# Parallel inexact Newton-Krylov and quasi-Newton solvers for nonlinear elasticity

Nicolás A. Barnafi[a], Luca F. Pavarino[a], Simone Scacchi[b]

[a]*Department of Mathematics, Università di Pavia, Via Ferrata 1, 27100 Pavia, Italy.* {nicolas.barnafi, luca.pavarino}@unipv.it
[b]*Department of Mathematics, Università di Milano, Via Saldini 50, 20133 Milano, Italy.* simone.scacchi@unimi.it

**Abstract**

In this work, we address the implementation and performance of inexact Newton-Krylov and quasi-Newton algorithms, more specifically the BFGS method, for the solution of the nonlinear elasticity equations, and compare them to a standard Newton-Krylov method. This is done through a systematic analysis of the performance of the solvers with respect to the problem size, the magnitude of the data and the number of processors in both almost incompressible and incompressible mechanics. We consider three test cases: Cook's membrane (static, almost incompressible), a twist test (static, incompressible) and a cardiac model (complex material, time dependent, almost incompressible). Our results suggest that quasi-Newton methods should be preferred for compressible mechanics, whereas inexact Newton-Krylov methods should be preferred for incompressible problems. We show that these claims are also backed up by the convergence analysis of the methods. In any case, all methods present adequate performance, and provide a significant speed-up over the standard Newton-Krylov method, with a CPU time reduction exceeding 50% in the best cases.

*Keywords:* Nonlinear elasticity, Newton-Krylov, Inexact-Newton, BFGS, Scalable solvers

## 1. Introduction

Nonlinear elasticity is a continuum framework for modeling the deformation of an elastic body, and has a large spectrum of applications such as iron, rubber and gels as well as living materials such as skin, muscle and bone. The flexibility of continuum mechanics resides in its abstract formulation starting from fundamental principles, which allows for an accurate representation of many different materials through constitutive modeling and complex boundary conditions [Hol02]. After a precise physical phenomenon has been devised and modeled, the resulting equations yield a complex system of nonlinear Partial Differential Equations (PDEs). Limited progress has been achieved in terms of its numerical analysis [CD04, ADVL$^+$13], so it is difficult to know a-priori efficient strategies for its numerical approximation. Indeed, the problem is not convex but only polyconvex [Cia21], which deteriorates the performance of many well established numerical methods. The gold standard is Newton's method [ZTNZ77], consisting of a second order approximation of the associated variational principle (or a first order approximation of the Euler-Lagrange equations, referred to as the Newton-Raphson method). This method is popular mainly due to its robustness and low iteration count, as it converges quadratically whenever a good initial guess of the solution is considered [WN99]. Its main drawback is that it requires the repeated assembly of the Jacobian matrix, associated to the tangent problem, which usually becomes a bottleneck of the solution process (see [JHN11] for an example in CFD). This can be more evidently appreciated when using higher order approximations, required for avoiding numerical locking effects [EG13].

In general, not much attention has been given in the community to other methods, such as linearly convergent (gradient descent [Rud16], Richardson [Saa03]) and superlinearly convergent (quasi-Newton [WN99], inexact-Newton [DES82]) ones. These methods yield a higher iteration count due to their reduced convergence rate, but can potentially present a drastically overall reduced computational complexity. Indeed, the use of different nonlinear solvers has shown enormous speed-ups in other problems [SHOL06, BRKN17],

where the properties of the equations have been exploited to devise a more adequate nonlinear solution method. Some recent works have numerically explored quasi-Newton methods for the mechanics [LLS05, GP88, LBK17], with their main focus being on the nonlinear iterations and CPU time. This type of study does not take into account the linear system involved in each iteration of the method, and thus is not conclusive with respect to the applicability of these methods in an HPC setting, where problems with millions of degrees of freedom need to be solved, possibly for many time steps. It is well known that iterative methods outperform direct methods in nonlinear elasticity, for example see [EMFTF10] for a comparative study using quadratic finite elements. A numerical method can be deemed adequate in such case if at least it satisfies the following requirements: (i) the nonlinear and linear iterations incurred during the solution procedure are independent of the mesh size used, (ii) the method is robust with respect to the model parameters, at least in the scenarios of interest and (iii) the method is (strongly) scalable, meaning that the overall solution time improves when more processors are used. We finally highlight [WDE07], where a novel conjugate Newton method was proposed and analysed for nonlinear elasticity.

It is important to highlight that in all of the aforementioned approaches, the nonlinearity is treated implicitly, meaning that it is differentiated together with the other terms. This is of course inevitable in quasi-static formulations, but if the inertia is not neglected, it is possible to circumvent the complex nonlinearity by using an explicit time discretization for the nonlinear terms. This effectively means to delay in time the nonlinearity at the cost of using smaller time step. This approach has been proved to be convergent in linear elastodynamics [Wu06], and it has also been used in a stabilized, lowest-order mixed formulation in [LRCC15].

Our application of interest is cardiac mechanics, for which we have considered a dedicated test. Work regarding the solution of cardiac mechanics has mainly addressed two difficulties: the first one refers to the numerical instabilities that arise when considering quasi-incompressible or incompressible materials, for which stabilized formulations have been proposed that address this problem [LNLS14, KGH$^+$22]. The second one pertains the development of efficient preconditioners for the tangent problem solved at each Newton iteration. In this context, work has been devoted to both mixed [CDSS18] and primal formulations [CFPS15, PSZ15]. In all of the aforementioned works, only a Newton method is considered for the solution of the nonlinear problem at hand.

The scope of this work is to provide a first step in the adequate usage of alternatives to Newton's method for nonlinear elasticity in an HPC infrastructure, where we focus on superlinearly convergent methods as they provide an excellent overall computational complexity [WN99]. We stress that all gradient descent algorithms we tested on a preliminary phase failed, and this is indeed consistent with the literature: there are no works, up to our knowledge, of descent nor fixed point algorithms for nonlinear mechanics. We focus on three tests: (i) Cook's membrane test, an almost incompressible problem, (ii) a twist problem, where we test an incompressible material and (iii) a cardiac modeling test, where we use an idealized left ventricle geometry to model a human heartbeat.

The work is structured as follows: in Section 2 we review the mechanics problem and fix some notation. In Section 3 we describe the nonlinear solvers to be used throughout this study, namely the BFGS and inexact-Newton algorithms. In Section 4 we define the scenarios in which the methods will be tested, together with a thorough description of the linear solver involved in each case. In Section 5 we recall the convergence results of the methods and discuss what their expected performance should be in each of the proposed tests. In Section 6 we show and comment the results obtained from the numerical simulations performed, and we conclude with a discussion of the results in Section 7.

## 2. The hyperelasticity problem

Consider a connected domain $\Omega \subset \mathbb{R}^{d\in\{2,3\}}$ that represents the geometry to be deformed, and define its Dirichlet and Neumann boundaries as $\partial\Omega_D$ and $\partial\Omega_N$ respectively, such that $\overline{\partial\Omega} = \overline{\partial\Omega_D} \cup \overline{\partial\Omega_N}$. We look for

a displacement $\mathbf{d} : [0,T] \times \Omega \to \mathbb{R}^d$ such that it solves the momentum conservation equation

$$\begin{aligned} \ddot{\mathbf{d}} - \operatorname{div} \mathbf{P} &= \mathbf{f} && \text{in } (0,T) \times \Omega, \\ \mathbf{d} &= \mathbf{d}_0 && \text{in } \{0\} \times \Omega, \\ \dot{\mathbf{d}} &= \mathbf{v}_0 && \text{in } \{0\} \times \Omega, \\ \mathbf{P}n &= \mathbf{t} && \text{on } [0,T) \times \partial\Omega_N, \\ \mathbf{d} &= \mathbf{d}_D && \text{on } [0,T) \times \partial\Omega_D, \end{aligned} \tag{1}$$

where $\mathbf{d}_0$ is the initial displacement, $\mathbf{v}_0$ is the initial velocity,$\mathbf{t}$ is the surface traction, $\mathbf{d}_D$ is a distributed load, $\dot{()}$ denotes a time derivative, and $\mathbf{P}$ is known as the Piola stress tensor, usually obtained from a predefined Helmholtz potential $\Psi$ such that

$$\mathbf{P}(\mathbf{F}) = \frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{F}),$$

where $\mathbf{F} = \mathbf{I} + \nabla \mathbf{d}$. The existence of such a potential is actually a hypothesis, and whenever it exists we refer to the material as a *hyperelastic material*. Approximating this problem by neglecting the inertial term $\ddot{\mathbf{d}}$ results in the well-known *static* mechanics, or quasi-static whenever one of the problem's data is time-dependent. If we require the volume to be locally conserved through $J \coloneqq \det \mathbf{F} = 1$, we obtain an *incompressible* elasticity problem. This can be instead approximated by further penalizing the deviation of $J - 1$ from 0, which results in a modified potential

$$\Psi(\mathbf{F}) = \Psi_{\text{sol}}(\bar{\mathbf{F}}) + \Psi_{\text{vol}}(J),$$

where the isochoric component $\bar{\mathbf{F}} = J^{-1/d}\mathbf{F}$ is such that $\det \bar{\mathbf{F}} = 1$. This separates the energetic contribution of volumetric deformation, and is referred to as almost (or quasi-) incompressibility whenever the term $\Psi_{\text{vol}}$ yields $J \approx 1$. More details can be found in [Hol02].

### *2.1. Finite elements approximation*

We briefly show how problem (1) is discretized. We first require the weak form of problem (1), for which we assume that the solution belongs to a Hilbert space $U$. The problem is then given by: Find a displacement $\mathbf{d}$ in a solution space $V = \{\mathbf{v} \in U : \mathbf{v} = \mathbf{d}_D \text{ on } \partial\Omega_D\}$ such that

$$\int_\Omega \ddot{\mathbf{d}} \cdot \mathbf{v} \, dx + \int_\Omega \mathbf{P}(\mathbf{F}) : \nabla \mathbf{v} \, dx = \int_{\partial\Omega_N} \mathbf{t} \cdot \mathbf{v} \, dS + \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx \quad \forall \mathbf{v} \in V_0, \tag{2}$$

where $V_0 = \{\mathbf{v} \in U : \mathbf{v} = \mathbf{0} \text{ on } \partial\Omega_D\}$. The weak form of the incompressible case can be obtained using the Lagrange multipliers technique, more details in [Hol02]. Considering the Hilbert space $Q = L^2(\Omega)$ for scalar multipliers, the incompressible elasticity problem reads: Find a displacement $\mathbf{d}$ in $V$ and a multiplier $p$ in $Q$ such that

$$\begin{aligned} \int_\Omega \ddot{\mathbf{d}} \cdot \mathbf{v} \, dx + \int_\Omega \left(\mathbf{P}(\mathbf{F}) - pJ\mathbf{F}^{-T}\right) : \nabla \mathbf{v} \, dx &= \int_{\partial\Omega_N} \mathbf{t} \cdot \mathbf{v} \, dS + \int_\Omega \mathbf{f} \cdot \mathbf{v} \, dx && \forall \mathbf{v} \in V_0, \\ \int_\Omega q(J-1) \, dx &= 0 && \forall q \in Q. \end{aligned} \tag{3}$$

For the discretization of (2), we consider two conforming finite elements (discrete) spaces $V^h \subset V$ and $V_0^h \subset V_0$ and an implicit time discretization $\ddot{\mathbf{d}} \approx \frac{\mathbf{d}^n - 2\mathbf{d}^{n-1} + \mathbf{d}^{n-2}}{\Delta t^2}$, where $\mathbf{d}(t^n) \approx \mathbf{d}^n$ and $t^0 = 0, \dots, t^N = T$ is a uniform partition of the time interval of interest such that $t^n - t^{n-1} = \Delta t$. With these definitions, the space and time discrete problem can be written as follows: Given two previous displacements $\mathbf{d}_h^{n-1}$ and $\mathbf{d}_h^{n-2}$, find the displacement $\mathbf{d}_h^n$ in $V^h$ such that

$$\int_\Omega \frac{1}{\Delta t^2} \mathbf{d}_h^n \cdot \mathbf{v}_h \, dx + \int_\Omega \mathbf{P}(\mathbf{F}_h) : \nabla \mathbf{v}_h \, dx = \int_{\partial\Omega_N} \mathbf{t} \cdot \mathbf{v}_h \, dS + \int_\Omega \left(\mathbf{f} + \frac{2\mathbf{d}^{n-1} - \mathbf{d}^{n-2}}{\Delta t^2}\right) \cdot \mathbf{v}_h \, dx \quad \forall \mathbf{v} \in V_0^h. \tag{4}$$

The discretization of the incompressible case is analogous. Other time discretizations could be considered as well, such as the well-known Newmark scheme, which is additionally simplectic [Hug12].

