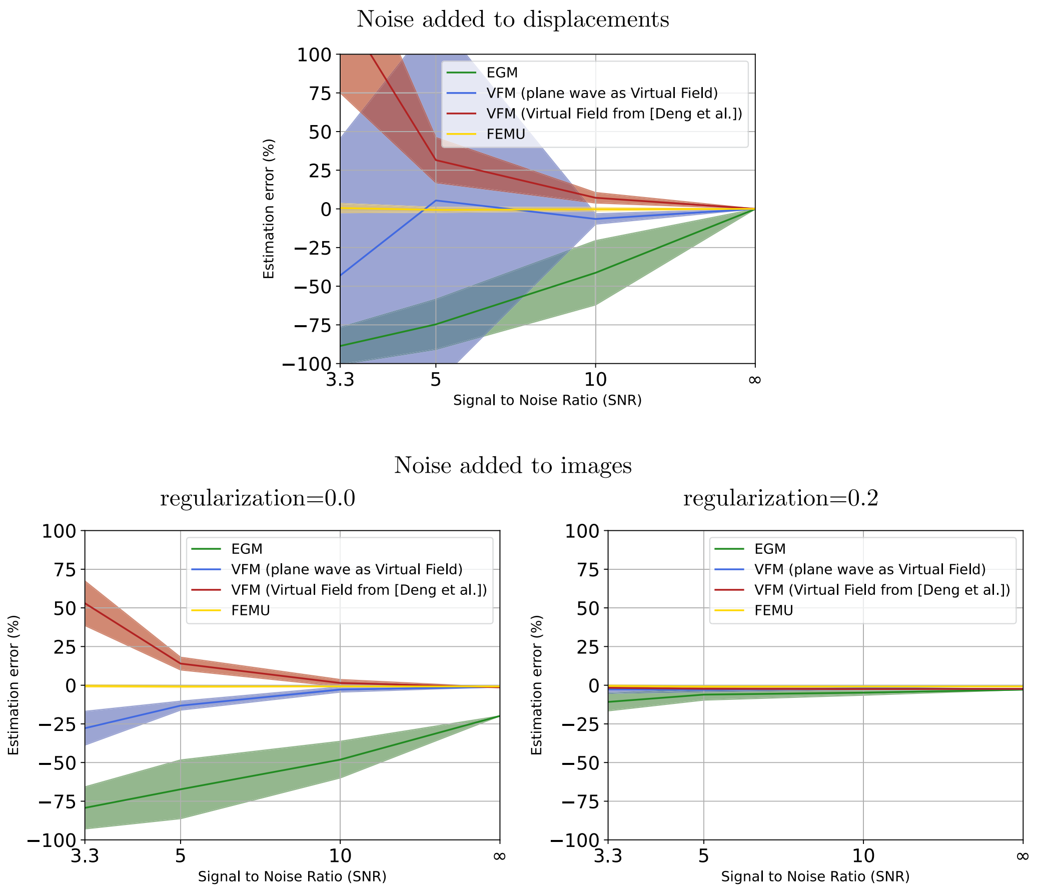


Figure 6. Estimation error for the boundary force test, with a characteristic element size of 0.1 mm. This figure presents the error distribution on the estimated Young modulus depending on the noise, for the EGM (in green), the VFM from [8] (in red), the VFM with a plane wave (in blue), and the FEMU method (in yellow). For the top row, noise is directly added to the displacement field, and for the bottom row, the synthetic measurements are generated by adding noise to the images, with a regularization of 0.0 (left) and 0.2 (right). For all methods, for noise applied to the displacements, for a SNR going to $+\infty$, the Young modulus is perfectly identified. When noise increases (smaller SNRs), the error on the estimated parameter increases as well. FEMU is the most robust method, while the EGM and the VFM are far less robust. For noise added to the images, a notable error is introduced for a SNR of $+\infty$ for the EGM. When regularization increases, the estimation drastically improves for all methods.

This figure illustrates the error committed on the estimated parameter depending on the noise. The lines and transparent areas (green for the EGM, blue for the VFM using a plane wave as the virtual field, red for the VFM using [8]'s method, and yellow for the FEMU method) correspond to the averages and the standard deviations of the distributions respectively.

Estimation using the synthetic measurements generated by adding noise to the displacements. When noise is added directly to the synthetic displacement field, the estimation quality depends strongly on the chosen method. The same—classical—pattern can however be observed for all methods: when no noise is applied to a synthetic measurement, the parameter can be perfectly identified. As noise added to the data increases—i.e., as the SNR decreases—, the quality of the estimation decreases: the average of the distribution grows further from the ground-truth value of the parameter. Additionally, the uncertainty on the parameter increases—i.e., the standard deviation is higher.

The FEMU method is the most accurate. It displays high robustness to noise, with a mean error remaining close to 0 for all noise levels, and a standard deviation remaining under 3% for the smaller SNR studied—i.e., for the highest noise. On the other hand, the EGM displays the highest noise sensitivity. If the Young modulus is perfectly identified when no noise is applied, the estimation becomes very poor when noise increases, with for example and error of about $-40\% \pm 20\%$ for a SNR of 10. Additionally, for all SNRs, the parameter is underestimated; the smaller the SNR, the higher the negative error.

Additionally, the results obtained for the VFM illustrate well the impact of the choice of the virtual field on the estimation: the distribution is very different whether the virtual field was chosen as a plane wave or based on the work of [8]. For example, for all SNRs, the Young modulus tends to be overestimated with the virtual field from [8], while it is underestimated for the plane wave. For both studied virtual fields, the estimation remains good for SNRs higher or equal to 10. However, for smaller SNRs, the error on the estimation becomes much higher.

Estimation using synthetic measurements generated by adding noise to the images. The previous results were obtained by altering displacements with noise, which lead to unstructured non-physical displacement fields. In order to quantify the estimation performance on more realistic data, we then chose to generate synthetic measurements by adding noise to images, and to retrieve the displacement fields with a tracking process, using two regularization levels: no regularization and a regularization level of 0.2. For a null regularization, the behavior of all methods is very close to what was observed in the top row of Figure 6. The main difference consists in the introduction of an error of -25% for the EGM when no noise is applied to the measurements—SNR of $+\infty$. This error is due to errors committed during the tracking process, as the estimation improves drastically when regularization increases.

For all methods, for a regularization of 0.2, the mean error of the estimation remains close to 0 for all SNRs, and the standard deviation is much smaller than for a null regularization. Note that the improvement of the estimation is marginal for the FEMU method, for which the estimation is robust for every cases studied. The estimation with the VFM improves for both virtual fields. The EGM remains the less accurate method. However, the larger error, around $-10\% \pm 8\%$, is obtained for very small SNRs, and the error is much smaller for experimental values of SNRs (around 10 for CT-scans, for example). The EGM could therefore be used for inverse problems with our new formulation, since it shows a very good robustness to noise, as long as the Equilibrium Gap regularization is also used for the tracking.

3.1.2. *Error distribution on the finer mesh*

The results presented in the previous section were obtained on a rather coarse mesh, for which estimation is known to perform well. However, it can be expected to perform more poorly when the mesh is refined [12]. We therefore also chose to conduct our study on a finer mesh, in order to be closer to applications on real problems. The estimation is performed on a mesh with a characteristic element size of 0.05 mm. The results are presented in Figure 7.

Estimation using synthetic measurements generated by adding noise to the displacements. Similarly to previous observations, when noise is added directly to the synthetic displacement field, the quality of the estimation depends strongly on the method chosen. The estimation remains fairly stable compared to the estimation performed on the coarser mesh: there is no

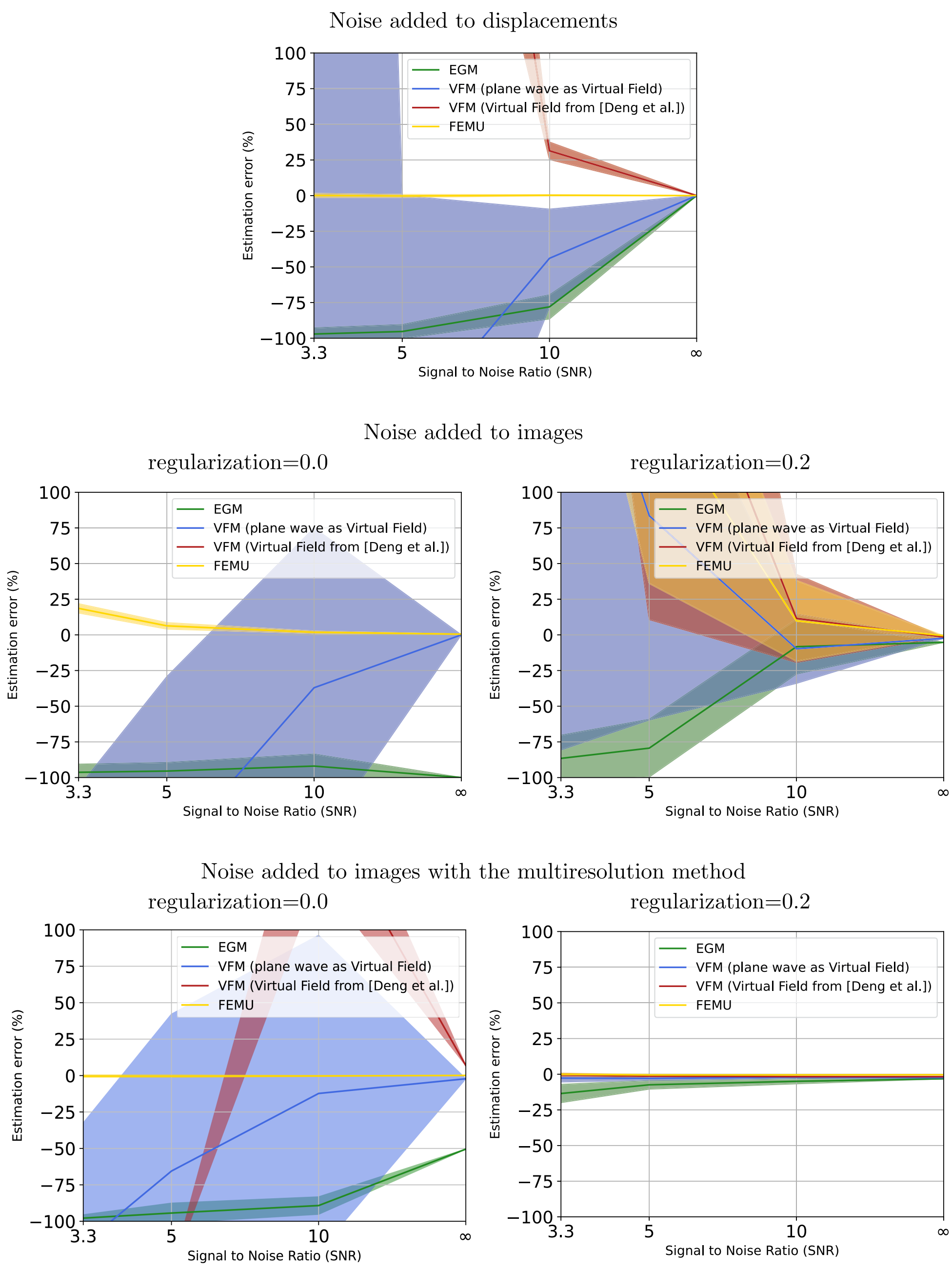


Figure 7. Estimation error for the boundary force test, with a characteristic element size of 0.05 mm. This figure presents the error distribution on the estimated Young modulus depending on the noise, for the EGM (in green), the VFM from [8] (in red), the VFM with a plane wave (in blue), and the FEMU method (in yellow). Three cases are investigated: (i) noise added to the synthetic displacement field (top row), (ii) noise added to images (middle row), (iii) noise added to images, refined using the multiresolution. If the estimation quality is very poor for the first two cases, the estimation improves drastically with the multiresolution.

notable difference for the distribution obtained with the FEMU method, and the global behavior of all direct methods remains stable, even thoug their estimation quality is far poorer. For example, for the VFM, the average identified Young modulus deviates from its ground truth value more rapidly than with the coarser mesh with both chosen virtual fields: for a SNR of 10, the average error for the VFM with a plane wave as virtual field is −6% with the coarser mesh, while it is around −20% with the finer mesh. The estimation is also far better with the coarser mesh for the EGM. The average error for a SNR of 10 is around −40% in that case, while it is almost equal to −90% for the finer mesh. If refining the mesh does not seem to impact the estimation quality with the FEMU method, it seems to deteriorate greatly the estimation with the direct methods.

Estimation using synthetic measurements generated by adding noise to the images. The estimation was performed for measurements obtained with a null regularization and a regularization of 0.2. For a null regularization, the FEMU method and the VFM which uses a plane wave as virtual field give good results when no noise is added to the data. However, the estimation quality deteriorates rapidly as noise increases, especially for the VFM. For the FEMU method, the estimation error also increases with noise, although the standard deviation of the distribution remains fairly small. For the VFM from [8], the estimation did not converge, even for a SNR of $+\infty$, due to tracking errors. For the EGM, when no noise is applied, an initial error of −100% can be observed, due to tracking errors. This error is much higher than for the coarser mesh and deteriorates when the SNR decreases.

When the regularization increases, the estimation improves for all direct methods. In particular, for the EGM, the error observed for a SNR of $+\infty$ is corrected. The same behavior can be observed for the VFM from [8], for which the parameter becomes identifiable when no noise is added to the images. For the VFM with a plane wave as virtual field, the estimation improves slightly, with a smaller standard deviation for small SNRs, and an average error closer to 0. However, the quality of the estimation becomes poorer when noise increases for all these cases. Although the estimation is better for direct methods, it becomes far less accurate for the FEMU method. If the Young modulus is still identifiable when no noise is added to the images, the estimation becomes less accurate when noise increases: the average error increases, as well as the standard deviation. Overall refining the mesh leads to a poorer estimation and does therefore not seem to be a good option for all estimation methods, with both synthetic data generation methods.