## 3. Numerical solvers

In this section we briefly review the nonlinear methods we consider, namely Newton-Krylov and quasi-Newton methods. Details regarding their geometric motivation can be found in [WN99], whereas the convergence properties are detailed in Section 5. Our study is mainly motivated by the applicability of these methods in an HPC infrastructure, so the use of iterative solvers within the nonlinear method is not only inevitable, but also desirable due to their well-established parallel performance. Inspired by this, we have classified our methods according to the degree of exactness of the linear solver they use, so that a fully inexact method will simply consider the action of the associated preconditioner instead of solving a linear system. Instead, a quasi-exact[1] method will consider an accurate linear solver given by small tolerances, such as $10^{-14}$ and $10^{-6}$ absolute and relative tolerances, respectively. Now we provide more details regarding the methods, and in particular emphasize where the linear solver is used.

**Newton-Krylov methods.** These methods can be seen as either a minimization procedure or as a root finding algorithm, the latter often referred to as Newton-Raphson method. We present it as a root finding method. Consider Equation (1) in quasi-static, residual form

$$\mathbf{R}(\mathbf{d}) \coloneqq \mathbf{f} - \operatorname{div} \mathbf{P}(\mathbf{F}) = \mathbf{0},$$

with $\mathbf{P} = \frac{\partial \Psi}{\partial \mathbf{F}}$ for a given Helmholtz potential $\Psi$, and consider an initial iterate $\mathbf{d}^0$. Then, given a previous iteration $\mathbf{d}^{k-1}$, the next one is given as the solution $\delta\mathbf{d}^k$ of the linearized problem

$$\partial_{\mathbf{d}}\mathbf{R}(\mathbf{d}^{k-1})[\delta\mathbf{d}^k] = -\mathbf{R}(\mathbf{d}^{k-1}), \tag{5}$$

where $\partial_{\mathbf{d}}$ stands for the Frechét derivative with respect to $\mathbf{d}$, and the update is then given by $\mathbf{d}^k = \mathbf{d}^{k-1} + \delta\mathbf{d}^k$. Equation (5) gives rise to a linear system of equations often referred to as the tangent problem or simply the linearized problem, which is solved by means of an iterative Krylov space method, accelerated by a preconditioner, see e.g. [CGKT94, CGK+98]. In practice, the most used one is referred to as Newton-MG, meaning that problem (5) is solved with an iterative method, preconditioned by an algebraic multigrid preconditioner (AMG) [Hac13, CFPS15].

**Inexact Newton-Krylov methods.** Problem (5) is an approximation of the actual equation $\mathbf{R}(\mathbf{d}) = 0$, and thus it might not be true that an accurate solution of the tangent problem will also yield an accurate solution of the original equation. In fact, this usually gives rise to over-solving the problem [EW94] in the first iterations, which motivates the use of inexact solvers for (5), i.e. to use large tolerances for the solution of the tangent problem. In addition, the quality of the linearization improves as the iterates are closer to the solution, which motivates the use of adaptive (relative) tolerances. In particular, we use the Eisenstat-Walker strategy [EW96], given by

$$\texttt{tol}^k = \frac{|\|\partial_{\mathbf{d}}\mathbf{R}(\mathbf{d}^{k-1})[\delta\mathbf{d}^k] - \mathbf{R}(\mathbf{d}^{k-1})\| - \|\mathbf{R}(\mathbf{d}^{k-1})\||}{\|\mathbf{R}(\mathbf{d}^{k-1})\|}.$$

The choice of the norm $\|\cdot\|$ is arbitrary, so we consider in what follows the $\ell^2$ norm. This of course makes sense only in the discrete setting, meaning that for each residual vector $\mathbf{r}$ we consider the norm $\|\mathbf{r}\| = \left(\sum_i r_i^2\right)^{1/2}$. The resulting scheme guarantees superlinear convergence [DES82], which is of course worse than the quadratic convergence of a standard Newton-Krylov scheme, but gives an overall reduced complexity as it avoids oversolving the linearized problem.

**BFGS method.** This is a quasi-Newton method, and although it was initially devised as a minimization procedure [WN99], it can also be adapted to be used as a root-finding algorithms [DS96]. For this, we

[1] We emphasize that all methods are inexact given their iterative nature. Still, a sufficiently low tolerance can be regarded as an almost exact solver, so we refer to this as quasi-exact.

look at the minimization principle from which (1) in quasi-static form is obtained:

$$\min_{\mathbf{d}} \Pi(\mathbf{d}) \coloneqq \int_{\Omega} \Psi(\mathbf{F}) - \mathbf{f} \cdot \mathbf{d} \, d\mathbf{x}. \tag{6}$$

A quadratic approximation of this problem yields

$$\Pi(\mathbf{d}) \approx \Pi(\bar{\mathbf{d}}) + \partial_{\mathbf{d}} \Pi(\bar{\mathbf{d}})[\mathbf{d} - \bar{\mathbf{d}}] + \frac{1}{2}[\mathbf{d} - \bar{\mathbf{d}}]^T \partial^2_{\mathbf{d}} \Pi(\bar{\mathbf{d}})[\mathbf{d} - \bar{\mathbf{d}}],$$

and such scheme is indeed equivalent to (5) when applied as an iterative procedure with an exact Hessian. This method requires an initial approximation of the Hessian $\mathbf{B}^0 \approx [\partial^2_{\mathbf{d}} \Pi(\mathbf{d}^0)]^{-1}$, which is then enriched at each iteration with a rank two perturbation that includes curvature information given by

$$\mathbf{B}^{k+1} = (\mathbf{I} - \rho_k \mathbf{s}^k \otimes \mathbf{y}^k)\mathbf{B}^k(\mathbf{I} - \rho_k \mathbf{y}^k \otimes \mathbf{s}^k) + \rho_k \mathbf{s}^k \otimes \mathbf{s}^k,$$

where $\mathbf{s}^k = \mathbf{x}^{k+1} - \mathbf{x}^k$, $\mathbf{y}^k = \mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^k)$, and $\rho^k = 1/\langle \mathbf{s}^k, \mathbf{y}^k \rangle$ with $\langle \mathbf{a}, \mathbf{b} \rangle = \sum_i a_i b_i$. The approximate Hessian is then used to compute the next iteration as in Equation (5):

$$\delta \mathbf{d}^k = -\mathbf{B}^{k+1} \partial_{\mathbf{d}} \Pi(\mathbf{d}^{k-1}).$$

The action of $\mathbf{B}^{k+1}$ is implemented by means of a two-level recursion that allows for a limited memory implementation [Noc80]. This method, as the inexact Newton-Krylov, yields superlinear convergence, and has the additional advantage of requiring only one assembly of the Hessian matrix (or any other initial matrix). The initial approximation of the Hessian is critical for the convergence and performance of this method, and we have indeed observed that standard simpler approaches do not yield satisfactory results for this method in the context of nonlinear mechanics. Motivated by this, we leverage the preconditioners obtained from the initial matrix $\partial^2_{\mathbf{d}} \Pi(\mathbf{d}^0)$ to obtain better approximations of the Hessian. If we denote by $\mathbf{P}$ the preconditioner arising from the initial Hessian, we consider the following approximations:

- $\mathbf{B}^0$ = action of $\mathbf{P}$ obtained from Hessian matrix (BFGS-preonly)
- $\mathbf{B}^0$ = inexact Krylov solver, large ($\approx 10^{-2}$) relative tolerance (inexact BFGS)
- $\mathbf{B}^0$ = quasi-exact Krylov solver, small ($\approx 10^{-6}$) relative tolerance (quasi-exact BFGS)

We have added on the right the names corresponding to Figure 1. We note that the quasi-exact and inexact versions are not covered by the theory, as the action of $\mathbf{B}^0$ changes at each iteration. This happens because the number of linear iterations required at each nonlinear iteration varies according to the tolerance. We highlight that the use of a fixed tolerance in the inexact scenario is considered for simplicity, as an equivalent Eisenstat-Walker type of adaptive tolerance could be considered. This is an interesting alternative and by no means a trivial one to set up, so we leave this for future work.

We show a classification of these methods according to the exactness of the linear solver in Figure 1, where we have fixed three degrees of exactness: preconditioner only (preonly[2]), inexact and quasi-exact. The two families present different types of performance: the Newton family relies on a low iteration count, where the level of inexactness relaxes the linear solver at each iteration, at the cost of some additional Newton steps. The quasi-Newton family instead yields a larger number of iterations, but each iteration is much cheaper than a Newton iteration, as it does not requires the Jacobian matrix to be reassembled. The level of exactness in this case reduces the iteration count, at the cost of making the iterations more expensive. The extreme cases Newton-preonly and quasi-exact BFGS have shown to be noncompetitive in our experiments, so we do not consider them in what follows. Throughout the remaining parts of this work, whenever no confusion

[2]The name is the same as the PETSc option to by-pass the iterative solver.

arises, we may refer to Newton-Krylov and inexact Newton-Krylov methods as simply Newton methods. Throughout this manuscript, we will denote the Newton-Krylov, inexact Newton-Krylov, BFGS-preonly (or simply BFGS) and inexact-BFGS methods as **NK**, **iNK**, **B** and **iB** respectively, as shown in Figure 1. We also show in Appendix A the commands used to run each of the four methods under consideration.

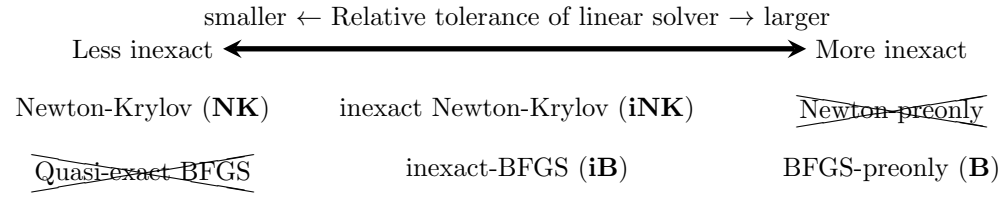



Figure 1: Description of exactness level in the linear solver and nonlinear solvers used in the tests, where we denote "preconditioner only" as "preonly". Newton-preonly and quasi-exact BFGS have been discarded from our experiments because they are not competitive: the first requires too many matrix assembles, the second spends too much time at each iteration.

## 4. Test problems and solvers details

In this section, we describe the test cases we use to compare the performance of each method. The scope of the heartbeat test is that of providing a more realistic scenario to test the methods. Details on the implementation will be given on each problem.

**Cook test.** This is a static almost incompressible mechanics benchmark [PM16]. We use homogeneous Dirichlet conditions on $\{x = 0\}$, a vertical traction given by $\mathbf{t} = (0, \tau, 0)$ with $\tau = 10^6$ in deformed configuration on $\{x = L\}$, a homogeneous distributed load $\mathbf{f} = \mathbf{0}$, and homogeneous Neumann conditions on the remaining parts of the boundary. The constitutive modeling is given by a almost incompressible Neo-Hookean material:

$$\Psi_{\text{Cook}}(\mathbf{F}) = C_1 \left(\text{tr}\left(\bar{\mathbf{F}}^T \bar{\mathbf{F}}\right) - 3\right) + k\left([\det(\mathbf{F})]^2 - 1 - 2\log(\det \mathbf{F})\right),$$

where $C_1 = \frac{1}{2}\mu$, $\mu = 8.194 \cdot 10^7$, $k = \lambda + \frac{2}{3}\mu$, $\lambda = 2\mu\nu/(1-2\nu)$, and $\nu = 0.3$. The Krylov solver was configured with a GMRES linear solver without restart and with a modified Gram-Schmidt procedure which is more robust [GVL96], preconditioned with HYPRE-BoomerAMG [FY02], using its default configuration and setting the matrix block size to 3 in PETSc to improve the efficiency of the preconditioner. The absolute tolerance was set to $10^{-14}$, whereas the relative tolerance was set to $10^{-6}$ for the exact solver and to $10^{-1}$ for the inexact ones. We show the solution in Figure 2 (a).