Estimation based on tracking using the multiresolution method. The decrease in accuracy of the estimation when using synthetic measurements with a refined mesh could have been anticipated, as using a finer mesh for the tracking process necessarily leads to a increased noise sensitivity of the solution. While mechanical regularization is designed to prevent such phenomenon, it may not be sufficient. Consequently, the multiresolution method, which consists in applying the tracking process on successively refined meshes, helps to avoid this problem. To perform the tracking on a finer mesh, the solution at a given frame is initialized by the solution of the tracking on a coarser mesh, at the same frame. The results for the estimation using synthetic measurements generated by adding noise to images on a finer mesh obtained with the multiresolution method, are presented in Figure 7, on the bottom row.

The multiresolution improves slightly the results for a null regularization, compared to the results obtained when performing the tracking directly with the refined mesh. The FEMU method displays the best results: its average is close to the ground truth value, and the standard deviation is small for all SNRs. The initial error for a SNR of $+\infty$ is also smaller for the EGM—even though it is still high. As noise increases, the average error is also high for this method, even though the standard deviation remains fairly small. Similarly, the estimation is of good quality for the VFM, when no noise is added to the images, but deteriorates as noise increases.

Increasing the regularization, together with the multiresolution, however improves greatly the estimation, leading to error distributions close to the results found with the coarser mesh. The FEMU method and the VFM display the same behavior: for all SNRs, the average of the distribution is close to the ground-truth value and the standard deviation is very small. A very good robustness to noise is observed for all methods. As before, the EGM presents the highest sensitivity to noise. However, the error committed during the estimation is highest for a SNR of 3.3 (12% ± 8%) and remains reasonable. As a result, the EGM can also be considered as a robust estimation method for finer meshes, should the multiresolution been used in addition to the Equilibrium Gap regularization.

3.2. *Estimation performance in the case of body forces*

The previous section focused on presenting results allowing the comparison of our new formulation of the EGM with the VFM and the FEMU method, for a boundary force test. As the form of the boundary force term and of the body force term differ greatly in the loss function, we would also like to evaluate the perfomance of our new formulation of the EGM for a body force test, which is presented in this section.

3.2.1. *Error distribution on the coarser mesh*

This section presents the error distribution on the identified Young modulus on a mesh with a characteristic element size of 0.1 mm, for the four methods, with a synthetic measurement created by adding noise to the displacements, and with a synthetic measurement created by adding noise to the images—for a regularization level of 0.0 and 0.2. The only load studied in this section is a body force. The obtained results are presented in Figure 8. Note that in this section, we compare the results obtained with the boundary force test and with the body force test, even though the two displacement fields are different. Indeed, their norm are of the same order (0.06 for the boundary test and 0.02 for the body force test). Additionally, the noise added to images is proportional to the displacement norm. As such, the results obtained with the body and the boundary force tests can be quantitatively compared.

Estimation using synthetic measurements generated by adding noise to the displacements. The sensitivity to noise is very similar to what was observed for the boundary force test, for all methods except for the EGM. However, for the VFM, if the global behavior of the distribution remains similar, the error is however globally smaller for the body force test than for the boundary force test (average closer to the ground-truth value and smaller standard deviation). On the contrary, the estimation of the Young modulus using the EGM method is poorer with the body force than for the boundary force test. If the parameter can indeed be identified perfectly when no noise is applied to the displacement field, the error becomes very high very fast as noise increases and the Young modulus is systematically underestimated.

Estimation using the synthetic measurements generated by adding noise to the images. Two cases are investigated: the estimation is first conducted with a null regularization, and then with a regularization of 0.2. For a null regularization, the behavior of all methods is very close to what was observed for synthetic measurements built by adding noise to the displacement field. The main difference with the previous case consists in the introduction of an error of -90% for the EGM and for a SNR of $+\infty$ due to tracking errors. Note that this error is much higher than for the boundary force test. When the regularization increases, the estimation improves drastically. As before, the VFM displays a behavior very close to the FEMU method. As for the EGM, the initial error is corrected, and the method seems far less sensitive to noise.

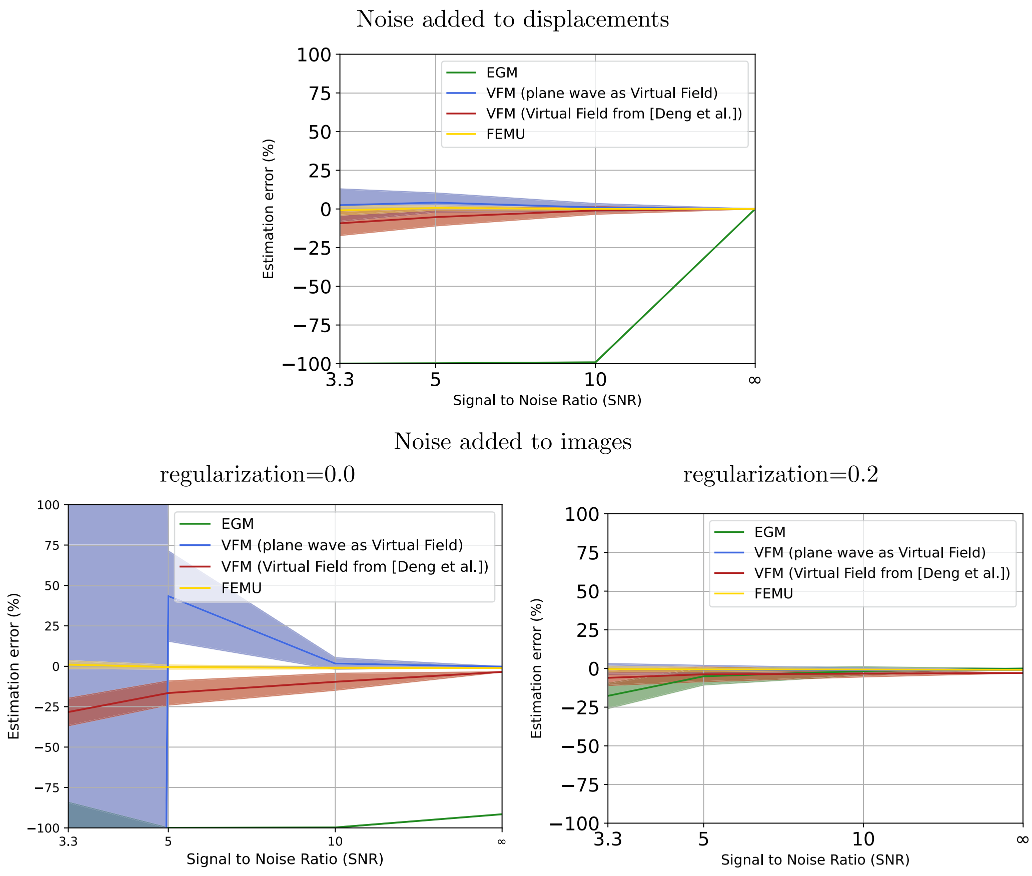


Figure 8. Estimation error for the body force test, with a characteristic element size of 0.1 mm. This figure presents the error distribution on the estimated Young modulus depending on the noise, for the EGM (in green), the VFM from [8] (in red), the VFM with a plane wave (in blue), and the FEMU method (in yellow), for synthetic measurements generated by adding noise to the displacements (top row), and synthetic measurements generated by adding noise to images (bottom row), for a regularization level of 0.0 (left) and 0.2 (right). The estimation with the body force test is globally better than with the boundary force test, for all methods except the EGM, when noise is added to the displacement field. When adding noise to the images, the results with no regularization are very close to the results generated by adding noise to the displacements. Adding regularization improves the estimation for all methods, and in particular for the EGM.

3.2.2. *Error distribution on the finer mesh*

As for the boundary force test, we chose to conduct the estimation on a refined mesh. Knowing that the results would be poor if the estimation was performed directly on a finer mesh, we chose to only present the results obtained on a finer mesh generated with the multiresolution method. The results obtained are presented in Figure 9. Note that contrarily to previous cases, the regularization added for this case is 0.8. A regularization of 0.2 indeed did not lead to a good estimation quality. Hence, the regularization strength had to be increased for better results.

When no regularization is applied during the tracking process, the estimation is of very poor quality for all methods except FEMU, but improves when regularization increases. In particular,

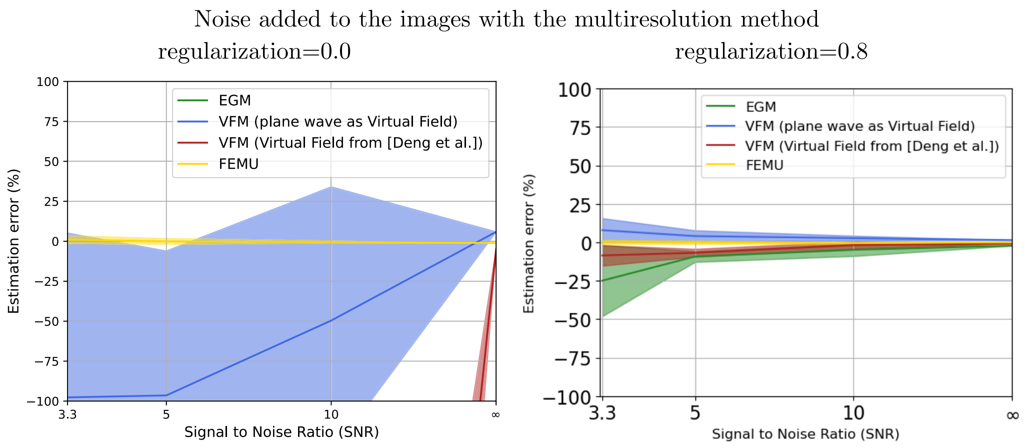


Figure 9. Error distribution on the estimated Young modulus depending on the noise for the body force test, with an element size of 0.5 mm obtained with the multiresolution method, for the EGM (in green), the VFM from [8] (in red), the VFM with plane waves (in green), and the FEMU method (in yellow). The synthetic measurement used for the estimation was generated by adding noise to images, with a regularization of 0 (left) and 0.8 (right). When no regularization is added to images, the estimation is very good for the FEMU method but very poor for all direct methods. The estimation improves for all direct methods when regularization increases.

for a null regularization, the mean and standard deviation for the EGM do not even appear on the graph because the error committed during the estimation process is too high. The estimation quality also improves drastically with regularization for the EGM. For SNRs larger than 5, the standard deviation remains small, and the average error close to 0. For a very small SNR—3.3—, the average error is however high, and the standard deviation large: the parameter is identified with an error of −25% ± 25%. A SNR of 3.3 however corresponds to very noisy data, and is out of the scope of many mechanical applications. Additionally, for small SNRs, the estimation quality for the EGM is slightly poorer for the body force compared to the boundary force test (the standard deviation is wider).

3.3. *Impact of the choice of virtual field for the VFM*

The presented results illustrate that the estimation quality with the Virtual Fields Method (VFM) varies significantly depending on the virtual field selected for the identification. Our results indeed indicate that the error distribution of the identified Young modulus differs depending on whether the virtual field was chosen as a plane wave or constructed based on the method described in [8]. For most tests and synthetic measurements considered, both virtual fields indeed tend to have a different impact on the estimation. For example, if the parameter is underestimated with the plane wave, it is often overestimated with the virtual field built from [8], and conversely.

This variation in estimation quality is also noticeable with the virtual field chosen as a plane wave, when modifying the value of Δ, which is a constant chosen empirically, and represents the period of the wave, which is directly linked to the plane wave frequency, as presented in Equation (39). The error distribution on the estimated Young modulus with the virtual field chosen as a plane wave is presented in Figure 10 for different values of Δ, depending on the

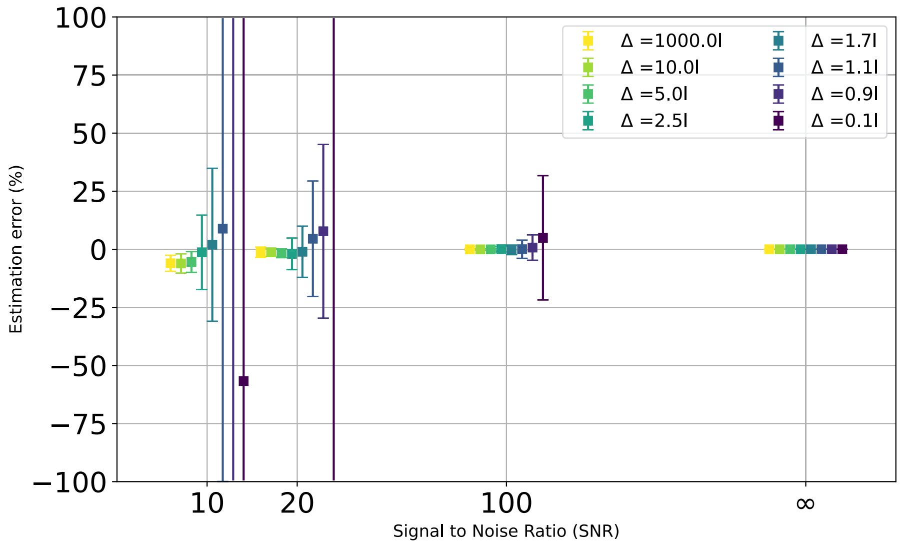


Figure 10. Error distribution for the Young modulus identified with the VFM, with a plane wave as virtual field, for different values of the period Δ—ranging from 0.1l to 1000.0l, with l the characteristic length of the cube—, for a boundary force test and with noise directly applied to the displacement field. The results are presented for SNRs ranging from 10 to +∞. When no noise is applied to the synthetic measure, all values of Δ lead to a perfect estimation of the Young modulus. As noise increases, smaller values of Δ lead very large estimation error.

SNR. The error distribution was investigated for measurements created by adding noise to the displacements, for a boundary force test. Since the Young modulus was not identifiable in that case for SNRs higher than 10, as presented in Figure 6, the study was conducted for SNRs ranging from 10 to $+\infty$. The values of Δ studied range from $0.01l$ to $1000.1l$, where l represents the characteristic length of the square. Note that the higher Δ, the smaller the plane wave frequency.