**Twist test.** This is a static incompressible mechanics problem [BGO15], where we use the inf-sup stable stable finite elements $\mathbb{P}_2 - \mathbb{P}_0$ for the approximation of the displacement and pressure. We use discontinuous elements for the pressure instead of linear ones since the accuracy of the pressure field approximation is not the focus of this work. Boundary conditions are given by a homogeneous on $\{z = 0\}$, a $\pi/6$ rotation in the $x$ and $y$ components on $\{z = L_z\}$, homogeneous data $\mathbf{f} = \mathbf{t} = \mathbf{0}$, and homogeneous Neumann conditions on the remaining parts of the boundary. The constitutive modelling is given by the following polyconvex potential:

$$\Psi_{\text{Twist}}(\mathbf{F}, p) = \alpha_p(\mathbf{F} : \mathbf{F} - 3) + \beta_p(\text{cof}\,\mathbf{F} : \text{cof}\,\mathbf{F} - 3) - (4\beta_s + 2\alpha_s)\log(\mathbf{F}) - p(\det(\mathbf{F}) - 1),$$

where $\alpha_p = \beta_p = \alpha_s = \beta_s = 9000$ and $\text{cof}\,\mathbf{F} := \det(\mathbf{F})\mathbf{F}^{-T}$. The logarithmic term with $\det(\mathbf{F})$ is required because of the weak imposition of the incompressibility constraint [ADVL+13]. The preconditioning of a saddle point problem is more challenging, so to obtain a more robust performance we used a lower Schur complement preconditioner based on a $(\mathbf{d}, p)$ field split [BKM+12], we provide details on the Schur complement preconditioner in Appendix B. The displacement block uses an AMG preconditioner, the Schur complement block instead uses the SIMPLE preconditioner from fluid mechanics

[EHS+06], meaning that the inverse of the displacement block is approximated by the inverse of its diagonal, and for the resulting block we used a block Jacobi preconditioner, which we observed to suffice. Two additional comments are in place: (i) to calibrate the effectiveness of each solver in the Schur complement, we started from using a direct solver in each block to guarantee convergence in at most two iterations [Man90], and started relaxing the blocks from there. The difficulty of the problem is predominantly the displacement block, which can be appreciated by the simple preconditioner used for the Schur complement block; (ii) the pressure block presents much less degrees of freedom (DoFs), so we used a telescopic approach that creates an MPI subcommunicator with a fixed 25% of the original processes to avoid excessive communication. We show the solution in Figure 2 (b).

**Heartbeat test.** This problem represents our application of interest. It models the contraction of a human left ventricle in an idealized geometry given by a prolate ellipsoid, with a pointwise set of coordinates representing the muscle fiber orientation $(\mathbf{f}_0, \mathbf{s}_0, \mathbf{t}_0)$ obtained through rule based methods [BBPT12]. We consider a Guccione hyperelastic potential [GMW91], given by

$$\Psi(\mathbf{F}) = \frac{C}{2}\left(e^{Q(\bar{\mathbf{F}})} - 1\right) + \frac{B}{2}(J-1)\log J,$$
$$Q(\mathbf{F}) = b_{ff}E_{ff}^2 + b_{ss}E_{ss}^2 + b_{nn}E_{nn}^2 + b_{fs}(E_{fs}^2 + E_{sf}^2) + b_{fn}(E_{fn}^2 + E_{nf}^2) + b_{sn}(E_{sn}^2 + E_{ns}^2),$$

where $\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^T\mathbf{F} - \mathbf{I}\right)$ and $E_{ab} = \mathbf{E}\mathbf{a}\cdot\mathbf{b}$ for $a, b \in \{f, s, n\}$, which are the components of $\mathbf{E}$ in the fiber-induced frame of reference $\{\mathbf{f}_0, \mathbf{s}_0, \mathbf{n}_0\}$. See [GMW91] for reference values of the related parameters. The Piola stress tensor is enriched with a fiber-wise force known as active stress that models the contraction of the muscle cells (cardiomyocites):

$$\mathbf{P}(\mathbf{F}) = \frac{\partial\Psi}{\partial\mathbf{F}}(\mathbf{F}) + \mathbf{P}_{\text{act}}(\mathbf{F}), \quad \mathbf{P}_{\text{act}}(\mathbf{F}) = \gamma(t)\frac{(\mathbf{F}\mathbf{f})\otimes\mathbf{f}}{|\mathbf{F}\mathbf{f}|},$$

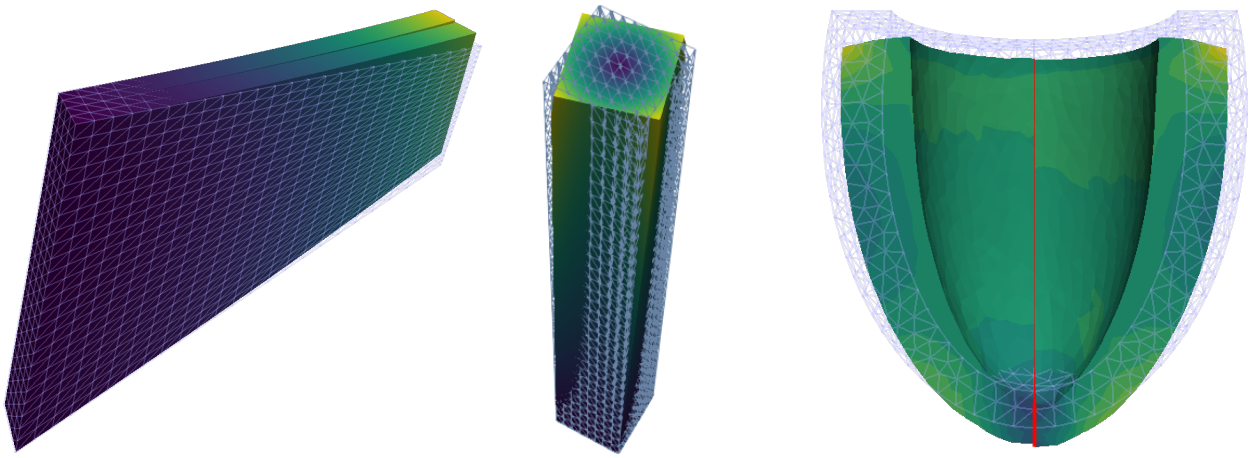
where $\gamma$ is a given function known as the activation function. For simplicity we consider an analytical activation given by

$$\gamma(t) = C_{\text{PA}}\max\{\sin(2\pi t/T), 0\},$$

where the peak activation constant is $C_{\text{PA}} = 10^4$ and the period is $T = 0.8$. More details on the generation of the fibers and the activation function can be found in [BBPT12] and [RSA+20] respectively. Boundary conditions are given by homogeneous Neumann on the endocardium $\partial\Omega_{\text{endo}}$ and the base $\partial\Omega_{\text{base}}$, whereas the epicardium $\partial\Omega_{\text{epi}}$ considers a Robin condition that models the friction of the epicardium with the pericardium [ULM02] and is given by

$$\mathbf{P}(\mathbf{F})\mathbf{n} =: -(\mathbf{n}\otimes\mathbf{n})\left(K_{\perp}^{\text{epi}}\mathbf{d} + C_{\perp}^{\text{epi}}\dot{\mathbf{d}}\right) - (\mathbf{I} - \mathbf{n}\otimes\mathbf{n})\left(K_{\parallel}^{\text{epi}}\mathbf{d} + C_{\parallel}^{\text{epi}}\dot{\mathbf{d}}\right) = \mathbf{0} \quad \text{on } \partial\Omega_{\text{epi}}. \tag{7}$$

More information of the physical meaning of these boundaries and the motivation for these choices can be found in [QLRRB17]. The Krylov solver in this case is identical to the one used in the Cook test. We show the solution at $t = 0.2$ in Figure 2 (c).



(a) Cook test. (b) Twist test. (c) Heartbeat test.

Figure 2: Solutions of the benchmark tests used. Color is scaled according to displacement norm $|\mathbf{d}|$, and the reference configuration is depicted through its mesh. In (a), the figure has been clipped to show the solutions using both $\mathbb{P}_1$ (front) and $\mathbb{P}_2$ (back) finite elements. Analogously, in (c) we show on the left of the red line the solution with $\mathbb{P}_1$, and on the right of the red line instead we show the solution with $\mathbb{P}_2$. Additionally, we have discretized the displacement magnitude in (c) to clarify the distinction between both solutions, as the difference is not clear when looking only at the deformed states.

## 5. Theoretical results and solvers expected behavior

The purpose of this section is to show that the hypotheses from the convergence analysis, despite not being directly applicable to this problem, provide fundamental insight to guide the choice of the nonlinear solver. To state the results we consider an arbitrary scalar function $f : \mathbb{R}^m \to \mathbb{R}$ to be minimized and an initial point $\mathbf{x}^0$. The results and their proofs can be found in [WN99]. First we state the convergence result of Newton's method, where we remark that the inexact variant requires no additional hypotheses.

**Theorem 1 (Newton methods).** *Assume that there exists a point $\mathbf{x}^*$ such that $\nabla f(\mathbf{x}^*) = 0$, $\nabla^2 f$ is Lipschitz continuous in a neighborhood of $\mathbf{x}^*$ and $\nabla^2 f(\mathbf{x}^*)$ is positive definite. Then, $\mathbf{x}^*$ is a local minimizer and, if the initial guess $\mathbf{x}^0$ is sufficiently close to $\mathbf{x}^*$, the iterates obtained by Newton's method converge quadratically to $\mathbf{x}^*$.*

Now we state the general convergence theorem for the BFGS method.

**Theorem 2 (Quasi-Newton methods).** *Assume that $f$ is twice continuously differentiable and that $\nabla^2 f$ is Lipschitz continuous at the minimizer $\mathbf{x}^*$. Assume also that the level set*

$$\mathcal{L} = \{\mathbf{x} \in \mathbb{R}^m : f(\mathbf{x}) \leq f(\mathbf{x}^0)\}$$

*is convex, that $\nabla^2 f$ is positive definite on $\mathcal{L}$ and that the initial matrix $\mathbf{B}^0$ is positive definite. Then, the sequence of iterates obtained by the BFGS method converges superlinearly to $\mathbf{x}^*$.*

We make the following observations, which are fundamental to understand the performance of these methods.

- Both methods require the initial point to be sufficiently close to a solution. This can be seen in the BFGS method through the convexity of $\mathcal{L}$.
- The BFGS method is better suited for convex problems, given that the Hessian is required to be definite positive not only at the minimizer as Newton, but also in the entire level set $\mathcal{L}$.

- Inexact Newton-Krylov convergence holds under the same hypotheses of Newton's method, plus some hypotheses on the relative tolerances $\eta_k$ that guarantee the superlinear convergence. We consider tolerances that satisfy such hypotheses [EW94], and note that they can sometimes yield global convergence, unlike Newton's method. This makes inexact Newton-Krylov methods attractive as they can be potentially more robust than a classic Newton method.
- The initial matrix approximation $\mathbf{B}^0$ is variable in the inexact BFGS method, meaning that its performance might not necessarily be better, nor more robust, than the case in which only the action of the preconditioner is considered. This is due to this case not being covered by the theory, so that in principle there could be additional hypotheses on the action of a variable $\mathbf{B}^0$.
- Notes for different types of problems:
  - **Cook test:** In this case, the problem being solved is polyconvex, but if an initial guess is considered that is sufficiently close to a solution, we can possibly fall in an attraction basin, i.e. a locally convex area. This means that all methods methods should converge whenever a sufficiently small load is considered, but outside of this setting it may be observed that inexact-BFGS and BFGS are less robust due to the stronger convexity requirement of $\mathcal{L}$.
  - **Twist test:** This is a saddle point problem, so BFGS is not guaranteed to converge even if a sufficiently good initial approximation is considered. This can be seen from the positive definiteness hypothesis of both the initial Hessian $\mathbf{B}^0$ and the Hessian on $\mathcal{L}$.
  - **Heartbeat test:** The inertia term in elastodynamics results in a convex contribution to the variational principle associated to the problem, so we expect this case to be easier to solve in spite of the more complex nonlinearities involved. Instead, the nonlinearities should impact the number of nonlinear iterations.

We note that our analysis focuses on properties related to convex analysis: local convexity within polyconvex materials and non-convexity of saddle-point problems as they represent min-max problems. There are two difficulties that we do not address within this study: the first one is the presence of non-conservative forces, which do not allow for a variational principle such as (6). Previous works have shown that the increased numerical costs due to this difficulty are negligible, while additional difficulties stem from the incompressible formulation. The second difficulty is buckling, i.e. the bifurcation observed when using incremental techniques such as ramping. This is a well-known issue which can make numerical studies extremely difficult, since it is not clear if the phenomena observed are due to the problem complexity or rather to the prescence of a bifurcation. To avoid this difficulty, we do not consider ramping techniques in this work, which explains why the solutions for the Cook and Twist problems in Figure 2 present such small deformations when compared to other studies. The numerical approximation of nonlinear elasticity problems in such contexts requires more involved formulations [GCM14], whose efficient numerical approximation has been studied in [FBF15] using deflation techniques. The study of robust methods within a deflation-based solver for finding distinct solutions of nonlinear PDEs remains an open challenge.

## 6. Numerical results

In this section, we compare the performance of the solvers introduced in Sec. 3, for all tests under consideration. The scope of these tests is to assess the sensitivity of the methods with respect to the problem size, some of the problem parameters, and the number of computational cores. For the heartbeat problem, we consider four different meshes with the degrees of freedom shown in Table 1 and a timestep given by $\Delta t = 0.01$. We note that meshes 3 and 4 are refinements of meshes 1 and 2 respectively. Note also that since BFGS-preonly does not solve any linear system, we do not report the linear iterations for this method. All models are implemented using the FEniCS library [ABH+15], and all interfaces with the underlying PETSc library are done by using the `petsc4py` interface [BAA+21]. The computations were performed on the EOS, INDACO and Galileo100 supercomputers.

| Mesh name | DoFs $\mathbb{P}_1$ | DoFs $\mathbb{P}_2$ |
|---|---|---|
| Mesh 1 | 9375 | 60081 |
| Mesh 2 | 20709 | 149511 |
| Mesh 3 | 60081 | 421191 |
| Mesh 4 | 149511 | 1123515 |

Table 1: Heartbeat test: Degrees of freedom yielded by each of the meshes under consideration. Note that the DoFs from second order and first order elements match between the meshes 1-3 and 2-4 respectively. This happens because Mesh 3 is a refinement of Mesh 1, and Mesh 4 is a refinement of Mesh 2.

| $\mathbb{P}_1$ discretization | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DoFs | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 9375 | 3 | 20.3 | 1.1 | 4 | 8.5 | 1.0 | 5 | 9.0 | 0.7 | 34 | 1.0 |
| 64827 | 3 | 22.0 | 14.7 | 4 | 9.0 | 11.8 | 6 | 10.2 | 12.4 | 39 | 13.1 |
| 207831 | 3 | 23.3 | 47.7 | 4 | 9.8 | 40.2 | 6 | 9.7 | 36.8 | 41 | 40.5 |
| 479859 | 3 | 23.3 | 116.4 | 4 | 8.8 | 89.9 | 6 | 10.3 | 88.1 | 40 | 90.4 |
| 922383 | 3 | 24.0 | 235.1 | 5 | 11.8 | 252.0 | 6 | 9.5 | 162.8 | 42 | 180.8 |
| 1576875 | 3 | 23.7 | 406.3 | 5 | 12.0 | 432.8 | 6 | 10.3 | 302.4 | 43 | 327.3 |
| $\mathbb{P}_2$ discretization | | | | | | | | | | | |
| DoFs | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 9375 | 3 | 37.0 | 3.5 | 5 | 16.4 | 3.6 | 6 | 15.3 | 2.6 | 93 | 4.2 |
| 64827 | 3 | 36.3 | 40.7 | 5 | 15.8 | 37.3 | 6 | 14.0 | 27.9 | 86 | 41.4 |
| 207831 | 3 | 35.0 | 131.7 | 5 | 15.8 | 128.0 | 6 | 13.3 | 88.7 | 82 | 124.5 |
| 479859 | 3 | 34.7 | 316.7 | 5 | 16.0 | 312.6 | 6 | 13.7 | 212.8 | 81 | 288.3 |
| 922383 | 3 | 35.3 | 638.1 | 5 | 16.0 | 631.9 | 6 | 13.7 | 426.8 | 82 | 578.8 |
| 1576875 | 3 | 34.3 | 1128.0 | 5 | 15.8 | 1087.5 | 6 | 13.5 | 822.3 | 81 | 1076.5 |

Table 2: Sensitivity with respect to problem size, Cook test. nit := nonlinear iterations, lit := average linear iterations per nonlinear iteration, $T_{sol}$ := CPU time in seconds.

### *6.1. Sensitivity with respect to problem size*

To compare the performance of all methods, we report the total nonlinear iterations counts, the average Krylov iterations per nonlinear step and the solution time as we increase the number of degrees of freedom.

**Cook test.** We present the nonlinear iterations, linear iterations and CPU time in Table 2, varying the dimension of the problem from about 10 thousand to 1.5 million degrees of freedom. We first observe that notably all methods exhibit a robust behavior with respect to the degrees of freedom, except for BFGS-preonly with first order elements, which presents a very mild increase in the nonlinear iterations. Indeed, the total and average linear iterations reported are roughly constant, but in general the inexact-BFGS yields lower average linear iterations. All methods exhibit an approximately linear increase of CPU time with respect to the degrees of freedom. In general, BFGS methods outperform Newton methods in terms of the execution time, and in fact the ratio between the CPU times of the best method (BFGS-preonly) and Newton's method in the case with the largest amount of DoFs is 0.74 for $\mathbb{P}_1$ and 0.73 for $\mathbb{P}_2$ elements.

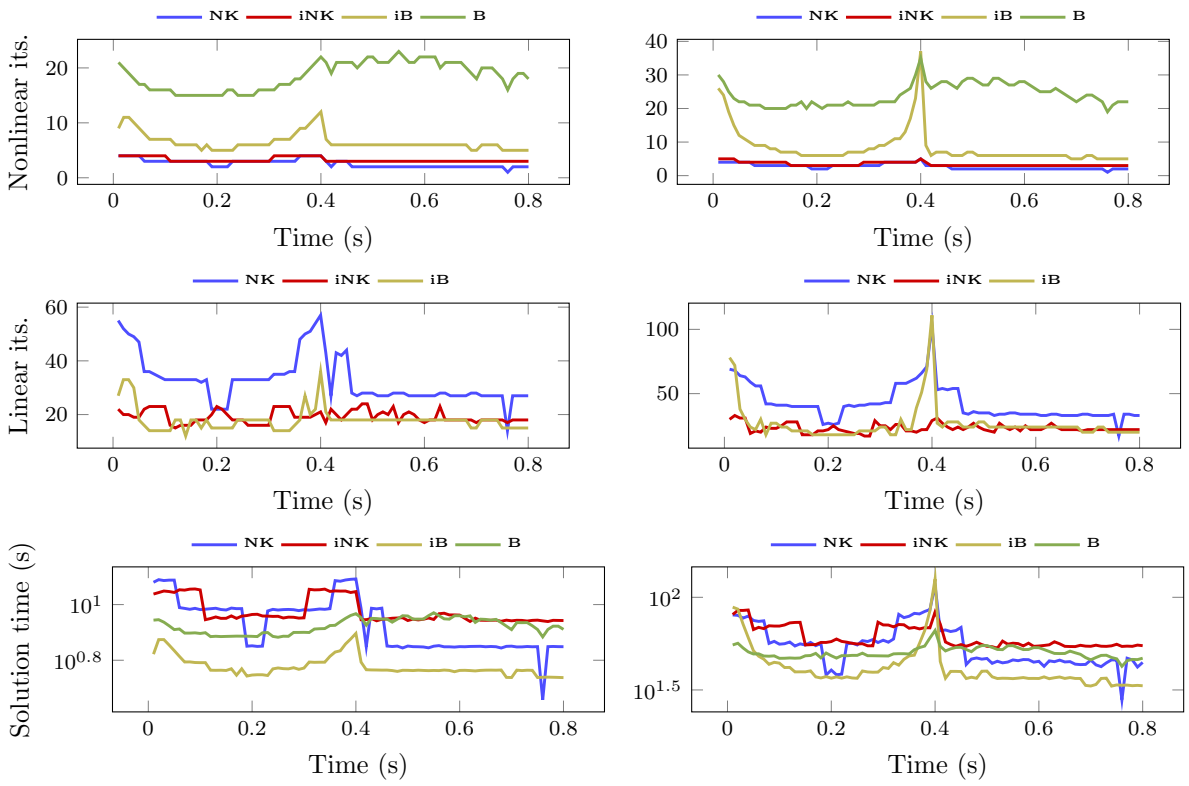
**Twist test.** We report the nonlinear iterations, average linear iterations per nonlinear step, and CPU times in Table 3. We observe that BFGS methods do not converge for this problem, so we can not report

| DoFs | **NK** | | | **iNK** | | |
|---|---|---|---|---|---|---|
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ |
| 131163 | 4 | 426.3 | 1924.0 | 13 | 109.2 | 1694.7 |
| 223347 | 5 | 546.0 | 5808.0 | 13 | 135.2 | 3869.9 |
| 350955 | 5 | 669.2 | 12052.7 | 14 | 180.8 | 9334.0 |
| 519747 | 6 | 800.0 | 27207.8 | 15 | 205.1 | 17689.0 |

Table 3: Sensitivity with respect to problem size, Twist test using $\mathbb{P}_2$ elements. nit := nonlinear iterations, lit := average linear iterations per nonlinear iteration, $T_{sol}$ := CPU time in seconds.

| $\mathbb{P}_1$ discretization | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DoFs | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 9375 | 3.3 | 13.2 | 3.7 | 3.7 | 5.4 | 3.7 | 8.5 | 2.8 | 2.4 | 19.3 | 3.1 |
| 20709 | 3.5 | 12.1 | 10.9 | 4.0 | 5.4 | 11.2 | 8.6 | 2.4 | 6.7 | 17.6 | 8.3 |
| 60081 | 3.5 | 14.5 | 33.4 | 4.0 | 6.1 | 32.8 | 9.8 | 2.8 | 20.9 | 22.1 | 26.6 |
| 149511 | 3.5 | 13.6 | 92.8 | 3.9 | 5.6 | 90.5 | 9.9 | 2.7 | 61.3 | 21.1 | 73.5 |
| $\mathbb{P}_2$ discretization | | | | | | | | | | | |
| DoFs | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 60081 | 3.6 | 17.0 | 30.6 | 4.1 | 6.8 | 25.4 | 14.0 | 3.0 | 21.5 | 26.8 | 19.4 |
| 149511 | 3.7 | 15.0 | 71.1 | 4.4 | 5.8 | 75.0 | 14.4 | 2.5 | 58.5 | 23.3 | 50.6 |
| 421191 | 4.1 | 18.7 | 229.8 | 4.7 | 7.0 | 242.0 | – | – | – | 31.5 | 177.3 |
| 1123515 | – | – | – | 4.6 | 6.7 | 677.0 | 17.9 | 3.3 | 634.8 | 32.2 | 525.6 |

Table 4: Sensitivity with respect to problem size, Heartbeat test. nit := nonlinear iterations, lit := average linear iterations per nonlinear iteration, $T_{sol}$ := CPU time in seconds. A unique number in each quantity is obtained by averging the results from the first 10 timesteps.