Figure 10 presents the error distribution of the estimated Young modulus for different values of Δ, to which different colors are associated. The highest value of Δ appears in yellow, while the smallest value of Δ is in blue. The error distribution is represented with error bars: the mean value of the distribution is represented as a square, while the bars represent the mean $\pm$ the standard deviation of the distribution. For a SNR of $+\infty$, the Young modulus is perfectly identified for all values of Δ. For a SNR of 100, i.e., a very small level of noise, the parameter is still identifiable with all Δ, except for the smallest value ($\Delta = 0.1l$). As noise increases, the smaller values of Δ lead to large errors on the identified parameter. For example, for a SNR of 20, which is still fairly small, the error on the identified parameter is high for all Δ greater than $1.7l$, i.e., the standard deviation is greater than 25%.

However, the estimation quality decreases for high values of Δ when noise increases. For example, the estimation is better for $\Delta = 2.5l$ than for $\Delta = 1000l$ for a SNR of 10, as the average of the distribution is not centered in zero for the larger value of Δ. As Δ increases, the plane wave indeed becomes flatter—i.e., the oscillations of the plane wave are barely noticeable— and the impact of the noise is much higher. Choosing the value of Δ hence consists in finding a good trade-off between precision and robustness.

Additional studies, which are not directly related to the main topic of this article but present interesting results regarding the limitations of our method, are included in the appendices. In particular, the robustness to model errors of the different methods is studied. Appendix A presents the impact of model errors on the Poisson coefficient on the estimated Young modulus, while Appendix B presents the impact of the value of the volume regularization on the estimation.

4. Discussion

The results obtained have highlighted that our new formulation of the EGM leads to very different results depending on the method of synthetic data generation, the mesh size, the noise level, and the type of applied load. Generally, the VFM leads to better results, but seems to be very dependent on the chosen virtual field. Lastly, the FEMU method performs better than the two other investigated methods. These results can be explained by observing the analytical forms of the cost functions of the methods.

4.1. *Estimation performance of the EGM*

For the sake of simplicity, we study here the cost function of the EGM in small elastic strains. The presented work can easily be extended to more complex problems. Let us first write the finite element discretization of the EGM cost function presented in Equation (26):

$$L^{\text{EGM}} = \left(\underline{\underline{\mathbb{D}}}\,\underline{\underline{\mathbb{K}}}\,\underline{\mathbb{U}}_{\text{meas}} - \underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}\right)^{\text{T}} \underline{\underline{\mathbb{M}}} \left(\underline{\underline{\mathbb{D}}}\,\underline{\underline{\mathbb{K}}}\,\underline{\mathbb{U}}_{\text{meas}} - \underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}\right). \tag{43}$$

In linear elasticity, the stiffness matrix $\underline{\underline{\mathbb{K}}}$ depends linearly on the Young modulus E. Let us thus define a matrix $\underline{\underline{\tilde{\mathbb{K}}}}$, defined as

$$\underline{\underline{\tilde{\mathbb{K}}}} = \frac{1}{E}\underline{\underline{\mathbb{K}}}. \tag{44}$$

Injecting (44) into (43) hence yields the following expression of the cost function:

$$L^{\text{EGM}} = \left(E\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\,\underline{\mathbb{U}}_{\text{meas}} - \underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}\right)^{\text{T}} \underline{\underline{\mathbb{M}}} \left(E\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\,\underline{\mathbb{U}}_{\text{meas}} - \underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}\right). \tag{45}$$

The estimation consists in finding the parameter E minimizing the cost function, i.e., the parameter E verifying

$$\frac{\partial L^{\text{EGM}}}{\partial E} = 0, \tag{46}$$

which leads to the following expression of E:

$$E = \frac{1}{2}\,\frac{\underline{\mathbb{F}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\,\underline{\mathbb{U}}_{\text{meas}} + \underline{\mathbb{U}}_{\text{meas}}^{\text{T}}\underline{\underline{\tilde{\mathbb{K}}}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}}{\underline{\mathbb{U}}_{\text{meas}}^{\text{T}}\underline{\underline{\tilde{\mathbb{K}}}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\,\underline{\mathbb{U}}_{\text{meas}}}. \tag{47}$$

Let us first investigate the case where the synthetic measurement is generated by adding noise to the displacements. $\underline{\mathbb{U}}_{\text{meas}}$ therefore corresponds to the approximated measured—and noisy—displacement field and can be expressed as

$$\underline{\mathbb{U}}_{\text{meas}} = \underline{\mathbb{U}}_0 + \underline{\mathbb{G}}, \tag{48}$$

where $\underline{\mathbb{G}}$ represents the noise added to the approximated reference—i.e., associated to $\underline{\theta}_0$ and $\underline{\eta}_0$—displacement field $\underline{\mathbb{U}}_0$. Injecting Equation (48) into Equation (47) therefore gives the following expression for the Young modulus:

$$E = \frac{1}{2}\,\frac{\underline{\mathbb{F}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\left(\underline{\mathbb{U}}_0 + \underline{\mathbb{G}}\right) - \left(\underline{\mathbb{U}}_0 + \underline{\mathbb{G}}\right)^{\text{T}}\underline{\underline{\tilde{\mathbb{K}}}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\mathbb{F}}}{\left(\underline{\mathbb{U}}_0 + \underline{\mathbb{G}}\right)^{\text{T}}\underline{\underline{\tilde{\mathbb{K}}}}^{\text{T}}\underline{\underline{\mathbb{D}}}^{\text{T}}\underline{\underline{\mathbb{M}}}\,\underline{\underline{\mathbb{D}}}\,\underline{\underline{\tilde{\mathbb{K}}}}\left(\underline{\mathbb{U}}_0 + \underline{\mathbb{G}}\right)}. \tag{49}$$

According to Equation (49), the Young modulus depends linearly on the noise on the numerator and quadratically on the denominator. As a result, the more the noise increases, the more the

Young modulus is underestimated, as the quadratic dependency of the denominator amplifies the impact of the high frequencies of noise on the estimation. When no noise is applied to the displacements, the parameter should moreover be identified perfectly. This behavior is consistent with the results obtained.

This is especially true when the noise is added directly to the displacements, resulting in highly unstructured measurements. As a result, this method for data generation, even though a classical approach for performance quantification in inverse problems, is non-physical. Using our second generation method, which consists in adding noise to images, would be a more consistent alternative.

When noise is added directly to the images without regularization, the generated measurement is similar to the synthetic measurement generated by adding noise to displacements: there is no additional term enforcing a physical behavior during the tracking process, and the noise is hence highly unstructured. Additionally, an initial error appears for a SNR of $+\infty$, due to tracking errors: the EGM being very sensitive to any small variation of the displacement field, small tracking errors can indeed lead to significant errors in the estimation. This initial error is moreover greater for the body force test than for the boundary test, as only the boundary where the pressure is applied is considered for the boundary force test, whereas the entire domain is taken into account for the body force test. As a result, more tracking errors are included in the body force test, leading to a greater initial error.

Increasing the mechanical regularization however allows to correct this initial error, smooths the form of the noise, and improves drastically the estimation for all noise levels. This result is also consistent with [12], who observed that the regularization allowed to decrease the error on the displacement field obtained with the tracking process for all noise levels. This result is consistent with the form of Equation (47) as well. Noise is indeed included in $\underline{\mathbb{U}}_{\text{meas}}$, but corrected during the tracking. As a result, it is more structured and hence leads to a better estimation. Note that the estimation for the boundary force test is slightly better than for the body force test, when regularization is added during the tracking process, as well as when noise is added directly to displacements, for the same reasons as those previously mentioned.

Overall, our new formulation of the EGM provides a less accurate estimation than the other methods studied for comparison. This finding aligns with classical behaviors documented in the literature [4], which suggest that direct methods typically yield significantly poorer results compared to indirect methods, such as the FEMU method, although it is significantly less expensive. Furthermore, a carefully selected virtual field for the VFM is expected to display a better performance than the EGM, as the latter considers all virtual fields, including the suboptimal ones. However, defining a good virtual field might be challenging for complex geometries. Additionally, the EGM is less accurate than other methods only when noise is added to the displacement field: it is not the case anymore when noise is added to images for a good regularization choice. The EGM can hence be very well adapted for the estimation, depending on the desired level of accuracy. Even when higher accuracy is wanted, the EGM may be very useful, insofar as it can be used to initialize the estimation and therefore improve and fasten the identification process.

4.2. *Estimation performance of the VFM*

The results presented in this article showed that the estimation with the VFM leads a better accuracy and noise robustness than the EGM. This is consistent with the fact that well-chosen virtual fields should lead to a better estimation quality than the EGM, which includes all possible virtual fields, even the ones leading to a poor estimation quality. The noise sensitivity however differs for both studied virtual fields—the plane wave and the virtual field constructed from [8].

This difference is noticeable for both boundary force and body force tests, and is in accordance with the literature: [17] has indeed shown that all virtual fields do not display the same sensitivity to noise. If new approaches have been designed to create virtual fields more robust to noise [19, 45], there does not exist a general method to create an optimal virtual field—i.e., the virtual field, which displays the highest robustness to noise—yet.

The variability in accuracy was for example illustrated in this work when the virtual field was chosen as a plane wave, by varying the frequency. This variation of accuracy depending on the frequency of the plane wave can be explained by studying the cost function of the method. As for the EGM, since the generalization is straightforward—the expression of the identified Young modulus is obtained directly from Equation (41)—, let us conduct the analysis for linearized quantities without loss of generality. Additionally, for illustrative purposes, let us study the expression of the Young modulus for a boundary force—which can easily be replaced by the body force. In that case, we have shown that

$$E^{\Gamma} = \frac{W_{\text{ext}}^{\Gamma}(\underline{U}^*)}{W_{\text{int}}(\underline{U}_{\text{meas}};E=1,\underline{U}^*)} = \frac{\displaystyle\int_{\Gamma} \tau\underline{n}\cdot\underline{U}^*\,\mathrm{d}\Gamma}{\displaystyle\int_{\Omega} \underline{\underline{\sigma}}(\underline{U}_{\text{meas}};E=1):(\underline{\underline{\nabla}}\,\underline{U}^*)_{\text{sym}}\,\mathrm{d}\Omega}. \tag{50}$$

Both numerator and denominator depend linearly on $\underline{U}^*$. In order to find E^{Γ}, an admissible virtual field $\underline{U}^*$ needs to be defined. As reminded in Equation (39), the family of virtual fields chosen as a plane wave can be written $\underline{U}^* = \sin(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}$. As a result, the gradient of $\underline{U}^*$ is:

$$\underline{\underline{\nabla}}\,\underline{U}^* = |\underline{k}|\cos(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}\otimes\underline{e}. \tag{51}$$

Note that in order to be well-defined, our virtual field should not cancel W_{ext} at the right boundary. As a result, Δ (appearing through the relation $|k| = 2\pi/\Delta$) should not be of the form α/l, where l is the characteristic length of the cube, and α an integer. Based on the previous equation, $|\underline{k}|$ can be factorized at the denominator, such that:

$$E^{\Gamma} = \frac{\displaystyle\int_{\Gamma} \tau\underline{n}\cdot\sin(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}\,\mathrm{d}\Gamma}{|\underline{k}|\displaystyle\int_{\Omega} \underline{\underline{\sigma}}(\underline{U}_{\text{meas}};E=1):\cos(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}\otimes\underline{e}\,\mathrm{d}\Omega}. \tag{52}$$

By observing Equation (52), we can note that noise is present at the denominator, where $1/\Delta$ is a factor. Noise is indeed included in the term $\underline{\underline{\sigma}}(\underline{U}_{\text{meas}};E=1)$, through $\underline{U}_{\text{meas}}$, which depends on the noise $\underline{g}$ through the relation:

$$\underline{g} = \underline{U}_{\text{meas}} - \underline{U}_0, \tag{53}$$

where $\underline{U}_0$ is the measurement without noise. As a result, Equation (52) can also be written:

$$E^{\Gamma} = \frac{\displaystyle\int_{\Gamma} \tau\underline{n}\cdot\sin(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}\,\mathrm{d}\gamma}{|\underline{k}|\displaystyle\int_{\Omega} \underline{\underline{\sigma}}(\underline{U}_0+\underline{g};E=1):\cos(\underline{k}\cdot(\underline{X}-\underline{X}_0))\underline{e}\otimes\underline{e}\,\mathrm{d}\Omega}. \tag{54}$$

For linear elasticity, $\underline{\underline{\sigma}}(\underline{U}_0+\underline{g};E=1)$ depends in reality linearly on the first derivative of $\underline{U}_{\text{meas}}$, and hence linearly on the first derivative of $\underline{U}_0+\underline{g}$. It is therefore clear that the impact of noise will increase as the value of Δ decreases. This phenomenon will in particular be noticeable for unstructured noise, i.e., random Gaussian noise added directly to the displacement field. However, in general cases, and contrarily to the EGM, it is impossible to predict whether the parameter will be overestimated or underestimated with the VFM, with a virtual field chosen as a plane wave. In large strains, the external work indeed depends on the measured—noisy—displacement field. For example, in the case considered here, $W_{\text{ext}}^{\Gamma} = \int_{\Gamma} T\underline{\underline{F}}^{-T}\cdot\underline{N}\cdot\underline{U}^*J\,\mathrm{d}\Gamma$ (in large strains) depends on the noisy displacement trough $\underline{\underline{F}}^{-T}$ and J. The expression of the Young modulus therefore depends on the noise through its numerator and denominator. It is hence not straightforward to predict the behavior of the estimation.

The estimation quality improves greatly for the VFM when noise is added to images—for a good regularization—compared to the case where synthetic data is generated by adding noise to displacements, due to the smoothing of noise with the second data generation method. Moreover, and contrarily to the EGM, the estimation seems more robust with the VFM when a body force is applied than for the boundary force test. We assume it is because less noise is considered at the numerator for the body force test—only included in J— than for the boundary test—included in J and in $\underline{\underline{F}}^{-T}$—, as detailed in the Methods section.