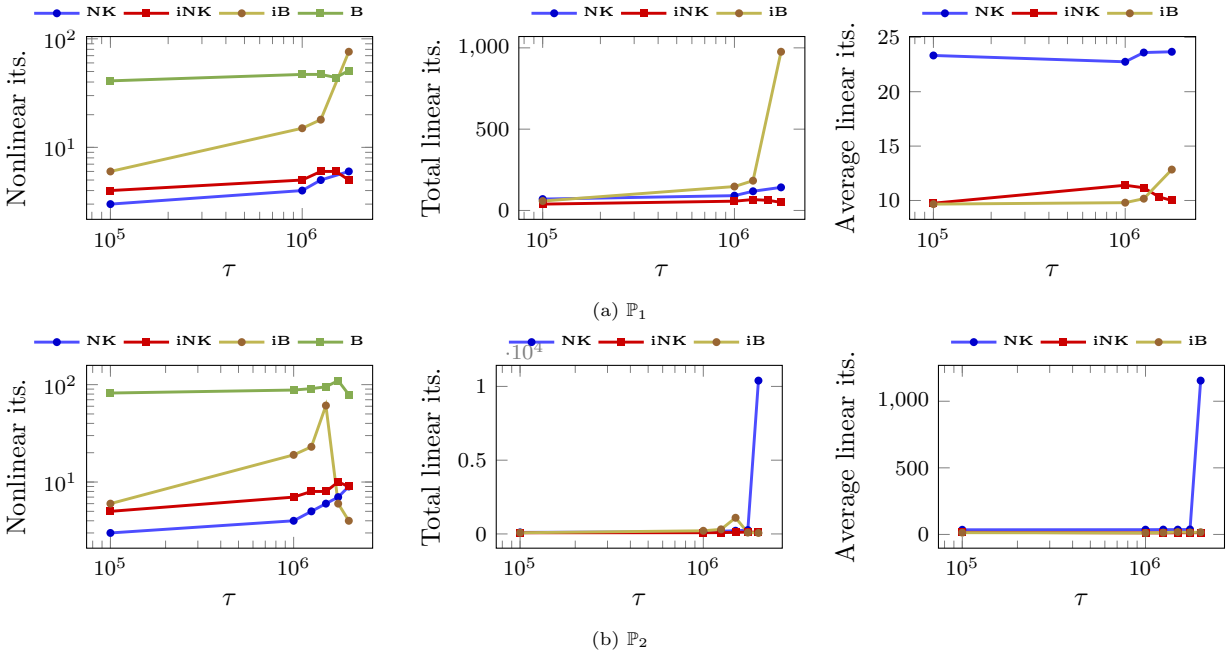

(a) $\mathbb{P}_1$ (b) $\mathbb{P}_2$

Figure 3: Sensitivity with respect to problem size, Heartbeat test. Time evolution of nonlinear iterations (nit), average linear iterations per nonlinear iteration (lit) and CPU time in seconds for $\mathbb{P}_1$ and $\mathbb{P}_2$ discretizations in mesh 4 (see Table 1). The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.

results for them. This holds not only for both methods presented (BFGS, inexact-BFGS), but also for the BFGS-exact variant described in Section 3, which was ineffective as well. Both Newton-Krylov and inexact Newton-Krylov methods present a very mild increase in the nonlinear iterations, but the inexact variant additionally presents only a small increase of average linear iterations. Because of this, the inexact method yields a faster overall performance. This advantage increases as the problem size increases, because the standard Newton-Krylov requires an increasing amount of linear iterations to converge. The ratio between the CPU times of the inexact Newton-Krylov method and Newton-Krylov method in the case with the largest amount of DoFs is 0.65.

**Heartbeat test.** We report the nonlinear iterations, average linear iterations per nonlinear iteration, and CPU time for first and second order finite elements in Table 4. To obtain a unique indicator, we consider the average of these quantities during the first 10 timesteps, and further plot their time evolution in Figure 3. We note that all methods are robust with respect to the degrees of freedom with first order elements, but instead with second order elements both Newton-Krylov and inexact-BFGS present a case in which they do not converge. Both Newton methods present similar nonlinear iteration counts, and decreasing average linear iterations are observed when going from Newton-Krylov to inexact-Newton and then to inexact-BFGS. In general, better performance is obtained with BFGS methods, where BFGS-preonly shows superior robustness. With first order elements, BFGS-inexact presents the best overall performance, and interestingly for second order instead the best performance is obtained with BFGS-preonly. The ratio between the CPU times of the best method (inexact-BFGS

in $\mathbb{P}_1$ , BFGS-preonly in $\mathbb{P}_2$ ) and the Newton-Krylov method is 0.66 for $\mathbb{P}_1$ and 0.77 for $\mathbb{P}_2$ elements. Interestingly, the superiority of BFGS-preonly in the second order case lasts only a few timesteps, as can be seen in the solution times in Figure 3. Most of the time, the best performance is still obtained using the inexact BFGS method. This shows that it is possible that the best method during a heartbeat depends on the phase of the PV loop, which will be one of our main interests in future studies.




(a) $\mathbb{P}_1$

(b) $\mathbb{P}_2$

Figure 4: Robustness with respect to the data, Cook test. Comparison of the nonlinear, total linear and average linear iterations incurred for each given value $\tau$ of the surface load magnitude. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.

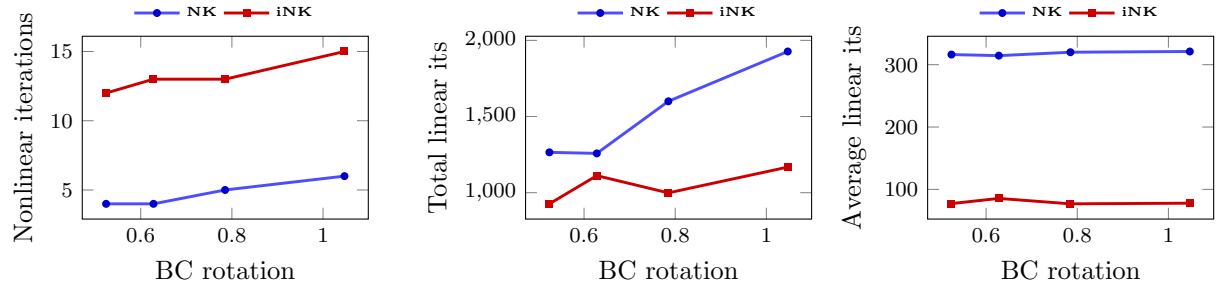
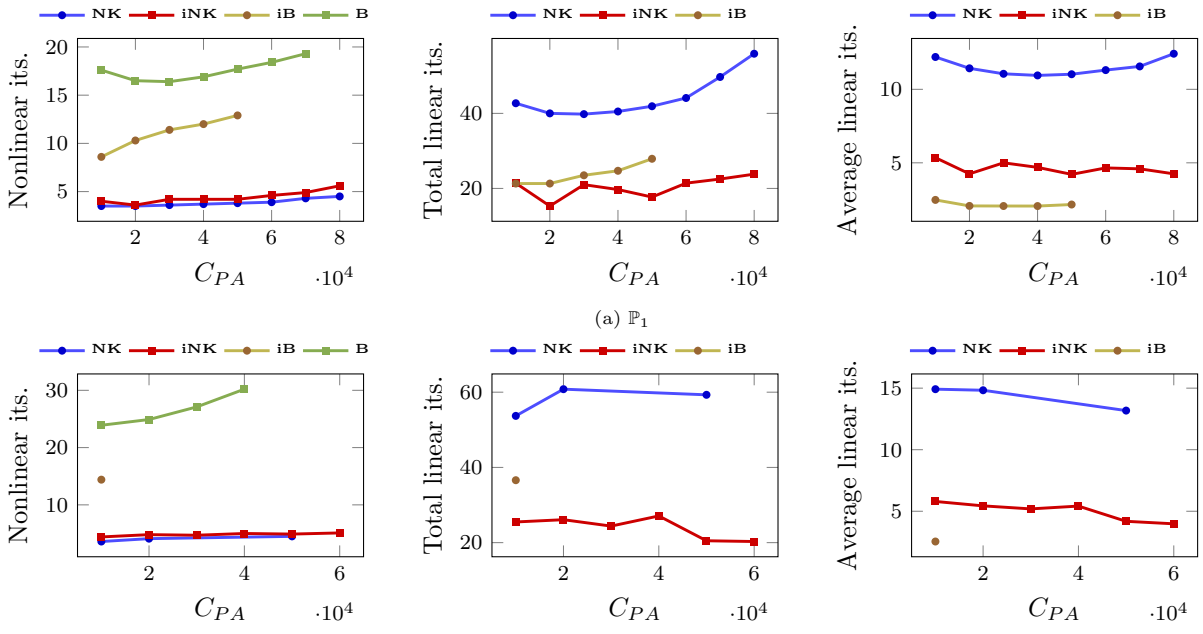


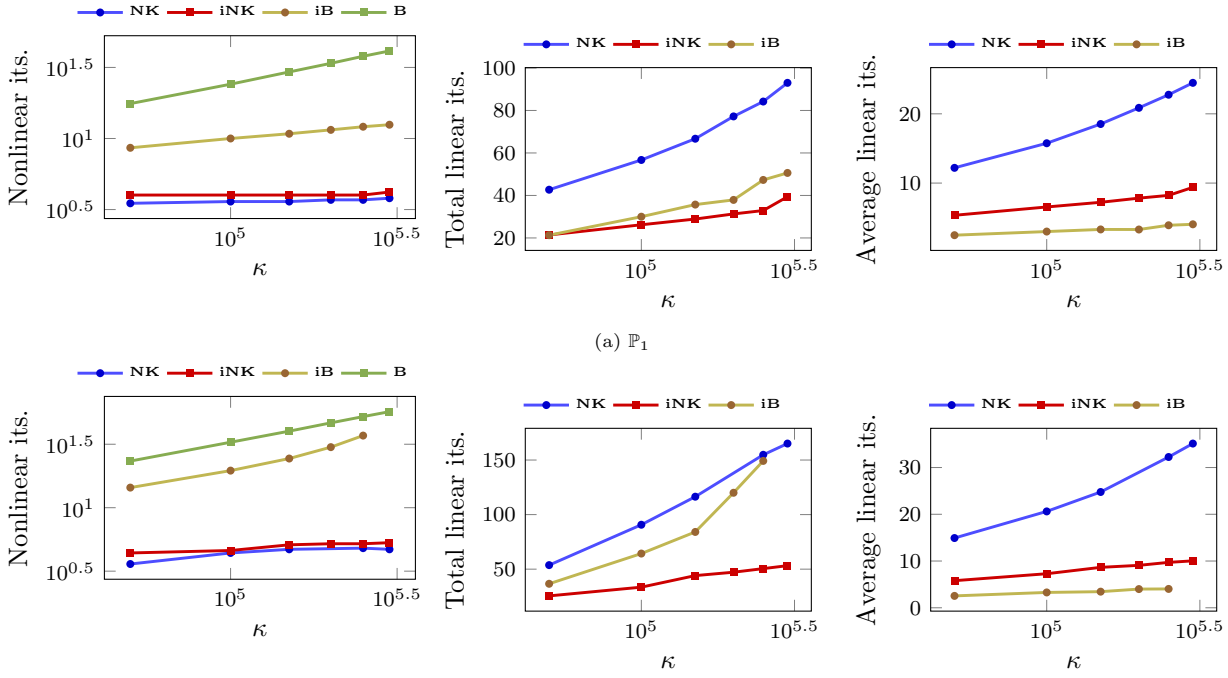

Figure 5: Robustness with respect to the data, Twist test. Nonlinear, total linear and average linear iterations with respect to the rotation angle in the boundary condition. The colors stand for ■ Newton-Krylov and ■ inexact Newton-Krylov.




(a) $\mathbb{P}_1$

(b) $\mathbb{P}_2$

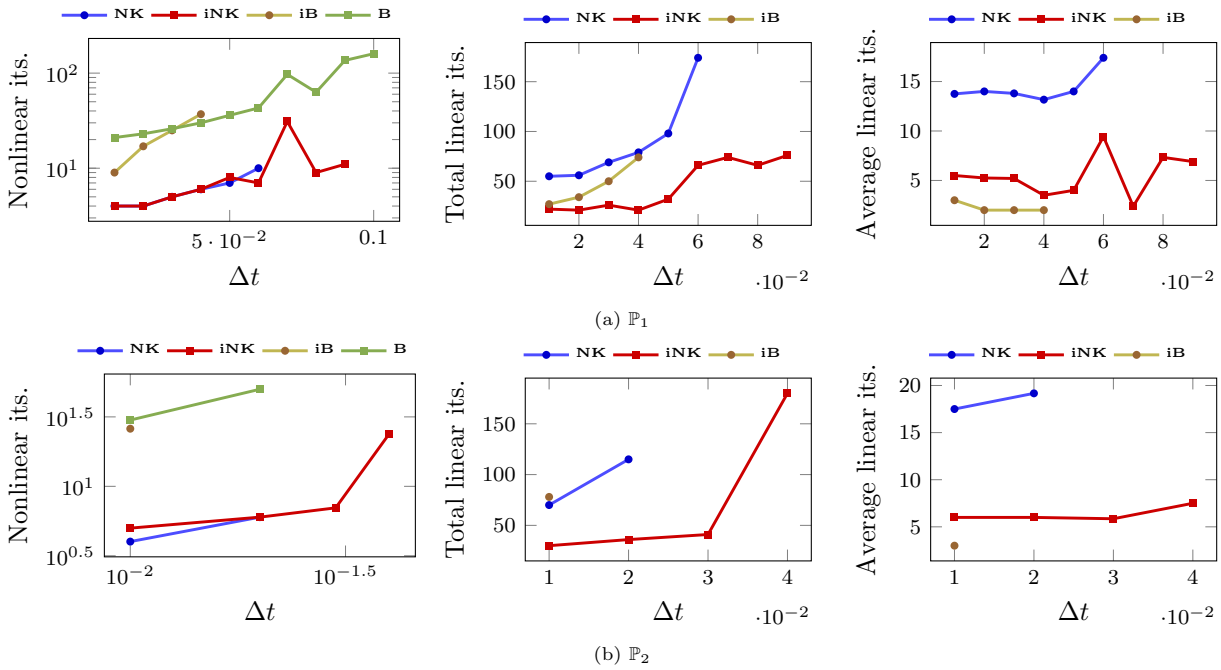
Figure 6: Robustness with respect to the peak activation, Heartbeat test. Comparison of the nonlinear, total linear and average linear iterations, averaged over the first 10 timesteps. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.