Overall, the VFM seems to give better results than the EGM. However, it was possible to define the plane wave as virtual field only given the simplicity of the considered geometry. Choosing a good kinematically admissible virtual field for more complex problems is rarely as straightforward. Methods have been proposed to overcome this issue, such as the one studied in this article, proposed by [8]. If this particular method allows to build a kinematically admissible displacement field, with good estimation performances, it however transforms the VFM in a indirect method, i.e., the method is not entirely direct anymore as it requires finite element computations for given steps of the estimation. As a result, if our new formulation of the EGM leads to a lower estimation quality, it may still be a better option than the VFM for computationally expensive problems.

4.3. *Estimation performance of the FEMU method*

In general, the noise robustness of indirect methods is higher than for direct methods. These methods are however very expensive, since they require finite element computations at every step of the optimization. The robustness to noise of the FEMU method was studied here for comparison. The FEMU method is indeed a direct method known to be very robust to noise [4], which is what is observed for all cases of our study. The only case where the estimation is not good is when noise is added to images, for a fine mesh generated without the multiresolution method. As explained in the Results section, [12] noticed this phenomenon in his study. The finer the mesh, the bigger the function space for the solution, which can lead to ill displacement fields solutions. The problem is however solved when using the multiresolution method.

Using measurements generated by adding noise to the displacements or to the images does not seem to significantly impact the estimation quality of the FEMU method. We assume this is due to the form of the cost function: unlike the direct methods studied here, which include first derivatives of the noise in their cost functions, and are therefore strongly impacted when the noise is unstructured, the FEMU method only aims to find the Young modulus that leads to a computed displacement field, which best matches the noisy measurements. Consequently, even when noise is added directly to the displacement field, the latter will not be able to fit exactly the unstructured data, and the solution displacement field matching best the data will automatically "average" the noise, which limits its impact on the estimation.

The robustness to noise of the FEMU method is therefore much higher than for the VFM and the EGM, and should be preferred when high accuracy is needed. This method might however be too expensive. In this case, our new formulation of the EGM may be a good alternative. It indeed displays a good noise robustness and presents the advantage of being a direct method, and is therefore far less expensive than the FEMU method.

4.4. *Comparison of the estimation performance of the three methods*

The choice of an estimation method is rarely simple. It indeed usually requires a good trade-off between accuracy and computation time. For example, the FEMU method represents the best option in terms of accuracy and robustness to noise and model errors. This method is however very expensive, since it requires a finite element computation at each iteration. Consequently, for complex problems requiring many inverse problems, the FEMU method will induce very high computation times and might not be adapted. In this case, direct methods such as the VFM and the EGM may be better options.

Based on the results presented in this article, the VFM seems at first sight to be the best option. The estimation with this method indeed leads to more accurate results: in general, the estimated Young modulus is closer to the ground-truth value, and is identified with less uncertainty than with the EGM. However, the results presented with the VFM were obtained for two well-chosen virtual fields. The choice of a plane wave as virtual field was only possible given the simplicity of the geometry. Defining a kinematically admissible displacement field is however much more difficult for more complex geometries. In order to build a good virtual field in such cases, the method proposed by [8] is particularly appropriate but is (as mentionned before) not a fully direct method anymore. Even if less accurate than the VFM for a well-chosen virtual field, the EGM is therefore a good option, since building a good virtual field for a complex problem is not straightforward or might be expensive.

Note however that the robustness to noise of the EGM is very low when the synthetic measurements used for the estimation are generated by adding noise directly to the displacement field, but improves greatly when noise is added to images, and the displacement field retrieved with tracking associated to the Equilibrium Gap regularization. Other methods, such as filtering high noise frequencies or generating the random Gaussian noise on a coarser mesh and projecting it on the finer mesh, could be considered to create synthetic measurements with smooth and physical noise. We however preferred our method of generating noisy images, because of its closeness to real data processing.

When using the second method of synthetic data generation, the estimation with the EGM becomes very good. Until high noise levels, the average of the estimated Young modulus distribution is close to the ground truth value, and the standard deviation is relatively small. However, the EGM remains overall less accurate than the FEMU method, which may be a better option when high accuracy is needed, even though it is very expensive.

5. Conclusion

When full-field measurements are available for inverse problems in large hyperelastic strains, different estimation methods can be considered. All methods present advantages and drawbacks in terms of accuracy and in terms of noise and/or model errors robustness. Four different methods were investigated in this article: the FEMU method, very robust even though quite expensive, the VFM, with two different virtual fields—a direct method, with virtual fields chosen as plane waves, and an indirect method, with virtual fields from [8]—and the EGM, which is less robust but also less expensive than the indirect methods. Note that no general formulation of the EGM has been proposed in large hyperelastic strains. The originality of our work hence lies in a new formulation of the EGM in large hyperelastic strains (which is directly based on a recently proposed continuous formulation and consistent discretization of the EG principle [12]), and its performance quantification, by comparing it with the other three aforementioned methods for various noise levels.

The quantitative study of the impact of the robustness to noise of each method relies on the generation of synthetic data. Two methods for generating such data are investigated here. The first method is based on a classical approach in uncertainty quantification: a random Gaussian noise is added to a displacement field computed with the model using reference parameters. The second method consists also to first compute a displacement field with the model with parameters chosen at reference values. Images of the reference and deformed configurations are then created, to which a random Gaussian noise is added. The displacement field between the two configurations is finally retrieved with a tracking process, to which mechanical regularization is added. The second method therefore allows to generate noisy displacement fields closer to experimental data, insofar as the noise produced with the first method is highly unstructured.

The results obtained allow to quantitatively study the estimation quality of each method, and in particular of our new formulation of the EGM. As expected, the FEMU method gives very good results: for all cases studied, this method displays a very high robustness to noise. The VFM is on contrary less accurate, but far less expensive when using plane waves as virtual fields. Additionally, this method is in average more accurate than the EGM.

The VFM however depends strongly on the chosen virtual field. It indeed requires the choice of a “good” kinematically admissible virtual field, which did not represent any difficulty in this article given the simplicity of the geometry considered, but might be less straightforward for more complex problems. General methods, such as the one presented in [8], can be developed to find “good” virtual fields for more complex geometries. These methods however require model computations for building the virtual fields, which transforms the VFM into a more expensive method.

The EGM therefore represents a good alternative for the estimation process. As expected, the estimation with the EGM is less accurate than with the VFM, since it takes into account all kinematically admissible virtual fields, including the suboptimal ones. Additionally, the EGM is not robust to noise at all when the synthetic measurements are generated by adding noise to displacements. However, when the synthetic data is generated by adding noise to images, with tracking preformed using the Equilibrium Gap mechanical regularization, the estimation quality improves drastically. Therefore, the EGM could be a good choice for complex estimation problems, even involving high noise levels.

Note that only the EGM, the VFM and the FEMU method have been investigated in the scope of this article. However, other methods can also offer a good compromise between robustness and cost. For example, the Reconditioned EGM [4, 10] could be an alternative, with higher noise robustness than the VFM and the raw EGM, but smaller computational costs than the FEMU method. Note also that we only identified one parameter in this article. We expect the uncertainty to increase for our EGM formulation with the number of identified parameters, which we will investigate in future studies.

Declaration of interests

The authors do not work for, advise, own shares in, or receive funds from any organization that could benefit from this article, and have declared no affiliations other than their research organizations.

Acknowledgments

We would like to thank Colin Laville, Sébastien Imperiale and Philippe Moireau for fruitful discussions. We would also like to thank the French National Research Agency (ANR) for providing financial support to this project (Grant no: ANR-19-CE45-0007).

Appendix A. Quantification of the impact of Poisson ratio errors on the estimation

In this article we have studied the estimation quality of our new formulation of the EGM in large strains and its robustness to noise, and have compared it to the performances of the FEMU

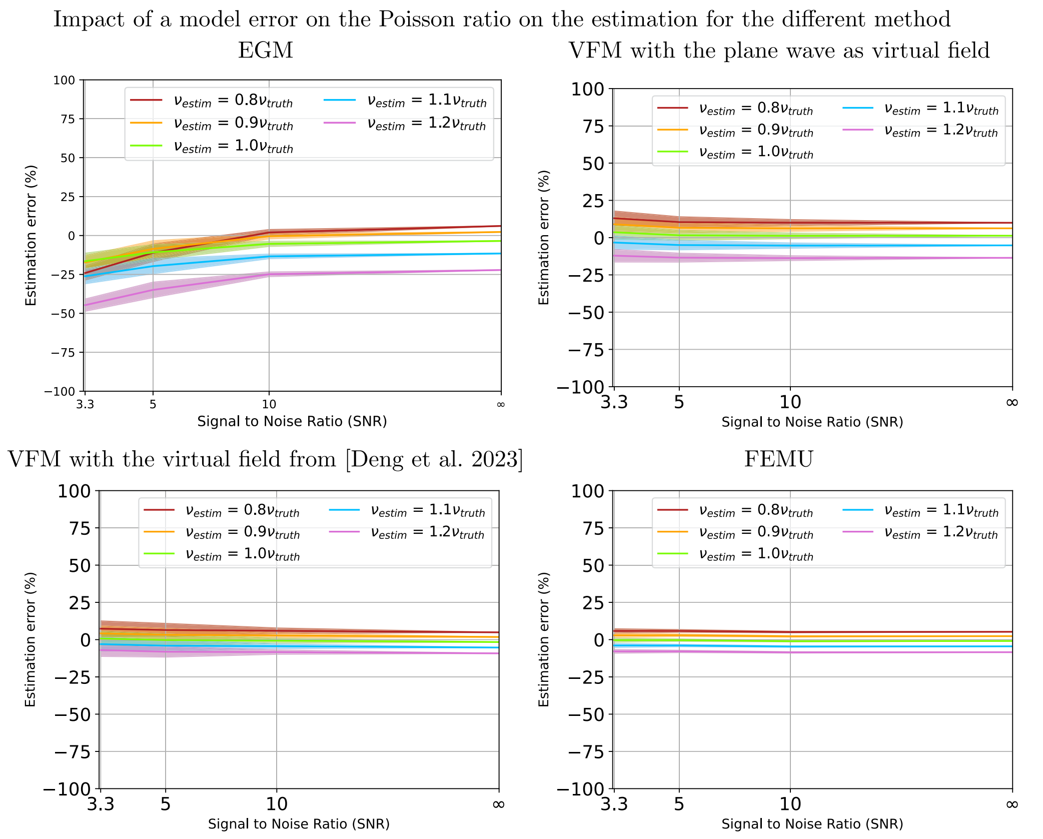


Figure 11. Error distribution on the identified Young modulus depending on the noise, by making an error of −20% (red), −10% (orange), 0% (green), +10% (blue) and +20% (pink) on the fixed Poisson ratio, for the EGM (top left), the VFM with a plane wave as virtual field (top right), the VFM from [8] (bottom left) and the FEMU method (bottom right).

method and of the VFM. However, other factor than the noise can influence the performance quality of an estimation method. In particular, each method presents different sensitivities to model errors. To investigate the robustness of the different methods to model errors, we choose to study in this Appendix the impact of errors of the Poisson ratio value fixed during the estimation. During the synthetic data generation, the Poisson ratio is indeed empirically fixed at 0.3. Model errors on this coefficient can be hence introduced by fixing the Poisson ratio at a value different from the ground-truth during the estimation, e.g., 0.24 or 0.36. This will automatically induce a bias in the estimation.

Figure 11 presents the error distribution of the estimated Young modulus depending on the noise for the four methods—the EGM, the VFM with a plane wave as virtual field, the VFM from [8] and the FEMU method—, when committing an error of −20% (in red), −10% (in orange), 0% (in green), +10% (in blue), +20% (in pink) on the Poisson ratio. The estimation was performed using synthetic data generated by adding noise to images, for a body force test.

For all methods, underestimating the Poisson ratio leads to underestimating the Young modulus, and conversely. This result can be explained by the (negative) linear dependency of the Poisson ratio to the Young modulus for isotropic materials (at fixed bulk modulus): when the Poisson ratio increases, we can expect the Young modulus to increase, which is indeed what is observed here. Additionally, adding an error on ν during the estimation shifts the average and standard deviation of the error distribution. The global tendencies of the distributions however remain similar for all methods.

The results however show that model errors have a great impact on the estimation accuracy for direct methods. For a Poisson ratio set to 120% of its ground-truth value during the estimation process, the error committed for a SNR of $+\infty$—i.e., no noise on the images—is around -18% for the EGM and for the VFM using a plane wave, while it is around 6% for the FEMU method and the VFM from [8].

This result can be explained by the direct dependency of the direct methods on the Poisson ratio in the cost function. Small variations of this coefficient therefore directly impact the expression of the Young modulus. The indirect methods, on the other hand, do not depend directly on the Poisson ratio, which explains their higher robustness to model errors.

Our new formulation of the EGM seems therefore very sensitive to model errors, which is a classical observation for direct methods. This is however a limitation of our new EGM formulation to keep in mind when selecting methods for inverse problems.

Appendix B. Quantification of the impact of regularization errors on the estimation

In this appendix, we study the impact of errors on the regularization value used during the tracking process on the estimation. For the body test, a volume regularization is indeed added during the tracking process. This regularization enforces that the divergence of the stress tensor should be as close as possible to an imposed body force of 0.3 $\mathrm{mN/mm^3}$. However, in some experimental tests, the exact value of the body force might not be known, and the values set for the estimation might therefore differ from the ground-truth value of 0.3 $\mathrm{mN/mm^3}$, which would induce model errors.

We therefore choose to study the impact of error on the regularization term on the estimation. Figure 12 presents the error distribution of the estimated Young modulus depending on the noise for the four methods—the EGM, the VFM with a plane wave as virtual field, the VFM from [8] and the FEMU method—, when committing an error of -20% (in red), -10% (in orange), 0% (in green), +10% (in blue), +20% (in pink) on the volume regularization term. The estimation was performed using synthetic data generated by adding noise to images for a body force test.

The model errors on the volume regularization term seems to mostly impact the EGM; the impact on other methods is indeed marginal. For a volume regularization term set to 120% of the body force ground-truth value, the average error committed for a SNR of $+\infty$—i.e., no noise on the images—is around -10% for the EGM and close to 0% for all other methods. This phenomenon can be explained by the high sensitivity to the measured displacement field of the EGM, already observed in the Results section. Any bias introduced on the regularization term will induce a variation of the synthetic measure. Given its sensitivity to such variations, a high impact on the estimation with the EGM can therefore be expected. Since all other methods are far less sensitive to small accuracy variations of the measured displacement field, estimation with such methods should be robust to a biased regularization, which is what is indeed observed.