(a) $\mathbb{P}_1$

(b) $\mathbb{P}_2$

Figure 7: Robustness with respect to the bulk modulus, Heartbeat test. Comparison of the nonlinear, total linear and average linear iterations, averaged over the first 10 timesteps. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.




(a) $\mathbb{P}_1$

(b) $\mathbb{P}_2$

Figure 8: Robustness with respect to the time step, Heartbeat test. Comparison of the nonlinear, total linear and average linear iterations in the first time instant. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.

### *6.2. Robustness with respect to the data*

To study the robustness of the methods, we varied in each test case the parameter which resulted in a larger deformation and in the Cook test this refers to the vertical load $\tau$, in the Twist case this refers to the angle of rotation in the boundary conditions. In the Heartbeat case we considered three parameters: the peak activation constant $C_{\text{AP}}$, the bulk modulus $\kappa$, and the timestep $\Delta t$.

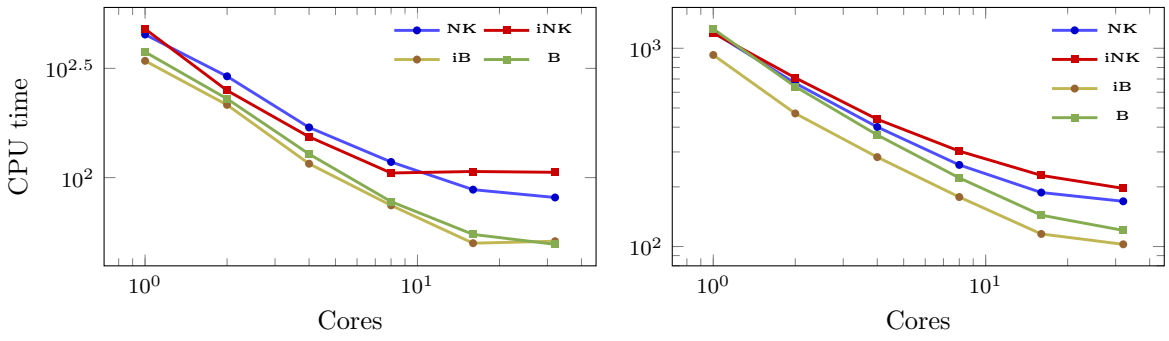
**Cook test.** We vary the magnitude of the distributed traction term $\tau$ on the right from $1.5 \cdot 10^6$ Pa to $2 \cdot 10^6$ Pa, where values over the upper bound considered make all methods diverge. The nonlinear iterations, total linear iterations and average linear iterations per nonlinear step are shown in Figure 4. We have computed the solution for six values of $\tau$, and plotted the result only when the solver converged. Most notably, all methods are capable of solving the problem, but only Newton-Krylov and BFGS-inexact present a clear deterioration when $\tau$ increases. We highlight that these are the same methods that presented a diverging scenario in Table 4, and in this case both fail in the second to last case. They additionally present an abrupt increase in the number of linear iterations required in the last values. When looking at the linear iteration counts, we see that the methods that converge tend to maintain a roughly constant number of average linear iterations, so that the increase in total linear iterations is given by the additional nonlinear iterations incurred. The only exception is the Newton-Krylov method for second order elements. Interestingly, inexact-BFGS is less robust than BFGS-preonly. There is no advantage in considering a Newton-Krylov method, but instead its inexact variant and the BFGS-preonly are much more robust with respect to $\tau$.

**Twist test.** We vary the angle in the boundary conditions from $\pi/6$ to $\pi/2$, and show the results in Figure 5. We note that both Newton methods converge for all values under consideration, and diverge when higher ones are considered, so there is no significant difference in terms of robustness between them. Still, the Newton-Krylov method presents a stronger increase in the total linear iterations, whereas the inexact variant shows a more robust iteration count.

**Heartbeat activation peak test.** We vary the peak activation $C_{PA}$ from $10^4$ to $10^5$, and plot only the results when the methods converged, see Figure 6. We considered 10 equidistant values for this test. In this case, we note that Newton methods provide an improved robustness for both first and second order finite elements, and as in the other tests, the inexact variant is more robust. The opposite effect is observed in the BFGS methods, where the BFGS-preonly shows an improved robustness with respect to the peak activation. We again observe that the total linear iterations of the Newton-Krylov method increase with the problem difficulty, whereas the inexact Newton-Krylov method maintains a roughly constant number of linear iterations despite the increase of the problem difficulty. Note that the problem with second order elements is much more difficult, as the values for which the methods converge are much less (up to $6 \cdot 10^4$ instead of $8 \cdot 10^4$ in the best case). In addition, in this case the inexact-Newton method is more robust than the Newton-Krylov method.

**Heartbeat bulk modulus test.** We vary the bulk modulus $\kappa$ from $5 \cdot 10^4$ to $3 \cdot 10^5$, and plot only the results when the methods converged. We considered 6 equidistant values for this test. In this case, we note that all methods are fairly robust with respect to the bulk modulus. Newton-Krylov presents the largest increase in linear iterations, both total and average. Instead, both inexact methods present a robust number of average linear iterations, so that the increase in total linear iterations is mainly driven by the increase of nonlinear iterations. BFGS-preonly presents a monotonic increase of the nonlinear iterations. Results are shown in Table 7.

**Heartbeat time step test.** In this case we considered ten equidistant timesteps ranging from 0.01 to 0.1, the results are displayed in Figure 8. We note that the results are similar to the ones obtained from the activation peak test. The second order formulation is much more difficult, and indeed the BFGS-inexact method converges only for the first timestep. Surprisingly, in this case the BFGS-preonly is the most robust method for first order elements, and is equivalent to Newton-Krylov for second order, where inexact-Newton is the most robust method. In this test, all methods deteriorate as the timestep grows, but Newton-Krylov is the only one where the average linear iterations grow monotonically before diverging. Yet again, this test confirms that BFGS-preonly is more robust than BFGS-inexact, and that inexact-Newton is more robust than Newton-Krylov.




(a) $\mathbb{P}_1$ (b) $\mathbb{P}_2$

Figure 9: Scalability, Cook test. CPU times with respect the number of cores. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.

| $\mathbb{P}_1$ discretization, DoFs = 1576875 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| cores | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 1 | 3 | 23.6 | 451.8 | 5 | 12.0 | 478.9 | 6 | 10.3 | 342.3 | 43 | 375.9 |
| 2 | 3 | 25.3 | 290.8 | 4 | 9.7 | 250.5 | 6 | 10.5 | 215.3 | 44 | 229.5 |
| 4 | 3 | 25.3 | 169.7 | 4 | 10.8 | 153.8 | 6 | 9.8 | 115.8 | 45 | 128.2 |
| 8 | 3 | 26.3 | 117.9 | 4 | 10.3 | 105.0 | 6 | 10.7 | 74.6 | 48 | 77.7 |
| 16 | 3 | 27.3 | 88.1 | 5 | 13.6 | 106.7 | 6 | 10.0 | 50.1 | 49 | 55.0 |
| 32 | 3 | 27.0 | 81.1 | 5 | 13.4 | 105.8 | 6 | 11.5 | 51.2 | 49 | 49.5 |

| $\mathbb{P}_2$ discretization, DoFs = 1576875 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| cores | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 1 | 3 | 34.3 | 1203.7 | 5 | 15.8 | 1195.1 | 6 | 13.5 | 925.3 | 81 | 1251.3 |
| 2 | 3 | 34.3 | 666.8 | 5 | 16.0 | 709.5 | 6 | 13.5 | 469.1 | 82 | 641.2 |
| 4 | 3 | 35.4 | 401.0 | 5 | 16.4 | 438.5 | 6 | 13.7 | 282.5 | 82 | 365.4 |
| 8 | 3 | 35.0 | 258.0 | 5 | 16.6 | 303.0 | 6 | 14.3 | 177.7 | 85 | 221.9 |
| 16 | 3 | 36.3 | 187.3 | 5 | 16.8 | 228.5 | 6 | 13.7 | 115.7 | 85 | 143.9 |
| 32 | 3 | 36.7 | 169.1 | 5 | 17.2 | 196.6 | 6 | 14.2 | 102.6 | 95 | 120.6 |

Table 5: Scalability, Cook test. nit $\coloneqq$ nonlinear iterations, lit $\coloneqq$ average linear iterations per nonlinear iteration, $T_{sol} \coloneqq$ CPU time in seconds.

| DoFs = 519747 | | | | | | |
|---|---|---|---|---|---|---|
| cores | **NK** | | | **iNK** | | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ |
| 1 | 6 | 800.0 | $3.68 \cdot 10^4$ | 16 | 240.4 | $2.61 \cdot 10^4$ |
| 2 | 6 | 806.2 | $2.13 \cdot 10^4$ | 16 | 225.3 | $1.62 \cdot 10^4$ |
| 4 | 6 | 806.0 | $1.37 \cdot 10^4$ | 16 | 222.4 | $1.06 \cdot 10^4$ |
| 8 | 6 | 826.3 | $8.02 \cdot 10^3$ | 16 | 237.3 | $6.56 \cdot 10^3$ |
| 16 | 6 | 835.5 | $4.62 \cdot 10^3$ | 16 | 240.3 | $3.94 \cdot 10^3$ |
| 32 | 6 | 828.5 | $3.44 \cdot 10^3$ | 16 | 244.8 | $3.10 \cdot 10^3$ |

Table 6: Scalability, Twist test. nit $\coloneqq$ nonlinear iterations, lit $\coloneqq$ average linear iterations per nonlinear iteration, $T_{sol} \coloneqq$ CPU time in seconds.

| $\mathbb{P}_1$ discretization, DoFs = 149511 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| cores | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 1 | 4.5 | 9.6 | 111.0 | 5.0 | 6.0 | 113.6 | 10.9 | 2.4 | 69.4 | 23.2 | 87.7 |
| 2 | 4.5 | 9.6 | 60.9 | 5.0 | 6.0 | 62.3 | 11.0 | 2.5 | 30.1 | 23.2 | 45.5 |
| 4 | 4.5 | 9.6 | 26.5 | 5.0 | 6.0 | 33.2 | 10.8 | 2.7 | 18.8 | 23.3 | 23.3 |
| 8 | 4.5 | 9.8 | 18.2 | 5.0 | 6.2 | 18.9 | 11.0 | 2.6 | 10.3 | 23.9 | 12.6 |
| 16 | 4.5 | 9.7 | 11.1 | 5.0 | 6.1 | 11.5 | 10.8 | 2.6 | 4.8 | 23.5 | 6.8 |
| 32 | 4.5 | 9.7 | 7.6 | 5.0 | 6.2 | 7.9 | 10.9 | 2.8 | 3.1 | 23.6 | 4.0 |
| $\mathbb{P}_2$ discretization, DoFs = 1123515 | | | | | | | | | | | |
| cores | **NK** | | | **iNK** | | | **iB** | | | **B** | |
| | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | lit | $T_{sol}$ | nit | $T_{sol}$ |
| 1 | 4.6 | 13.4 | 1061.0 | 5.3 | 6.5 | 965.4 | 14.1 | 3.5 | 637.4 | 30.2 | 515.1 |
| 2 | 4.6 | 13.5 | 512.8 | 5.3 | 6.4 | 569.0 | 13.4 | 3.6 | 287.4 | 30.4 | 287.7 |
| 4 | 4.6 | 13.5 | 324.5 | 5.0 | 6.5 | 294.3 | 15.3 | 3.4 | 184.9 | 30.5 | 149.8 |
| 8 | 4.6 | 13.4 | 184.1 | 5.3 | 6.6 | 176.0 | 14.5 | 3.6 | 100.7 | 30.1 | 81.7 |
| 16 | 4.6 | 13.4 | 113.2 | 5.2 | 6.6 | 107.8 | 14.5 | 3.3 | 59.1 | 30.5 | 40.5 |
| 32 | 4.6 | 13.4 | 82.0 | 5.1 | 6.5 | 74.5 | 14.2 | 3.5 | 43.3 | 30.3 | 33.6 |

Table 7: Scalability, Heartbeat test. nit := nonlinear iterations, lit := average linear iterations per nonlinear iteration, $T_{sol}$ := CPU time in seconds.

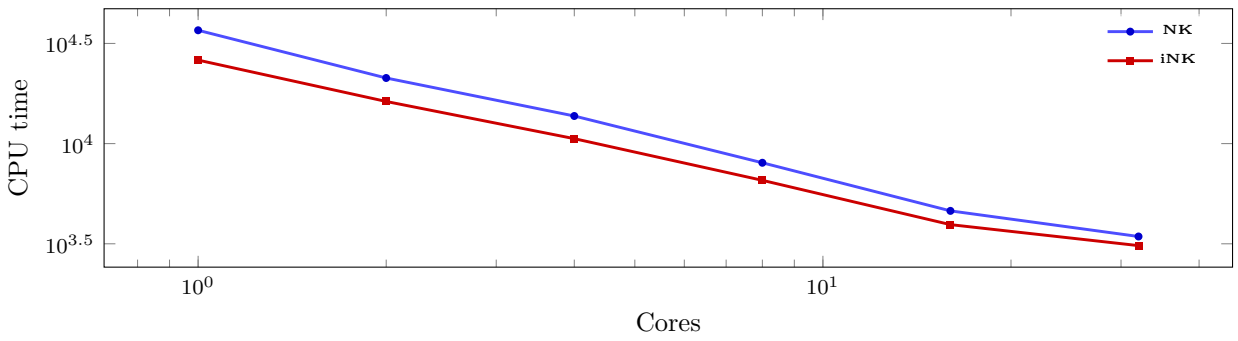
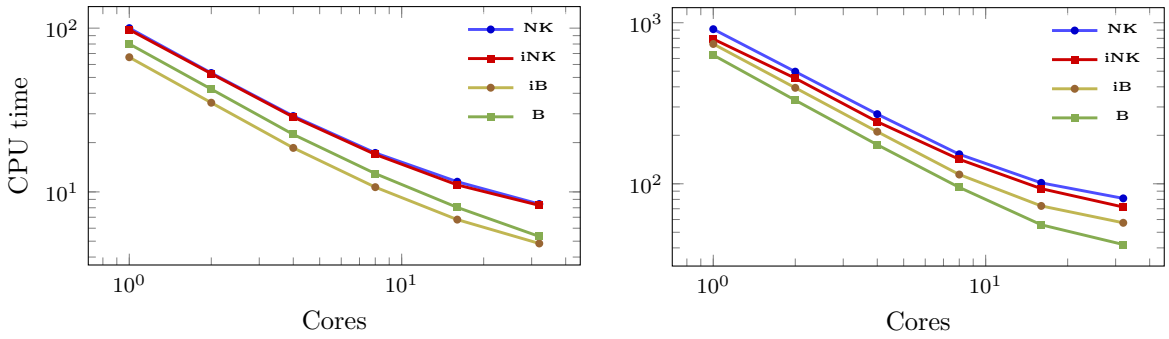



Figure 10: Scalability, Twist test. CPU time with respect to the number of cores. The colors stand for ■ Newton-Krylov and ■ inexact Newton-Krylov.




(a) $\mathbb{P}_1$ (b) $\mathbb{P}_2$

Figure 11: Scalability, Heartbeat test. CPU times with respect the number of cores. The colors stand for ■ Newton-Krylov, ■ inexact Newton-Krylov, ■ inexact-BFGS and ■ BFGS.

### *6.3. Scalability*

We test the strong scalability with respect to the finest mesh used in the performance tests. In the heartbeat test, we consider only the first 10 timesteps.

**Cook test.** We report the nonlinear and average linear iterations and the CPU times in Table 5. The CPU times are also displayed in Figure 9. We note that all methods are robust with respect to the number of cores, with only the exception of the inexact-Newton, which stagnates at 8 processors ($\approx$ 200k DoFs per processor). It is also interesting to note that when increasing the number of cores, BFGS-preonly becomes comparable to BFGS-inexact, as it becomes slightly better when considering 32 cores ($\approx$ 50k DoFs per process). This happens despite the mild deterioration in its performance as the number of cores increases.

**Twist test.** We report the nonlinear and average linear iteration counts, together with the CPU times in Table 6. The CPU times are also displayed in Figure 10. We note that this test in particular is in complete agreement with the previous ones, meaning that both Newton-Krylov methods perform similarly and exhibit an overall robust behavior. Both methods present adequate strong scaling, with the inexact method showing better solution times.

**Heartbeat test.** For this test, we report the nonlinear and average linear iterations, and CPU times averaged for the first 10 timesteps in Table 7. The total CPU times as function of the number of cores are displayed in Figure 11. We can appreciate that all methods are robust with respect to the number of cores, as these yield no significant variations. From the scalability curves shown in Figure 11, we note that all methods show adequate scaling, with the BFGS methods presenting an overall better performance. More specifically, inexact-BFGS is faster for $\mathbb{P}_1$ , and BFGS-preonly is faster for $\mathbb{P}_2$ . We highlight that in the parallel case, the speed-up yielded by BFGS methods is much better than the one obtained in serial, where the fraction of time incurred by the best method over the Newton-Krylov time in this case is 0.4 for both formulations.

### *6.4. Load distribution*

In this section we show the time distribution of the most important tasks throughout the solution process for each method. For all of them we consider Cook's problem with roughly 325000 degrees of freedom, first order finite elements, and a loading term given by $\tau = 10^5$. These times have been extracted from the PETSc logging routines, and have been summarized in Table 8, in which the main features of each method can be observed. For example, the time required for the evaluation of the residual increases according to the increase in the number of nonlinear iterations. Instead, the inexact Newton-Krylov methods spends more time assembling the Jacobian, and the BFGS methods both assemble it only once. The same phenomenon

can be observed in the preconditioner setup times. The time for solving the linear system required by the inexact Newton-Krylov is lower than the Newton-Krylov time, as expected for such methods. In addition, the inexact BFGS spends more time than the BFGS-preonly solving its linear system.

The weakness of each method can also be appreciated in this table: the Newton-Krylov method spends most of its solution time in the solution of the linear system, and it is also slowed down by the reassembly of the Jacobian and the set up of the preconditioner. In this regard, the inexact Newton-Krylov achieves a better balancing between the solution and the assembly phases, at the cost of a higher set up time. The BFGS-preonly method instead spends less time solving the linear system, but it requires many more residual assemblies, which is balanced by the inexact-BFGS, whose bottleneck is instead the solution of the linear system.

| | Newton-Krylov | Inexact Newton-Krylov | Inexact-BFGS | BFGS |
|---|---|---|---|---|
| Residual evaluation | 0.58 (2.21%) | 0.75 (3.11%) | 1.07 (5.36%) | 6.79 (33.90%) |
| Jacobian evaluation | 2.33 (8.89%) | 3.19 (13.24%) | 0.71 (3.56%) | 0.70 (3.49%) |
| Preconditioner setup | 7.86 (30.00%) | 10.36 (43.01%) | 2.65 (13.29%) | 2.63 (13.13%) |
| Linear system solution | 15.43 (58.89%) | 9.79 (40.64%) | 15.50 (77.73%) | 9.62 (48.03%) |
| Total solution time | 26.20 (100%) | 24.09 (100%) | 19.94 (100%) | 20.03 (100%) |

Table 8: Load distribution among each main task in each of the methods under consideration. The tests were performed in Cook's problem with roughly 325000 degrees of freedom using first order elements and a load given by $\tau = 10^5$. Times are displayed in seconds, with the percentage over the total solution time in parenthesis.

## 7. Conclusions

This work presents a detailed numerical study of some of the main numerical solvers used in large scale simulations of nonlinear mechanics, where all of the fundamental components of the methods are included: the nonlinear solver, the linear solver and the preconditioner. The main finding from this study is that superlinear solvers such as inexact Newton-Krylov and BFGS methods present an overall robustness similar to the gold standard represented by the Newton-Krylov method in nonlinear elasticity, but with improved solution times.

In difficult problems such as static and incompressible mechanics, the choice of the solver can be defined a-priori according to the convergence hypotheses required for each method, where in fact we have seen the BFGS methods fail for incompressible mechanics, as they usually require some form of convexity to guarantee convergence. Indeed, BFGS methods showed the best solution times in static mechanics, while inexact Newton-Krylov showed the best performance for the incompressible tests. The time-dependent heartbeat test showed us that the convexity contributed by the inertia term can further improve the performance of the BFGS method, which yielded much more robust iteration counts in this scenario than those shown in the static mechanics case. This comes as a surprise, given that the constitutive model used in the heartbeat test is much more complex than the models of the other tests. These nonlinear solvers present no drawbacks in terms of both performance and strong scalability, suggesting that in the context of nonlinear elasticity inexact Newton-Krylov and quasi-Newton methods should be preferred over the standard Newton-Krylov methods. In general, we observed the BFGS-preonly is more robust than BFGS-inexact, and inexact-Newton is more robust than Newton-Krylov.

We conclude by mentioning two possible extensions of this work. The first one is the inclusion in this study of Jacobian lagging techniques [BB13], which consist in avoiding the reassembly of the Jacobian for some Newton iterations. These lagging techniques could potentially further improve the performance of the Newton methods considered. Instead, inspired by our inexactness taxonomy, we note that a Jacobian reassembly could be considered for the BFGS methods, where the Jacobian gets sometimes reassembled to improve the quality of the initial Jacobian. Both strategies are non trivial to implement, and require a detailed problem-specific study to obtain the best performance. The second interesting extension is the

inclusion in our nonlinear mechanics study of a polyconvex potential [Bal76], even if solvers for this specific type of potential are not yet available, to the best of our knowledge. One interesting work in this direction is [BGO15], where a Hu-Washizu [Was68] formulation is used to exploit the convexity of the auxiliary function arising from the polyconvex potential. Still, this formulation results in a problem with many auxiliary variables, and its efficient numerical approximation remains an actively studied research topic.

## Appendix A. PETSc options

The commands used to invoke each of the methods described in Section 3 is detailed in what follows.

- **Newton-Krylov** (**NK**)

```
-snes_type newtonls
-ksp_type  gmres  # minres should work as well
-pc_type   hypre
-ksp_atol  1e-14
-ksp_rtol  1e-6
-snes_atol  1e-10
-snes_rtol  1e-6
-snes_stol  0.0
-snes_linesearch_type basic
```

Listing 1: PETSc commands to use Newton-Krylov.

- **Inexact Newton-Krylov** (**iNK**)

```
-snes_type newtonls
-ksp_type  gmres
-pc_type   hypre
-ksp_atol  1e-14
-snes_atol  1e-10
-snes_rtol  1e-6
-snes_stol  0.0
-snes_ksp_ew
-snes_ksp_ew_rtol0   1e-1
-snes_ksp_ew_rtolmax 0.1
-snes_linesearch_type basic
```

Listing 2: PETSc commands to use inexact Newton-Krylov.

- **Inexact-BFGS** (**iB**)

```
-snes_type qn
-ksp_type  gmres
-pc_type   hypre
-ksp_atol  1e-14
-ksp_rtol  1e-2
-snes_atol  1e-10
-snes_rtol  1e-6
-snes_stol  0.0
-snes_qn_type lbfgs
-snes_qn_m 20
-snes_qn_scale_type jacobian
-snes_linesearch_type basic
-snes_lag_jacobian 1000
-snes_lag_preconditioner 1000
-snes_qn_restart_type none
```

Listing 3: PETSc commands to use inexact-BFGS.