Our new formulation of the EGM is therefore more sensitive to regularization errors than non-direct methods. This limitation should be kept in mind when selecting this method to solve problems for which the load in not perfectly known.

Appendix C. Estimation performance for a study in 3D

In this article, we proposed a study on a very simple geometry (a 2D square), for a simple problem (only one parameter identified) as a proof of concept. Indeed, the EGM had never been used

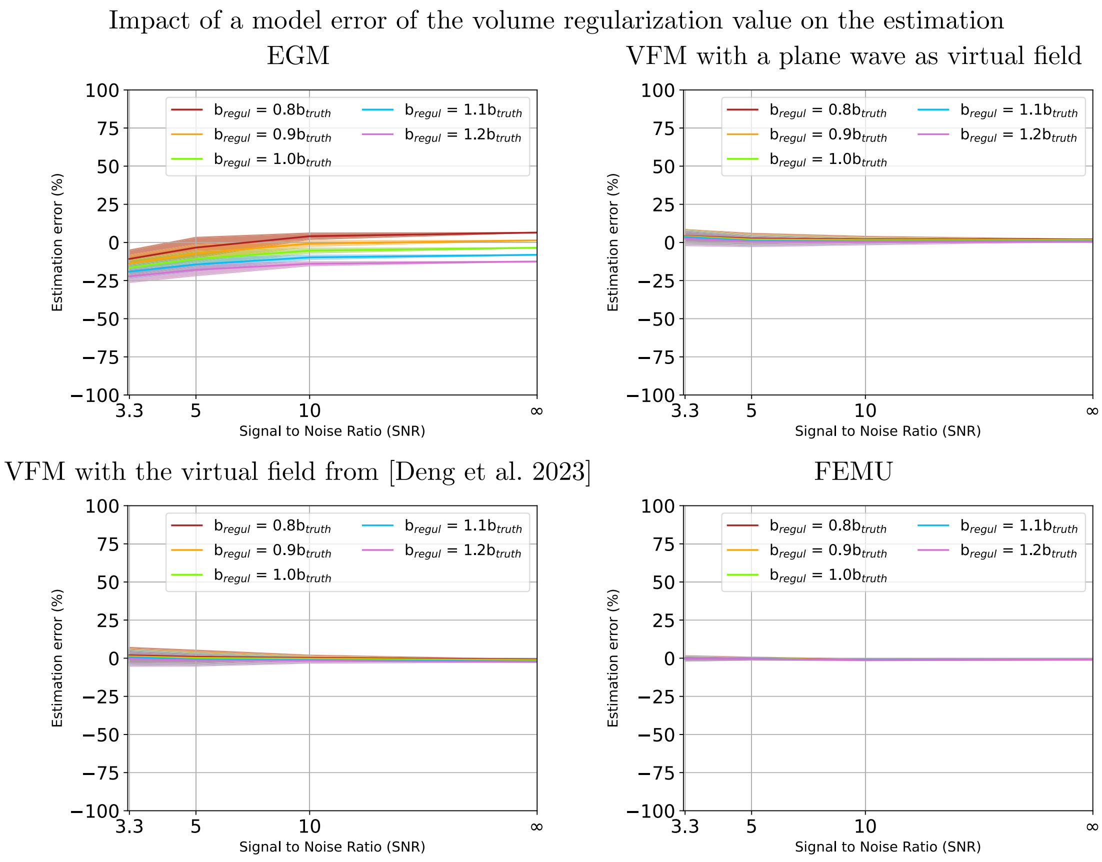


Figure 12. Error distribution on the identified Young's modulus, by making an error of −20% (red), −10% (orange), 0% (green), +10% (blue) and +20% (pink) on the volume regularization value.

in finite deformation cases. Since the method is very sensitive to noise, and as we studied the estimation performances for high noise levels, non-linear problems and fine meshes, it was not straightforward that the estimation would perform well. However, the method can easily be extended to more complex cases. As an illustration, Figure 13 presents the estimation quality, when the problem is in 3D. In this case, a cube is clamped on the left face, and a homogeneous pressure of 0.3 kPa is applied on the right face.

The estimation quality is similiar for both cases. In particular, for a good regularization level, the estimation for the EGM in 3D displays a good accuracy.

Appendix D. Estimation performance for a perfectly and a partially known distribution of boundary forces

We studied in this article the estimation performance of the different methods for a known boundary forces distribution. This case was chosen to evaluate the estimation performance of the boundary term introduced in our new EGM formulation, for cases where the boundary force is known, which is the case for some biomechanical applications [1]. However, the whole distribution is not always known for all experiments: sometimes, only the resultant load is known. As a result, we studied in Figure 14 the impact on the estimation performance of the EGM using

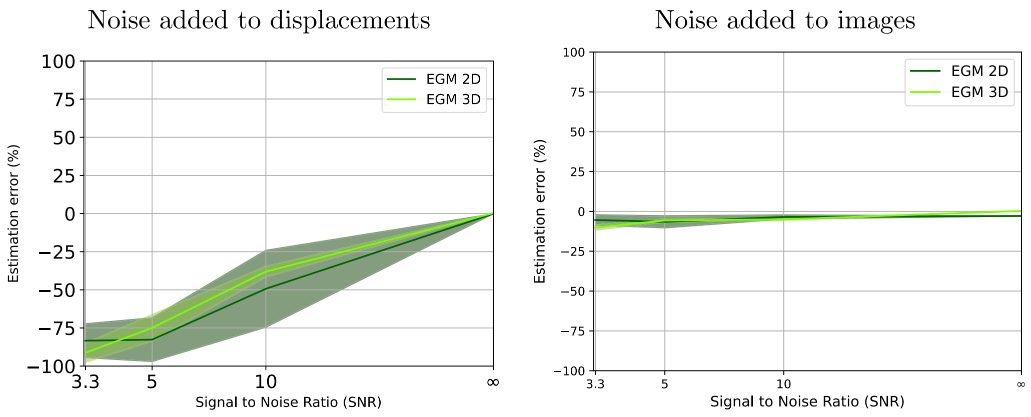


Figure 13. Estimation error using the EGM for a boundary force test for 2D test (dark green) and a 3D test (light green), when noise is added to displacements (left) and when noise is added to images with a regularization of 0.2 (right).

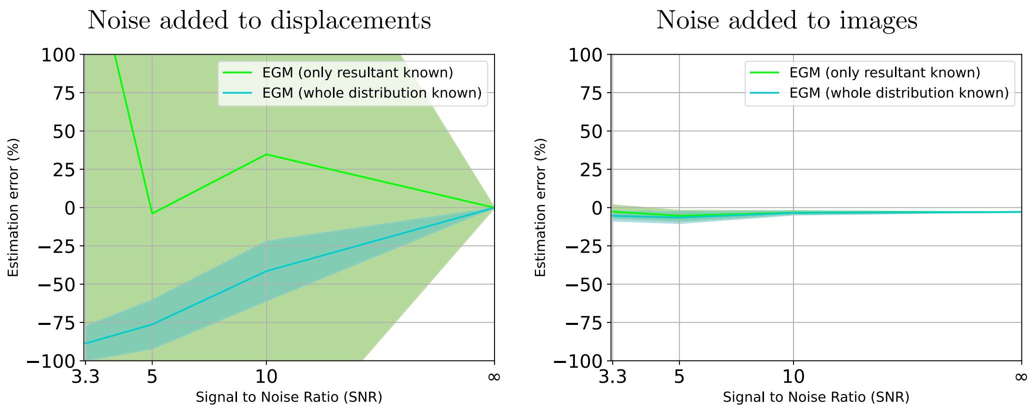


Figure 14. Estimation error using the EGM for a boundary force test when the whole distribution is known (blue) and when only the resultant is known (green), when noise is added to displacements (left) and when noise is added to images with a regularization of 0.2 (right).

the exact boundary force distribution or only the resultant. In the case where only the resultant is known, Equation (33) becomes:

$$L_{\Gamma}^{\mathrm{EGM},F} := \frac{1}{2}\left(\int_{\Gamma}(\underline{\underline{P}}\cdot\underline{N}-\underline{T})\,\mathrm{d}\Gamma\right)^{2}. \tag{55}$$

When noise is added to images, the estimation quality is equivalent whether the exact distribution is known or not. However, when noise is added to displacements, the estimation is of bad quality for both cases: the standard deviation for the case where only the load resultant is known is additionally much higher, which could have been expected as in that case, the estimation holds less information than for the case where the whole distribution is known.When using the EGM with noise added to images and an appropriate regularization, the estimation quality is however very good. Our method could therefore be a good option, even for experimental data which only provide the resultant load and not the whole boundary force distribution.

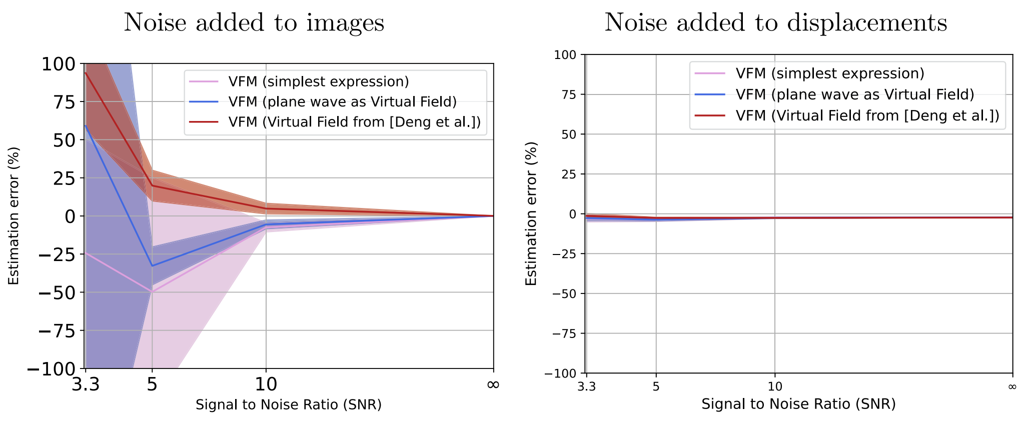


Figure 15. Estimation error using the VFM for a boundary force test when noise is added to displacements (left) and when noise is added to images with a regularization of 0.2 (right), for the virtual field chosen as a plane wave (blue), the virtual field from [8] (red) and the virtual field introduced in Equation (56) (pink).

Appendix E. Estimation performance of an additional virtual field

The two virtual fields used in this article were chosen because on the one hand, the plane wave was easy to define on our geometry, and one parameter (the frequency) could easily be varied to study its impact on the estimation, and on the other hand the virtual field from [8] can be applied to problems with more complex geometries. However, as highlighted by one of our reviewers, these fields are respectively more adapted to non homogeneous problems and to more complex problems. As a result, in this Appendix, we also study the estimation quality for the simplest virtual field possible, expressed as:

$$U^* = \begin{pmatrix} \frac{(\underline{X}[0]-\underline{X}_0[0])}{l} \\ 0 \end{pmatrix}. \tag{56}$$

The performances of the virtual field introduced in Equation (56) are presented in Figure 15, and are very similar to those of the two other virtual fields. Indeed, for noise added to images, there is no noticeable difference between the three virtual fields. Additionally, when noise is added to displacements, for SNRs larger than 10, the performance of the new virtual field is similar to the performances of the plane wave. As a result, this virtual field displays a good estimation quality and its simplicity makes it very adapted to our problem.

Appendix F. Code for Figures 6, 7, 8, 9

F.1. *Imports*

```
[ ]: import dolfin
     import matplotlib.pyplot as plt
     import numpy

     from generate_images import generate_images_and_meshes_from_RivlinCube ### this file is available in
        the repository, and can be accessed to see details in the code

     import dolfin_estim as destim
     import dolfin_warp as dwarp

     ### disable deprecation warning to avoid heavy output
     import warnings
     from ffc.quadrature.deprecation import QuadratureRepresentationDeprecationWarning
     warnings.simplefilter("ignore", QuadratureRepresentationDeprecationWarning)
```

F.2. *Parameters*

F.2.1. *Geometry*

```
[ ]: dim = 2 ### the geometry studied is a square
     mesh_size_lst  = [         ] ### in mm
     mesh_size_lst += [0.1      ] # 0.10
     mesh_size_lst += [0.1/2**1] # 0.05

     cube_params = {"X0":0.2, "Y0":0.2, "X1":0.8, "Y1":0.8, "l":0.1} # cube of characteristic length 0.6
       mm, default element size 0.1 mm
```

F.2.2. *Loading and boundary conditions*

```
[ ]: deformation_type_lst  = [        ]
     deformation_type_lst += ["grav" ] ### body forces
     deformation_type_lst += ["compx"] ### boundary forces

     load_params_boundary = {"type":"pres", "f":0.3} ### boundary force, pressure = 0.3 kPa
     load_params_body = {"type":"volu", "f":0.3} ### body force = 0.3 mN/mm3

     const_params = {"type":"blox"} ### defining load and boundary conditions: here, the square is
       clamped on the left x boundary
```

F.2.3. *Material behavior*

```
[ ]: mat_params = {"model":"CGNH", "parameters":{"E":1., "nu":0.3}} ### defining material constants for
       estimation, E is the Young's modulus in kPa, nu is the Poisson ratio [-]
```

F.2.4. *Images*

```
[ ]: images_folder = "generate_images"
     n_voxels = 100 ### number of voxels on the created images

     noise_level_lst  = [   ] ### investigated noise levels
     noise_level_lst += [0.3]
     noise_level_lst += [0.2]
     noise_level_lst += [0.1]
     noise_level_lst += [0.0]

     n_runs_for_noisy_images = 10 ### number of images generated for a given noise levels; for a same
       noise level, and for convergence of the estimation, different images are created since the noise
       added to the images is a random Gaussian field

     regul_level_lst  = [         ] ### regularization levels
     regul_level_lst += [0.1*2**3] # 0.8
     regul_level_lst += [0.1*2**1] # 0.2
     regul_level_lst += [0.       ] # 0.0
```

F.2.5. *Varying parameters for the estimation*

```
[ ]: ### different cases investigated
     noise_from_images_lst = [True, False] ### synthetic measurement generated by adding noise to images
       (True) or to displacements (False)
     refine_mesh = [True, False] ### refining the mesh or not
     load_type = ["body_force", "boundary_force"] ### either boundary force or body force
     fine_mesh_size = min(mesh_size_lst) ### getting the value for the finer mesh: for this value, the
       mesh will not be refined
```

F.3. *Synthetic measurements*

F.3.1. *Synthetic images*

[ ]:
```
for deformation_type in deformation_type_lst:
    generate_images_and_meshes_from_RivlinCube( ### creating very fine mesh for image creation
        images_n_dim = dim,
        images_n_voxels = n_voxels,
        deformation_type = deformation_type,
        texture_type = "no",
        noise_level = 0,
        run_model = 1,
        generate_images = 0)
    for noise_level in noise_level_lst :
        n_runs = n_runs_for_noisy_images if (noise_level > 0) else 1
        for k_run in range(1, n_runs+1):
            generate_images_and_meshes_from_RivlinCube(### creating images
                images_n_dim = dim,
                images_n_voxels = n_voxels,
                deformation_type = deformation_type,
                texture_type = "tagging",
                noise_level = noise_level,
                k_run = k_run if (n_runs > 1) else None,
                run_model = 0,
                generate_images = 1)
```

F.3.2. *Ground-truth motion*

[ ]:
```
for deformation_type in deformation_type_lst:
    for mesh_size in mesh_size_lst:
        generate_images_and_meshes_from_RivlinCube(### ground-truth motion + meshes
            images_n_dim = dim,
            # images_n_voxels = 1,
            mesh_size = mesh_size,
            deformation_type = deformation_type,
            texture_type = "no",
            noise_level = 0,
            run_model = 1,
            generate_images = 0)
```

F.3.3. *Tracking*

[ ]:
```
for deformation_type in deformation_type_lst:
    for noise_level in noise_level_lst:
        n_runs = n_runs_for_noisy_images if (noise_level > 0) else 1 ### if no noise, there is no
  need for different samples; for noise levels higher than 0, a Gaussian noise id generated; there
  is hence the need for different samples, as randomness is introduced

        for k_run in range(1, n_runs+1):
            for mesh_size in mesh_size_lst:
                for regul_level in regul_level_lst:
                    ### getting regularization type, depending on the type of deformation applied
                    if deformation_type == "compx":
                        regul_type = "discrete-equilibrated-tractions-tangential"
                        working_folder = "run_warp_compx"
                        regul_b = 0.0
                    else:
                        regul_type = "discrete-equilibrated-tractions-normal-tangential"
                        working_folder = "run_warp_grav"
                        regul_b = 0.3

                    #### getting files and folder names
                    images_basename = "square"
                    images_basename += "-"+deformation_type
                    images_basename += "-tagging"
                    images_basename += "-noise="+str(noise_level)
                    if (n_runs > 1):
                        images_basename += "-run="+str(k_run).zfill(2)
                    mesh_basename = "square"
                    mesh_basename += "-"+deformation_type
                    mesh_basename += "-h="+str(mesh_size)
                    mesh_basename += "-mesh"
                    if mesh_size != fine_mesh_size:
                        meshes = [dolfin.Mesh(images_folder+"/"+mesh_basename+"coarse.xml"), dolfin.
Mesh(images_folder+"/"+mesh_basename+"refined.xml")]
                    else:
                        meshes = [dolfin.Mesh(images_folder+"/"+mesh_basename+"coarse.xml")]
                    if deformation_type == "grav":
                        regul_surface_subdomain = []
                        regul_surface_subdomain_id = []
                        for mesh in meshes:
                            regul_surface_subdomain_ = dolfin.MeshFunction("size_t", mesh, mesh.
topology().dim()-1)
                            regul_surface_subdomain_.set_all(1)
                            xmin_sd = dolfin.CompiledSubDomain("near(x[0], x0) && on_boundary", x0 =
0.2)
                            xmin_sd.mark(regul_surface_subdomain_, 0)
                            regul_surface_subdomain.append(regul_surface_subdomain_)
                            regul_surface_subdomain_id.append(1)
                    else:
                        regul_surface_subdomain = None
                        regul_surface_subdomain_id = None
                    working_basename = images_basename
                    working_basename += "-h="+str(mesh_size)
                    working_basename += "-"+regul_type
                    working_basename += "-regul="+str(regul_level)
                    working_basename += "-b="+str(regul_b)

                    #### tracking process
                    if mesh_size != fine_mesh_size: ### get refined solution for multiresolution
                        refinement_levels = [0,1]
                    else:
                        refinement_levels = [0]
                    dwarp.warp_and_refine(
                        working_folder = working_folder,
                        working_basename = working_basename,
                        images_folder = images_folder,
                        images_basename = images_basename,
                        meshes = meshes,
                        regul_type = regul_type,
                        regul_model = "ciarletgeymonatneohookean",
                        regul_level = regul_level,
                        regul_poisson = 0.3,
                        regul_b = [regul_b]+[0.]*(dim-1),
                        regul_surface_subdomain_data = regul_surface_subdomain,
                        regul_surface_subdomain_id = regul_surface_subdomain_id,
                        relax_type = "backtracking",
                        tol_dU = 1e-2,
                        n_iter_max = 100,
                        normalize_energies = 1,
                        refinement_levels = refinement_levels,
                        silent = True)
```

F.4. *Generating plots*

F.4.1. *Helpers*

```
[ ]: SNR_lst = [] ### defining SNRs -- Signal-to-Noise ratios--
     for noise in noise_level_lst:
         if noise == 0.:
             SNR_lst.append(40.) ### setting the SNR arbitrarily when should be + infinity
         else:
             SNR_lst.append(1/noise)
```

```
[ ]: def writing_results_to_pdf(results_all = [], SNR_lst = [], method_lst = [], mesh_size = 0.1,
       noise_from_images = False, load_type = "body_force", regul = "", refine = True): ### writing
       results to pdf

         ### plotting parameters
         fig, ax = plt.subplots()
         plt.rc("xtick", labelsize = 16)
         plt.rc("ytick", labelsize = 16)
         plt.rc("legend", fontsize = 12)
         plt.ylim([-100, 100])
         color_lst = ['forestgreen', 'royalblue', 'firebrick', 'gold'] #### colors associated to each
       method

         ### labels
         plt.xlabel("Signal to Noise Ratio (SNR)", fontsize = 12)
         plt.ylabel("Estimation error (%)", fontsize = 12)
         label_lst = ["EGM", "VFM (plane wave as Virtual Field)", "VFM (Virtual Field from [Deng et al.
       ])", "FEMU"] ### labels for legends

         for method in method_lst:
             plt.plot(SNR_lst, results_all[str(method)]["E_average"], color = color_lst[0], label =
       label_lst[0])
             plt.xlim([3.3, 20.])
             ax.fill_between(SNR_lst, results_all[str(method)]["E_+"], results_all[str(method)]["E_-"],
       alpha = 0.5, color = color_lst[0])
             plt.gca().set_xscale('log')

             color_lst = color_lst[1:]
             label_lst = label_lst[1:]

         ax.set_xticks([])
         ax.set_xticks([], minor = True)
         plt.xticks(SNR_lst, [3.3, 5, 10, '$\infty$'])
         plt.legend(loc = 'upper right')
         plt.grid()

         plt.show()

         plt.savefig("./plot_noise_error_different_methods-mesh_size = "+str(mesh_size)+"refine =
       "+str(refine)+"-noise_from_images = "+str(noise_from_images)+"-load_type = "+str(load_type)+"regul
       = "+str(regul)+".pdf", bbox_inches = 'tight')
```

```
[ ]: def run_noise_on_disp(method_lst = [], load_type = "body_force", load_params = {}, mesh_size = 0.1,
       cube_params = {}, refine = True, SNR_lst = [], regul_number = 0.3): ### synthetic data by adding
       noise to displacements
         results_all = {}
         for method in method_lst: ## for each method
             results_std = {}
             results = {}
             noise_results = []
             E_average, E_plus, E_minus = [], [], []
             E_all = []
             for noise in noise_level_lst: ### for each noise level
                 E_results = []
                 for i in range(1, 11): ### 10 iterations for each noise level, several iterations are
       required since we added Gaussian noise
```

```
                noise_results.append(noise)
                try:
                    E = destim.identifying_parameter(method = method, delta = 9*0.6, load_type =
load_type, load_params = load_params, mesh_size = mesh_size, cube_params = cube_params, refine =
refine, noise_from_images = False, noise = noise, regul_number = regul_number)
                    if "refine" in cube_params.keys() and not refine:
                        cube_params.pop("refine")
                    E_error = (E-1)*100
                    E_all.append(E_error)
                    E_results.append(E_error)
                except: ##### when adding highly unstructured noise, the computation may not always
converge; in this case the results are not taken into account
                    pass
            E_average.append(numpy.average(E_results))
            E_plus.append(numpy.average(E_results) + numpy.std(E_results))
            E_minus.append(numpy.average(E_results) - numpy.std(E_results))
        results_std["noise"] = noise_level_lst
        results_std["E_+"] = E_plus
        results_std["E_-"] = E_minus
        results_std["E_average"] = E_average
        results["noise"] = noise_results
        results["E"] = E_all
        results_all[str(method)] = results_std
    writing_results_to_pdf(mesh_size = mesh_size, SNR_lst = SNR_lst, results_all = results_all,
noise_from_images = False, load_type = load_type, method_lst = method_lst, refine = refine)
```

```
[ ]: def run_noise_on_images(method_lst = [], load_type = "body_force", load_params = {}, mesh_size = 0.
1, cube_params = {}, refine = True, SNR_lst = [], regul_number = 0.3): ### synthetic data by
adding noise to images
    results_all = {}
    for regul in regul_level_lst: ### for each regularization level
        for method in method_lst: ### for each method
            results_std = {}
            results = {}
            noise_results = []
            E_average, E_plus, E_minus = [], [], []
            E_all = []
            for noise in noise_level_lst: ### for each noise level
                E_results = []
                for i in range(1, 11): ### 10 iterations for each noise level, several iterations
are required since we added Gaussian noise --10 corresponding to the number of images generated
for each noise level, for each regularization number--
                    run = str(i).zfill(2)
                    noise_results.append(noise)
                    try:
                        E = destim.identifying_parameter(method = method, delta = 9*0.6, load_type =
load_type, load_params = load_params, mesh_size = mesh_size, cube_params = cube_params, refine =
refine, noise_from_images = True, noise = noise, regul = regul, run = run, regul_number =
regul_number)
                        E_error = (E-1)*100
                        E_all.append(E_error)
                        E_results.append(E_error)
                        print("E estimated", E)
                    except: ### in case the computation does not converge
                        pass
                E_average.append(numpy.average(E_results))
                E_plus.append(numpy.average(E_results)+numpy.std(E_results))
                E_minus.append(numpy.average(E_results)-numpy.std(E_results))
                results_std["noise"] = noise_level_lst
            results_std["E_+"] = E_plus
            results_std["E_-"] = E_minus
            results_std["E_average"] = E_average
            results["noise"] = noise_results
            results["E"] = E_all
            results_all[str(method)] = results_std
        writing_results_to_pdf(mesh_size = mesh_size, SNR_lst = SNR_lst, results_all = results_all,
noise_from_images = True, load_type = load_type, method_lst = method_lst, regul = regul, refine =
refine)
```

F.4.2. *Plots*

[60]:
```
method_lst = ["EGM", "VFM", "VFM_deng", "FEMU"] ### 4 different estimation methods investigated

results_all = {}

for mesh_size in mesh_size_lst:
    for refine in refine_mesh:
        print('refine', refine)
        if refine and mesh_size == fine_mesh_size: ### meshes with elements smaller than 0.05 --in
  this example-- were not investigated in the scope of this article
            pass
        else:
            for load in load_type:
                print("load", load)
                assert (load in ("body_force", "boundary_force" )), "Load should be body_force or
  boundary_force, aborting..."
                if load == "body_force":
                    load_params = load_params_body
                    regul_number = 0.3    ### sets volume regularization, chosen at the ground-truth
  value --0.3 mN--
                if load == "boundary_force":
                    load_params = load_params_boundary
                    regul_number = 0.0
                for noise_from_images in noise_from_images_lst:
                    print("noise_from_images", noise_from_images)
                    if noise_from_images:
                        run_noise_on_images(method_lst = method_lst, load_type = load, load_params =
  load_params, mesh_size = mesh_size, cube_params = cube_params, refine = refine, SNR_lst = SNR_lst,
  regul_number = regul_number)
                    else:
                        run_noise_on_disp(method_lst = method_lst, load_type = load, load_params =
  load_params, mesh_size = mesh_size, cube_params = cube_params, refine = refine, SNR_lst = SNR_lst,
  regul_number = regul_number)
```

Appendix G. Code for Figure 10

G.1. *Imports*

[ ]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy

import dolfin_estim as destim
import dolfin_mech as dmech

### disable deprecation warning to avoid heavy output
import warnings
from ffc.quadrature.deprecation import QuadratureRepresentationDeprecationWarning
warnings.simplefilter("ignore", QuadratureRepresentationDeprecationWarning)
```

G.2. *Parameters*

G.2.1. *Geometry and varying parameters*

[ ]:
```
mesh_size = 0.1 ### mesh size for the resolution
l = 0.6 ### characteristic length of the cube
delta_lst = [1000, 10, 5, 2.5, 1.7, 1.1, 0.9, 0.099] ### list of wave lengths chosen
noise_lst = [0.0, 0.01, 0.05, 0.1] ### noise levels chosen
```

G.2.2. *Material behavior*

[ ]:
```
params = {
    "E":1.,         # kPa
    "nu":0.3}       # [-]
mat_params = {"model":"CGNH", "parameters":params} ### hyperelastic law
```

G.2.3. *Loading*