- **BFGS-preonly** (**B**)

```
-snes_type qn
-ksp_type  preonly
-pc_type   hypre
-snes_atol  1e-10
-snes_rtol  1e-6
-snes_stol  0.0
-snes_qn_type lbfgs
-snes_qn_m 20
-snes_qn_scale_type jacobian
-snes_linesearch_type basic
-snes_lag_jacobian 1000
-snes_lag_preconditioner 1000
-snes_qn_restart_type none
```

Listing 4: PETSc commands to use BFGS.

We note that caution must be taken when using quasi-Newton methods from PETSc. Indeed, we have observed erratic behavior of the solvers when using their default restart procedure, which we have ultimately turned off (with `snes_qn_restart_type none`). In addition, there is a default number of iterations after which the jacobian gets reassembled, which motivates the setting of `snes_lag_jacobian` and `snes_lag_preconditioner` by hand. These options are fundamental to obtain competitive performance from the proposed methods.

## Appendix B. Schur complement preconditioners

A Schur complement preconditioner is one that arises from a block LU factorization according to two (arbitrary) index sets of a matrix. If we consider a block matrix $\mathbf{M}$ given by the general structure

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B}_1 \\ \mathbf{B}_2 & \mathbf{C} \end{bmatrix},$$

with $\mathbf{A}$ invertible, a Gaussian elimination procedure yields

$$\mathbf{M} = \begin{bmatrix} I & \mathbf{0} \\ \mathbf{B}_2\mathbf{A}^{-1} & \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} - \mathbf{B}_2\mathbf{A}^{-1}\mathbf{B}_1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{B}_1 \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{B.1}$$

We note that if $\mathbf{C}$ is invertible, the same procedure can be applied with respect to it. Schur complement based preconditioners enjoy excellent theoretical properties, as the preconditioned system possesses at most 3 distinct eigenvalues [MGW00], implying that it converges in at most 3 iterations of a Krylov subspace method. This is true whenever the Schur complement $\mathbf{S} = \mathbf{C} - \mathbf{B}_2\mathbf{A}^{-1}\mathbf{B}_1$ is evaluated exactly, which is usually computationally intractable. One simple approximation, which is the one we use for the preconditioning the incompressible mechanics problem, is to consider the approximation $\mathbf{A}^{-1} \approx \text{diag}^{-1}(\mathbf{A})$. Another important point is that in general using all three blocks arising from the factorization in (B.1) is not necessary, and instead it is sufficient to consider a lower factorization (first two blocks) or an upper factorization (last two blocks). Whenever $\mathbf{C} = \mathbf{0}$, it is also possible to use a diagonal factorization (middle block only).


## Acknowledgments

N. Barnafi and L. F. Pavarino have been supported by grants of MIUR (PRIN 2017AXL54F_002) and INdAM–GNCS. N. Barnafi and S. Scacchi have been supported by grants of MIUR (PRIN 2017AXL54F_003) and INdAM-GNCS. The Authors are also grateful to the University of Pavia, the University of Milan, and the CINECA laboratory for the usage of the EOS, INDACO and Galileo100 clusters, respectively. We also acknowledge the comments of the anonymous reviewers, which substantially improved the quality of this work.

## References


[ABH+15] M Alnæs, J Blechta, J Hake, A Johansson, B Kehlet, A Logg, C Richardson, J Ring, ME Rognes, and GN Wells. The FEniCS project version 1.5. *Archive of Numerical Software*, 3(100), 2015.

[ADVL+13] F Auricchio, LB Da Veiga, C Lovadina, A Reali, RL Taylor, and P Wriggers. Approximation of incompressible large deformation elastic problems: some unresolved issues. *Computational Mechanics*, 52(5):1153–1167, 2013.

[BAA+21] S Balay, S Abhyankar, MF Adams, J Brown, P Brune, K Buschelman, L Dalcin, A Dener, V Eijkhout, W Gropp, D Karpeyev, D Kaushik, M Knepley, D May, L Curfman McInnes, R Mills, T Munson, K Rupp, P Sanan, B Smith, S Zampini, H Zhang, and H Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.13, Argonne National Laboratory, 2021.

[Bal76] JM Ball. Convexity conditions and existence theorems in nonlinear elasticity. *Archive for rational mechanics and Analysis*, 63(4):337–403, 1976.

[BB13] J Brown and P Brune. Low-rank quasi-Newton updates for robust jacobian lagging in Newton methods. In *Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, volume 201, pages 1–2, 2013.

[BBPT12] JD Bayer, RC Blake, G Plank, and NA Trayanova. A novel rule-based algorithm for assigning myocardial fiber orientation to computational heart models. *Annals of Biomedical Engineering*, 40(10):2243–2254, 2012.

[BGO15] J Bonet, AJ Gil, and R Ortigosa. A computational framework for polyconvex large strain elasticity. *Computer Methods in Applied Mechanics and Engineering*, 283:1061–1094, 2015.

[BKM+12] J Brown, MG Knepley, DA May, LC McInnes, and B Smith. Composable linear solvers for multiphysics. In *2012 11th International Symposium on Parallel and Distributed Computing*, pages 55–62. IEEE, 2012.

[BRKN17] M Borregales, FA Radu, K Kumar, and JM Nordbotten. Robust iterative schemes for non-linear poromechanics. *arXiv preprint arXiv:1702.00328*, 2017.

[CD04] C Carstensen and G Dolzmann. An a priori error estimate for finite element discretizations in nonlinear elasticity for polyconvex materials under small loads. *Numerische Mathematik*, 97(1):67–80, 2004.

[CDSS18] JO Campos, RW Dos Santos, and BM Sundnes, Jand Rocha. Preconditioned augmented Lagrangian formulation for nearly incompressible cardiac mechanics. *International journal for numerical methods in biomedical engineering*, 34(4):e2948, 2018.

[CFPS15] P Colli Franzone, LF Pavarino, and S Scacchi. Parallel multilevel solvers for the cardiac electromechanical coupling. *Appl. Numer. Math.*, 95:140–153, 2015.

[CGK+98] X-C Cai, WD Gropp, DE Keyes, RG Melvin, and DP Young. Parallel Newton–Krylov–Schwarz algorithms for the transonic full potential equation. *SIAM J. Sci. Comput.*, 19(1):246–265, 1998.

[CGKT94] X-C Cai, WD Gropp, DE Keyes, and MD Tidriri. Newton-Krylov-Schwarz methods in CFD. In *Numerical methods for the Navier-Stokes equations*, pages 17–30. 1994.

[Cia21] PG Ciarlet. *Mathematical elasticity: Three-dimensional elasticity*. SIAM, 2021.

[DES82] RS Dembo, SC Eisenstat, and T Steihaug. Inexact Newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982.

[DS96] JE Dennis and RB Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM, 1996.

[EG13] A Ern and JL Guermond. *Theory and practice of finite elements*, volume 159. Springer Science & Business Media, 2013.

[EHS+06] H Elman, VE Howle, J Shadid, R Shuttleworth, and R Tuminaro. Block preconditioners based on approximate commutators. *SIAM Journal on Scientific Computing*, 27(5):1651–1668, 2006.

[EMFTF10] A El Maliki, M Fortin, N Tardieu, and A Fortin. Iterative solvers for 3d linear and nonlinear elasticity problems: Displacement and mixed formulations. *International journal for numerical methods in engineering*, 83(13):1780–1802, 2010.

[EW94] SC Eisenstat and HF Walker. Globally convergent inexact Newton methods. *SIAM Journal on Optimization*, 4(2):393–422, 1994.

[EW96] SC Eisenstat and HF Walker. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[FBF15] PE Farrell, A Birkisson, and SW Funke. Deflation techniques for finding distinct solutions of nonlinear partial differential equations. *SIAM Journal on Scientific Computing*, 37(4):A2026–A2045, 2015.

[FY02] RD Falgout and UM Yang. hypre: A library of high performance preconditioners. In *International Conference on Computational Science*, pages 632–641. Springer, 2002.

[GCM14] A Gimenez, V Chausse, and A Meseguer. Numerical continuation in classical mechanics and thermodynamics. *European journal of physics*, 36(1):015015, 2014.

[GMW91] JM Guccione, AD McCulloch, and LK Waldman. Passive material properties of intact ventricular myocardium determined from a cylindrical model. *Journal of biomechanical engineering*, 113(1):42–55, 1991.

[GP88] JC Gelin and P Picart. Use of quasi-Newton methods for large strain elastic-plastic finite element computations. *Communications in applied numerical methods*, 4(4):457–469, 1988.

[GVL96] GH Golub and CF Van Loan. Matrix computations, 1996.

[Hac13] W Hackbusch. *Multi-grid methods and applications*, volume 4. Springer Science & Business Media, 2013.

[Hol02] GA Holzapfel. Nonlinear solid mechanics: A continuum approach for engineering science. *Meccanica*, 37(4):489–490, 2002.

[Hug12] TJR Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[JHN11] N Jansson, J Hoffman, and M Nazarov. Adaptive simulation of turbulent flow past a full car model. In *SC'11: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–8. IEEE, 2011.

[KGH+22] E Karabelas, MAF Gsell, G Haase, G Plank, and CM Augustin. An accurate, robust, and efficient finite element framework with applications to anisotropic, nearly and fully incompressible elasticity. *Computer Methods in Applied Mechanics and Engineering*, 394:114887, 2022.

[LBK17] T Liu, S Bouaziz, and L Kavan. Quasi-Newton methods for real-time simulation of hyperelastic materials. *ACM Transactions on Graphics (TOG)*, 36(3):1–16, 2017.

[LLS05] S Linge, G Lines, and J Sundnes. Solving the heart mechanics equations with Newton and quasi Newton methods–a comparison. *Computer methods in biomechanics and biomedical engineering*, 8(1):31–38, 2005.

[LNLS14] S Land, SA Niederer, P Lamata, and NP Smith. Improving the stability of cardiac mechanical simulations. *IEEE Transactions on Biomedical Engineering*, 62(3):939–947, 2014.

[LRCC15] NM Lafontaine, R Rossi, M Cervera, and M Chiumenti. Explicit mixed strain-displacement finite element for dynamic geometrically non-linear solid mechanics. *Computational Mechanics*, 55(3):543–559, 2015.

[Man90] J Mandel. On block diagonal and schur complement preconditioning. *Numerische Mathematik*, 58(1):79–93, 1990.

[MGW00] MF Murphy, GH Golub, and AJ Wathen. A note on preconditioning for indefinite linear systems. *SIAM Journal on Scientific Computing*, 21(6):1969–1972, 2000.

[Noc80] J Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[PM16] J-P Pelteret and A McBride. The deal.ii code gallery: Quasi-static finite-strain compressible elasticity, 2016.

[PSZ15] LF Pavarino, S Scacchi, and S Zampini. Newton–Krylov-BDDC solvers for nonlinear cardiac mechanics. *Computer Methods in Applied Mechanics and Engineering*, 295:562–580, 2015.

[QLRRB17] AM Quarteroni, T Lassila, S Rossi, and R Ruiz-Baier. Integrated Heart—Coupling multiscale and multiphysics models for the simulation of the cardiac function. *Computer Methods in Applied Mechanics and Engineering*, 314:345–407, 2017.

[RSA+20] F Regazzoni, M Salvador, PC Africa, M Fedele, L Dede, and AM Quarteroni. A cardiac electromechanics model coupled with a lumped parameters model for closed-loop blood circulation. part I: model derivation. *arXiv e-prints*, 2020.

[Rud16] S Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[Saa03] Y Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[SHOL06] T Smith, R Hooper, C Ober, and A Lorber. Intelligent nonlinear solvers for computational fluid dynamics. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, page 1483, 2006.

[ULM02] TP Usyk, IJ LeGrice, and AD McCulloch. Computational model of three-dimensional cardiac electromechanics. *Computing and Visualization in Science*, 4(4):249–257, Jul 2002.

[Was68] K Washizu. Variational methos in elasticity and plasticity. *International Series of Monographs in Aeronautics and Astronautics*, 1968.

[WDE07] M Weiser, P Deuflhard, and B Erdmann. Affine conjugate adaptive Newton methods for nonlinear elastomechanics. *Optimisation Methods and Software*, 22(3):413–431, 2007.

[WN99] S Wright and J Nocedal. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

[Wu06] SR Wu. Lumped mass matrix in explicit finite element method for transient dynamics of elasticity. *Computer Methods in Applied Mechanics and Engineering*, 195(44-47):5983–5994, 2006.

[ZTNZ77] OC Zienkiewicz, RL Taylor, P Nithiarasu, and JZ Zhu. *The finite element method*, volume 3. McGraw-hill London, 1977.