```
[ ]: load_params_boundary = {"type":"pres", "f":0.3} ### studied in the case of a boundary force, f = 3
       kPa
```

G.3. *Generating plots*

G.3.1. *Helpers*

```
[ ]: cube_params = {"X0":0.2, "Y0":0.2, "X1":0.8, "Y1":0.8, "l":0.01} # exact solution --without noise--
       computed on a very fine mesh for better resolution, computed here and not in dolfin_estim in order
       to avoid heavy computations

     u, v = dmech.run_RivlinCube_Hyperelasticity(
         dim          = 2,
         cube_params  = cube_params,
         mat_params   = mat_params,
         step_params  = {"dt_ini":1/20},
         const_params = {"type":"blox"},
         load_params  = load_params_boundary,
         get_results  = 1,
         res_basename = "./")
```

```
[ ]: SNR_lst = [] ### defining SNRs -- Signal-to-Noise ratios--
     for noise in noise_lst:
         if noise == 0.:
             SNR_lst.append(40.) ### setting the SNR arbitrarily when should be + infinity
         else:
             SNR_lst.append(1/noise)
```

```
[ ]: def writing_results_to_pdf(results_all = [], delta_lst = [], SNR_lst = []): ### function to write
       results --boxplots--

         fig, ax = plt.subplots()
         ### plotting parameters
         plt.rc("xtick", labelsize = 20)
         plt.rc("ytick", labelsize = 20)
         plt.rc("legend", fontsize = 12)
         plt.xlabel("Signal to Noise Ratio (SNR)", fontsize = 12)
         plt.ylabel("Estimation error (%)", fontsize = 12)
         fig.set_size_inches(10, 6)
         plt.grid()
         plt.xlim([5, 2000.])
         plt.ylim([-100, 100.])

         nb_deltas = len(delta_lst)
         spacing = numpy.linspace(-0.0305*(nb_deltas//2), 0.0305*(nb_deltas//2), nb_deltas) ### spacing
       between the different boxplots
         color_lst = matplotlib.cm.viridis(numpy.linspace(0, 1, len(delta_lst))) #### colors in the
       viridis scale

         count_spacing = 0
         for delta in delta_lst:
             results_delta = results_all[str(delta_lst[count_spacing])] ### getting results for each
       value of delta, stored into the dictionary "results_delta"
             down_up_lst = [[abs(results_delta["E_-"][i]-results_delta["E_average"][i])for i in
       range(len(results_delta["noise"]))], [abs(results_delta["E_+"][i]-results_delta["E_average"][i])
       for i in range(len(results_delta["noise"]))]] #### to create y-error bars, defining up and down
       whisker
             mean_lst = [results_delta["E_average"][i] for i in range(len(results_delta["noise"]))]
             SNR_lst_scaled = [SNR_lst[i]*numpy.exp((spacing[count_spacing]))**numpy.log(10) for i in
       range(len(SNR_lst))] ### to include spacing, that will remain even with the log scale
             plt.errorbar(SNR_lst_scaled, mean_lst, yerr = down_up_lst, capsize = 4, fmt = "s", ecolor =
       color_lst[-1], markeredgecolor = color_lst[-1], markerfacecolor = color_lst[-1], label =
       r'$\Delta$'+"  = " +'{0:.1f}'.format(delta)+"l")
```

```
        plt.gca().set_xscale('log') ### plotting y-error bars
        color_lst = color_lst[:-1] ### different color for each error bar
        count_spacing+= 1
    ax.set_xticks([])
    ax.set_xticks([], minor = True)
    plt.xticks(SNR_lst, ['$\infty$', 100, 20, 10])
    plt.legend(loc = "upper right", fontsize = 13, ncol = 2)

    plt.savefig("./different_values_beta_VFM_plane_waves.pdf", bbox_inches = 'tight')
    plt.show()
```

G.3.2. *Plots*

```
[ ]: results_all = {}
     for delta in delta_lst: ### for each value of the period
         ### creating arrays and lists to store results
         results_std = {}
         results = {}
         noise_results = []
         E_all = []
         results_std["noise"] = noise_lst
         E_average, E_plus, E_minus = [], [], []
         for noise in noise_lst:
             E_results = []
             for i in range(1, 50): #### many iterations for convergence of the distribution
                 noise_results.append(noise)
                 E = destim.identifying_parameter(method = "VFM", delta = delta*1, load_type =
       "boundary_force", load_params = load_params_boundary, mesh_size = mesh_size, cube_params =
       cube_params, refine = False, noise_from_images = False, noise = noise,  u_params = {"u":u, "v":v})
       #  this computation was only conducted for a boundary force test, without refining the mesh, and
       with synthetic data generated by adding noise to displacements
                 E_error = (E-1)*100 ### computing the error compared to the ground truth, i.e., E = 1
       kPa
                 E_all.append(E_error)
                 E_results.append(E_error)
             E_average.append(numpy.average(E_results))
             E_plus.append(numpy.average(E_results)+numpy.std(E_results))
             E_minus.append(numpy.average(E_results)-numpy.std(E_results))
         results_std["E_+"] = E_plus
         results_std["E_-"] = E_minus
         results_std["E_average"] = E_average
         results_all[str(delta)] = results_std
         results["noise"] = noise_results
         results["E"] = E_all
     writing_results_to_pdf(results_all = results_all, delta_lst = delta_lst, SNR_lst = SNR_lst)
```

Appendix H. Code for Figures 11–12

H.1. *Imports*

```
[ ]: import matplotlib.pyplot as plt
     import numpy
     import os
     import dolfin

     import dolfin_warp as dwarp
     import dolfin_estim as destim

     from generate_images import generate_images_and_meshes_from_RivlinCube

     ### disable deprecation warning to avoid heavy output
     import warnings
     from ffc.quadrature.deprecation import QuadratureRepresentationDeprecationWarning
     warnings.simplefilter("ignore", QuadratureRepresentationDeprecationWarning)
```

H.2. *Parameters*

H.2.1. *Geometry*

```
[ ]: ### Geometry
     dim = 2 ### the geometry studied is a square
     cube_params = {"X0":0.2, "Y0":0.2, "X1":0.8, "Y1":0.8, "l":0.1} ### l: mesh size, in mm, by default
       0.1, X0, Y0, X1, Y1 correspond to the min and max x and y coordinates of the square

     ### mesh sizes investigated
     mesh_size = 0.1 ## in mm
```

H.2.2. *Loading and boundary conditions*

```
[ ]: ### Loading parameters
     const_params = {"type":"blox"} ### defining load and boundary conditions: here, the square is
       clamped on the left x boundary
     load_params_body = {"type":"volu", "f":0.3}  ### a volume force of 0.3 mN/mm3 is applied on the cube

     ### volume test
     deformation_type_lst = ["grav"]
```

H.2.3. *Material behavior*

```
[ ]: ### Material

     nu_ref = 0.3 ### defining ground-truth value of Poisson ratio [-]
     b_ref = 0.3 ### defining ground-truth value of the volume regularization term, in mN/mm3
     E_ref = 1 ### kPa, Young's modulus

     mat_params = {"model":"CGNH", "parameters":{"E":E_ref, "nu":nu_ref}} ### defining material constants
       for estimation
```

H.2.4. *Images*

```
[ ]: ### Noises levels
     noise_level_lst  = [   ]
     noise_level_lst += [0.3]
     noise_level_lst += [0.2]
     noise_level_lst += [0.1]
     noise_level_lst += [0.0]
     ### Images parameters
     n_runs_for_noisy_images = 10 ### number of images generated for a given noise levels; for a same
       noise level, and for convergence of the estimation, different images are created since the noise
       added to the images is a random Gaussian field
     n_voxels = 100
     ### volume regularization constants
     regul_b_lst  = []
     regul_b_lst += [0.0]
     regul_b_lst += [0.24]
     regul_b_lst += [0.27]
     regul_b_lst += [0.3]
     regul_b_lst += [0.33]
     regul_b_lst += [0.36]
     ### regularization levels
     regul_level_lst  = [        ]
     regul_level_lst += [0.1*2**1] # 0.2
     regul_level_lst += [0.1     ] # 0.1
     regul_level_lst += [0.1/2**1] # 0.05
     regul_level_lst += [0.] # 0.0
     ### name and folder creation
     images_folder = "generate_images"
     if not os.path.exists(images_folder): ### checking if folder already exists
         os.mkdir(images_folder)
```

H.2.5. *Bias*

```
[ ]: bias_lst = [0.8, 0.9, 1., 1.1, 1.2] ### studying the impact of an error of -20, -10, 0, 10, and 20 %
       on a model parameter on the estimation
     bias_param_lst = ['nu', 'b'] ### studying model errors on the Poisson ratio and on the
       regularization b
```

H.3. *Synthetic measurements*

H.3.1. *Synthetic images*

```
[ ]: for deformation_type in deformation_type_lst:
         generate_images_and_meshes_from_RivlinCube( ### creating fine mesh for image computation
             images_n_dim = dim,
             images_n_voxels = n_voxels,
             deformation_type = deformation_type,
             texture_type = "no",
             noise_level = 0,
             run_model = 1,
             generate_images = 0)
         for noise_level  in noise_level_lst :
             n_runs = n_runs_for_noisy_images if (noise_level > 0) else 1
             for k_run in range(1, n_runs+1):
                 generate_images_and_meshes_from_RivlinCube(
                     images_n_dim = dim,
                     images_n_voxels = n_voxels,
                     deformation_type = deformation_type,
                     texture_type = "tagging",
                     noise_level = noise_level,
                     k_run = k_run if (n_runs > 1) else None,
                     run_model = 0,
                     generate_images = 1)
```

H.3.2. *Ground-truth motion*

```
[ ]: for deformation_type in deformation_type_lst:
         generate_images_and_meshes_from_RivlinCube(### ground truth motion + meshes
                 images_n_dim = dim,
                 images_n_voxels = 1,
                 mesh_size = mesh_size,
                 deformation_type = deformation_type,
                 texture_type = "no",
                 noise_level = 0,
                 run_model = 1,
                 generate_images = 0)
```

H.3.3. *Tracking*

```
[ ]: for deformation_type in deformation_type_lst:
         for noise_level in noise_level_lst:
             n_runs = n_runs_for_noisy_images if (noise_level > 0) else 1 ### if no noise, there is no
       need for different samples; for noise levels higher than 0, a Gaussian noise id generated; there
       is hence the need for different samples, as randomness is introduced
             for k_run in range(1, n_runs+1):
                 for regul_b in regul_b_lst:
                     for regul_level in regul_level_lst:
                         regul_type = "discrete-equilibrated-tractions-normal-tangential"
                         working_folder = "run_warp_grav"

                         #### getting files and folder names
                         images_basename = "square"
                         images_basename += "-"+deformation_type
                         images_basename += "-tagging"
                         images_basename += "-noise="+str(noise_level)
                         if (n_runs > 1):
                             images_basename += "-run="+str(k_run).zfill(2)
```

```
                    mesh_basename = "square"
                    mesh_basename += "-"+deformation_type
                    mesh_basename += "-h="+str(mesh_size)
                    mesh_basename += "-mesh"
                    working_basename = images_basename
                    working_basename += "-h="+str(mesh_size)
                    working_basename += "-"+regul_type
                    working_basename += "-regul="+str(regul_level)
                    working_basename += "-b="+str(regul_b)

                    meshes = [dolfin.Mesh(images_folder+"/"+mesh_basename+"coarse.xml"), dolfin.
 Mesh(images_folder+"/"+mesh_basename+"refined.xml")]
                    regul_surface_subdomain = []
                    regul_surface_subdomain_id = []
                    for mesh in meshes:
                        regul_surface_subdomain_ = dolfin.MeshFunction("size_t", mesh, mesh.
 topology().dim()-1)
                        regul_surface_subdomain_.set_all(1)
                        xmin_sd = dolfin.CompiledSubDomain("near(x[0], x0) && on_boundary", x0 = 0.
 2)
                        xmin_sd.mark(regul_surface_subdomain_, 0)
                        regul_surface_subdomain.append(regul_surface_subdomain_)
                        regul_surface_subdomain_id.append(1)

                    refinement_levels = [0,1]
                    dwarp.warp_and_refine(
                        working_folder = working_folder,
                        working_basename = working_basename,
                        images_folder = images_folder,
                        images_basename = images_basename,
                        meshes = meshes,
                        regul_type = regul_type,
                        regul_model = "ciarletgeymonatneohookean",
                        regul_level = regul_level,
                        regul_poisson = 0.3,
                        regul_b = [regul_b]+[0.]*(dim-1),
                        regul_surface_subdomain_data = regul_surface_subdomain,
                        regul_surface_subdomain_id = regul_surface_subdomain_id,
                        relax_type = "backtracking",
                        tol_dU = 1e-2,
                        n_iter_max = 100,
                        normalize_energies = 1,
                        refinement_levels = refinement_levels,
                        silent = True)
```

H.4. *Generating plots*

H.4.1. *Helpers*

```
[ ]: SNR_lst = [] ### defining SNRs -- Signal-to-Noise ratios--
     for noise in noise_level_lst:
         if noise == 0.:
             SNR_lst.append(40.) ### setting the SNR arbitrarily when should be + infinity
         else:
             SNR_lst.append(1/noise)
```

```
[ ]: def writing_results_to_pdf(mesh_size = 0.1, SNR_lst = [], results_all = {}, noise_from_images =
       True, regul = "", bias_lst = [], bias_param = "", method = "EGM"):

         ### plotting parameters
         fig, ax = plt.subplots()
         plt.rc("xtick", labelsize = 16)
         plt.rc("ytick", labelsize = 16)
         plt.rc("legend", fontsize = 12)
         plt.ylim([-100, 100])
         if bias_param == "nu":
```

```
        bias_param = r'$\nu_{truth}$'
        param_math = r'$\nu_{estim}$'
    else:
        param_math = bias_param+"$_{regul}$"
        bias_param = bias_param+"$_{truth}$"

    plt.xlabel("Signal to Noise Ratio (SNR)", fontsize = 12)
    plt.ylabel("Estimation error (%)", fontsize = 12)
    color_lst = ['firebrick', 'orange', 'lawngreen', 'deepskyblue', 'orchid']

    for bias in bias_lst:
        plt.plot(SNR_lst, results_all[str(bias)]["E_average"], color = color_lst[0], label =
  param_math+" = "+str(bias)+bias_param)
        plt.xlim([3.3, 20.])
        ax.fill_between(SNR_lst, results_all[str(bias)]["E_+"], results_all[str(bias)]["E_-"], alpha
  = 0.5, color = color_lst[0])
        plt.gca().set_xscale('log')
        color_lst = color_lst[1:]

    ax.set_xticks([])
    ax.set_xticks([], minor = True)
    plt.xticks(SNR_lst, [3.3, 5, 10, '$\infty$'])

    plt.legend(loc = "upper right", fontsize = 13, ncol = 2)
    plt.grid()

    plt.savefig("./model_error_for_error_on"+str(bias_param)+"with_method =
  "+str(method)+str(mesh_size)+"-noise_from_images = "+str(noise_from_images)+"regul =
  "+str(regul)+".pdf", bbox_inches = 'tight')
    plt.show()
```

```
[ ]: def run_noise_on_images(method_lst = [], load_type = "body_force", load_params = {}, mesh_size = 0.
       1, cube_params = {}, refine = 0, SNR_lst = [], noise_level_lst = [], bias_param = "", bias_lst =
       [], noise_from_images = True, regul_number = 0.3):
         results_all = {}
         for method in method_lst:
             for bias in bias_lst:
                 results_std = {}
                 results = {}
                 noise_results = []
                 E_average, E_plus, E_minus = [], [], []
                 E_all = []
                 nu_biased = nu_ref
                 if bias_param == "nu":
                     nu_biased = bias*nu_ref
                 elif bias_param == "b":
                     regul_number = bias*b_ref
                 for noise in noise_level_lst:
                     E_results = []
                     for i in range(1, 11):
                         run = str(i).zfill(2)
                         noise_results.append(noise)
                         E = destim.identifying_parameter(method = method, nu = nu_biased, delta = 5*0.6,
       load_type = load_type, load_params = load_params, mesh_size = mesh_size, cube_params =
       cube_params, refine = refine, noise_from_images = noise_from_images, noise = noise, regul = 0.2,
       regul_number = regul_number, run = run)
                         E_error = (E-E_ref)/(E_ref)*100
                         E_all.append(E_error)
                         E_results.append(E_error)
                     E_average.append(numpy.average(E_results))
                     E_plus.append(numpy.average(E_results)+numpy.std(E_results))
                     E_minus.append(numpy.average(E_results)-numpy.std(E_results))
                     results_std["noise"] = noise_level_lst
                     results_std["E_+"] = E_plus
                     results_std["E_-"] = E_minus
                     results_std["E_average"] = E_average
                     results["noise"] = noise_results
                     results["E"] = E_all
                     results_all[str(bias)] = results_std
             writing_results_to_pdf(mesh_size = mesh_size, SNR_lst = SNR_lst, results_all = results_all,
       noise_from_images = True, method = method, regul = 0.2, bias_lst = bias_lst, bias_param =
       bias_param)
```

H.4.2. *Plots*

```
[ ]: method_lst = ["EGM", "VFM", "VFM_deng", "FEMU"]

     results_all = {}

     for bias_param in bias_param_lst:
         run_noise_on_images(method_lst = method_lst, load_type = "body_force", load_params =
       load_params_body, mesh_size = mesh_size, cube_params = cube_params, refine = False, SNR_lst =
       SNR_lst, bias_param = bias_param, bias_lst = bias_lst, noise_level_lst = noise_level_lst,
       noise_from_images = True, regul_number = b_ref)
```

References

[1] C. Patte, P.-Y. Brillet, C. Fetita, J.-F. Bernaudin, T. Gille, H. Nunes, D. Chapelle and M. Genet, "Estimation of regional pulmonary compliance in idiopathic pulmonary fibrosis based on personalized lung poromechanical modeling", *J. Biomech. Eng.* **144** (2022), no. 9, article no. 091008.

[2] C. Laville, C. Fetita, T. Gille, P. Brillet, H. Nunes, J.-F. Bernaudin and M. Genet, "Comparison of optimization parametrizations for regional lung compliance estimation using personalized pulmonary poromechanical modeling", *Biomech. Model. Mechanobiol.* **22** (2023), no. 5, pp. 1541–1554.

[3] S. Avril, M. Bonnet, A.-S. Bretelle, et al., "Overview of identification methods of mechanical parameters based on full-field measurements", *Exp. Mech.* **48** (2008), pp. 381–402.

[4] S. Roux and F. Hild, "Optimal procedure for the identification of constitutive parameters from experimentally measured displacement fields", *Int. J. Solids Struct.* **184** (2020), pp. 14–23. ISSN: 00207683.

[5] J. D. Collins, F. Hart, T. Hasselman and B. Kennedy, "Statistical identification of structures", *AIAA J.* **12** (1974), no. 2, pp. 185–190.

[6] M. Bonnet and A. Constantinescu, "Inverse problems in elasticity", *Inverse Probl.* **21** (2005), no. 2, article no. R1.

[7] A. Wittek, W. Derwich, K. Karatolios, C. P. Fritzen, S. Vogt, T. Schmitz-Rixen and C. Blase, "A finite element updating approach for identification of the anisotropic hyperelastic properties of normal and diseased aortic walls from 4D ultrasound strain imaging", *J. Mech. Behav. Biomed. Mater.* **58** (2016), pp. 122–138.

[8] J. Deng, X. Guo, Y. Mei and S. Avril, "FEniCS implementation of the Virtual Fields Method (VFM) for nonhomogeneous hyperelastic identification", *Adv. Eng. Softw.* **175** (2023), article no. 103343. ISSN: 09659978.

[9] D. Claire, F. Hild and S. Roux, "A finite element formulation to identify damage fields: the equilibrium gap method", *Int. J. Numer. Methods Eng.* **61** (2004), no. 2, pp. 189–208.

[10] S. Roux and F. Hild, "Digital image mechanical identification (DIMI)", *Exp. Mech.* **48** (2008), no. 4, pp. 495–508.

[11] J. Boddapati, M. Flaschel, S. Kumar, L. De Lorenzis and C. Daraio, "Single-test evaluation of directional elastic properties of anisotropic structured materials", *J. Mech. Phys. Solids* **181** (2023), article no. 105471.

[12] M. Genet, "Finite strain formulation of the discrete equilibrium gap principle: application to mechanically consistent regularization for large motion tracking", *C. R. Méc.* **351** (2023), no. G2, pp. 429–458. ISSN: 1873-7234.

[13] L. Crouzeix, J.-N. Périé, F. Collombet and B. Douchin, "An orthotropic variant of the equilibrium gap method applied to the analysis of a biaxial test on a composite material", *Compos. A: Appl. Sci. Manuf.* **40** (2009), no. 11, pp. 1732–1740.

[14] K. T. Kavanagh and R. W. Clough, "Finite element applications in the characterization of elastic solids", *Int. J. Solids Struct.* **7** (1971), no. 1, pp. 11–23.

[15] E. Pagnacco, A.-S. Caro-Bretelle and P. Ienny, "Parameter identification from mechanical field measurements using finite element model updating strategies", in *Full-Field Measurements and Identification in Solid Mechanics*, John Wiley & Sons, 2013, pp. 247–274.

[16] M. Rossi, P. Lava, F. Pierron, D. Debruyne and M. Sasso, "Effect of DIC spatial resolution, noise and interpolation error on identification results with the VFM", *Strain* **51** (2015), no. 3, pp. 206–222. ISSN: 0039-2103, 1475-1305.

[17] S. Avril, M. Grediac and F. Pierron, "Sensitivity of the virtual fields method to noisy data", *Comput. Mech.* **34** (2004), no. 6, pp. 439–452. ISSN: 0178-7675, 1432-0924.

[18] F. Pierron, "Material Testing 2.0: a brief review", *Strain* **59** (2023), no. 3, article no. e12434.

[19] M. Grediac, F. Pierron, S. Avril and E. Toussaint, "The virtual fields method for extracting constitutive parameters from full-field measurements: a review", *Strain* **42** (2006), no. 4, pp. 233–253. ISSN: 0039-2103, 1475-1305.

[20] M. A. Sutton, J. H. Yan, S. Avril, F. Pierron and S. M. Adeeb, "Identification of heterogeneous constitutive parameters in a welded specimen: uniform stress and virtual fields methods for material property estimation", *Exp. Mech.* **48** (2008), no. 4, pp. 451–464. ISSN: 0014-4851, 1741-2765.

[21] S. Avril, F. Pierron, Y. Pannier and R. Rotinat, "Stress reconstruction and constitutive parameter identification in plane-stress elasto-plastic problems using surface measurements of deformation fields", *Exp. Mech.* **48** (2008), no. 4, pp. 403–419. ISSN: 0014-4851, 1741-2765.

[22] G. Louedec, F. Pierron, M. A. Sutton and A. P. Reynolds, "Identification of the local elasto-plastic behavior of FSW welds using the virtual fields method", *Exp. Mech.* **53** (2013), no. 5, pp. 849–859. ISSN: 0014-4851, 1741-2765.

[23] M. Rossi, F. Pierron and M. Štamborská, "Application of the virtual fields method to large strain anisotropic plasticity", *Int. J. Solids Struct.* **97-98** (2016), pp. 322–335. ISSN: 00207683.

[24] P. Ladevèze, M. Reynier and N. Maia, "Error on the constitutive relation in dynamics", *Inverse Probl. Eng.* (1994), pp. 251–256.

[25] M. B. Azzouna, P. Feissel and P. Villon, "Robust identification of elastic properties using the modified constitutive relation error", *Comput. Methods Appl. Mech. Eng.* **295** (2015), pp. 196–218.

[26] P. Feissel and O. Allix, "Modified constitutive relation error identification strategy for transient dynamics with corrupted data: the elastic case", *Comput. Methods Appl. Mech. Eng.* **196** (2007), no. 13-16, pp. 1968–1983.

[27] H. D. Bui, A. Constantinescu and H. Maigre, "Numerical identification of linear cracks in 2D elastodynamics using the instantaneous reciprocity gap", *Inverse Probl.* **20** (2004), no. 4, article no. 993.

[28] C. Patte, *Personalized pulmonary mechanics: modeling, estimation and application to pulmonary fibrosis*, PhD thesis, Institut Polytechnique de Paris, 2020.

[29] E. Berberoğlu, C. Stoeck, P. Moireau, S. Kozerke and M. Genet, "In-silico study of accuracy and precision of left-ventricular strain quantification from 3D tagged MRI", *PLoS One* **16** (2021), no. 11, article no. e0258965.

[30] M. Genet, C. Stoeck, C. Von Deuster, L. C. Lee and S. Kozerke, "Equilibrated warping: Finite element image registration with finite strain equilibrium gap regularization", *Med. Image Anal.* **50** (2018), pp. 1–22. ISSN: 13618415.

[31] E. Berberoğlu, C. Stoeck, P. Moireau, S. Kozerke and M. Genet, "Validation of finite element image registration-based cardiac strain estimation from magnetic resonance images", *PAMM* **19** (2019), no. 1, article no. e201900418.

[32] H. Leclerc, J.-N. Périé, S. Roux and F. Hild, "Voxel-scale digital volume correlation", *Exp. Mech.* **51** (2011), pp. 479–490.

[33] A. Mendoza, J. Neggers, F. Hild and S. Roux, "Complete mechanical regularization applied to digital image and volume correlation", *Comput. Methods Appl. Mech. Eng.* **355** (2019), pp. 27–43.

[34] Z. Tomivcević, F. Hild and S. Roux, "Mechanics-aided digital image correlation", *J. Strain Anal. Eng. Design* **48** (2013), no. 5, pp. 330–343.

[35] K. Škardová, M. Rambausek, R. Chabiniok and M. Genet, "Mechanical and imaging models-based image registration", in *VipIMAGE 2019: Proceedings of the VII ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing, October 16–18, 2019, Porto, Portugal*, Springer, Cham, 2019, pp. 77–85.

[36] E. Berberoğlu, C. Stoeck, S. Kozerke and M. Genet, "Quantification of left ventricular strain and torsion by joint analysis of 3D tagging and cine MR images", *Med. Image Anal.* **82** (2022), article no. 102598.

[37] L. Lee and M. Genet, "Validation of equilibrated warping—image registration with mechanical regularization—on 3D ultrasound images", in *Functional Imaging and Modeling of the Heart: 10th International Conference, FIMH 2019, Bordeaux, France, June 6–8, 2019, Proceedings 10*, Springer, Cham, 2019, pp. 334–341.

[38] Y. Mei, *VFM-for-hyperelastic-materials*, 2022. Online at https://github.com/meiyue1989/VFM-for-hyperelastic-materials.

[39] M. Genet, C. Patte, M. Manoochehrtayebi and A. Peyraut, *dolfin_mech*, 2024. Online at https://doi.org/10.5281/zenodo.10533833. Version 2024.01.19.

[40] M. Genet, *dolfin_warp*, 2023. Version 2023.06.06.post1. https://doi.org/10.5281/zenodo.8010786.

[41] A. Logg and G. N. Wells, "DOLFIN: automated finite element computing", *ACM Trans. Math. Softw.* **37** (2010), no. 2, pp. 1–28.

[42] M. S. Alnaes, J. Blechta, J. Hake, et al., "The FEniCS Project Version 1.5", *Arch. Numer. Softw.* **3** (2015), no. 100, pp. 9–23.

[43] W. Schroeder, K. Martin and B. Lorensen, *The Visualization Toolkit*, 4th edition, Kitware, New York, 2006. ISSN: 978-1-930934-19-1.

[44] A. Peyraut and M. Genet, *dolfin_estim*, 2024. Online at https://gitlab.inria.fr/apeyraut/dolfin%5C_estim.

[45] A. Marek, F. M. Davis and F. Pierron, "Sensitivity-based virtual fields for the non-linear virtual fields method", *Comput. Mech.* **60** (2017), pp. 409–431.