On the Numerical Modelization of Moving Load Beam Problems by a Dedicated Parallel Computing FEM Implementation

Diego Froio, Luca Verzeroli, Rosalba Ferrari & Egidio Rizzi

Archives of Computational Methods in Engineering
State of the Art Reviews

ISSN 1134-3060
Volume 28
Number 4

Arch Computat Methods Eng (2021)
28:2253-2314
DOI 10.1007/s11831-020-09459-5

Springer

Archives of Computational Methods in Engineering (2021) 28:2253–2314
https://doi.org/10.1007/s11831-020-09459-5

ORIGINAL PAPER

On the Numerical Modelization of Moving Load Beam Problems by a Dedicated Parallel Computing FEM Implementation

Diego Froio[1] · Luca Verzeroli[1] · Rosalba Ferrari[1] · Egidio Rizzi[1]

Received: 24 December 2019 / Accepted: 18 June 2020 / Published online: 18 August 2020

Abstract
The present work outlines an original numerical modelization approach for Moving Load (ML) beam problems, by a dedicated object-oriented C++ parallel computing FEM implementation, with the purposes of performing efficient numerical analyses resolving the complete dynamic response of beams under the effect of a high-velocity ML. Alongside, main framing state-of-the-art reviews are attempted, on the principal involved issues of: ML context and physical description, numerical FEM modelization, parallel computing implementation. Running ML example cases are explored, for a (long) finite beam on a (visco)elastic foundation and for a continuous beam of a historic railway iron bridge, with per se interesting engineering outcomes. The contribution may serve as a guideline paradigm to readers that may be novel to the treated topics, though motivated in adventuring in the computational challenges involved in the present mechanical research context.

1 Introduction

1.1 Generalities and Contextualization

Many recurring practical dynamical engineering applications may be modeled as Moving Load (ML) problems, namely as those peculiar mechanical problems occurring when the point of application of concentrated or distributed loads acting on a certain structural element changes along time, at a constant or variable velocity. For example, in transportation engineering, the need of correctly predicting the dynamic response of structural systems under (growing) high-speed moving vehicles is becoming crucial, for guaranteeing vehicle stability, monitoring maintenance costs and avoiding possible passenger discomfort. Thereby, railways, bridges, cable ways, overhead cranes constitute selected characteristic contexts of application of ML problems (and other moving object problems, see e.g. Metrikine and Dieterman [110], Metrikine and Verichev [111], Mazilu [108, 109], Dimitrovová [35–37], and Rodrigues et al. [131], with therein quoted references, even, very recently, in the nonlinear range of mechanical response, Fărăgău et al. [60], Sanches et al. [139]).

For instance, among the most modern and pioneering projects, the TransPod company has recently proposed a next-generation mass tube-based vacuum transportation system consisting of a ultra-high-speed vehicle designed to carry passengers at a speed exceeding 1000 km/h, as declared by Janzen [88]. The tube infrastructure that would interconnect cities has been designed as a canted spiral-welded steel tube segment, supported by a substructure anchored to the foundation. In such a context, it becomes fundamental to appreciate the response of the whole supporting structure, when an ultra-high-speed vehicle is moving along. Although the TransPod project shall still look a bit futuristic, high-speed trains traveling at $300 \div 500$ km/h are almost becoming commonplace in everyday life, bringing to light the importance of the present ML dynamical context at a fast travelling velocity, and of achieving relevant consistent modelizations and attached efficient computational simulations.

It appears fundamental to investigate how the interaction between the structure and the ML influences the amplitude of the resulting structural vibration, in terms of different kinematic components, and in which manner the latter may depend on the source ML characteristics, and specifically on its propagating velocity. As it had been theoretically demonstrated by Timoshenko [148] and then by various other researchers, as discussed in detail in the following, an excessive dynamic amplification of structural vibrations induced

✉ Egidio Rizzi
egidio.rizzi@unibg.it

[1] Dipartimento di Ingegneria e Scienze Applicate, Università degli studi di Bergamo, Viale G. Marconi 5, 24044 Dalmine (BG), Italy

by MLs may become apparent, when the ML velocity attains a certain characteristic value, named "critical velocity" [38, 39].

Hence, demands for safe infrastructures and cost-effective design solutions are becoming necessary, in such an increasingly challenging ML context. A successful design shall strongly depend on the availability of robust and efficient computational tools, as conceived to achieve accurate, reliable and accessible numerical simulations of the dynamic response induced by MLs. By the analysis of the outcomes of the numerical simulations, potential practical implications in contemporary transportation engineering, especially in terms of lowering down the admissible ranges of high-speed vehicle velocities, shall be revealed, and possible remediation measures for the mitigation of the induced system vibrations may be explored.

1.2 General Framing

The determination of such a numerical dynamic response under ML constitutes a rather challenging task, since it shall involve a quite sophisticated modelization and a rather intensive computation. Toward that, the Finite Element Method (FEM) shall represent a wide-spread and powerful numerical method by which an accurate description of the physical structural response may numerically be obtained.

The multi-fold history of the FEM began long time ago, likely back to the '50s, and shall be hard to be reported in complete form, especially in general terms (thus, exceeding the scope of describing modelization approaches specifically devoted to ML problems). However, four main researchers may here be mentioned, for framing, referring to the first origins of the FEM in modern terms. Firstly, Argyris [8] demonstrated the efficacy of his numerical techniques, by providing a constant matrix formulation for the principles of virtual displacements and forces. He managed to apply that method to a rectangular planar elasticity "element", even though, this term had not actually been coined yet. As reported in Gupta and Meek [73] and debated during the 5th World Congress on Computational Mechanics (WCCM V, 2002), professor Clough first baptized the method as follows (see Clough [27]):

> «On the basis that a deflection analysis done with these new 'pieces' (or elements), of the structure is equivalent to the formal integration procedure of integral calculus, I decided to call the procedure the FEM because it deals with finite components rather than differential slices.»

However, Turner et al. [153] was likely the first scientist who exported the FEM to everyday use, while he was working at the Boeing company, over period 1952–1964. Initially, Turner et al. [153] demonstrated the displacement convergence characteristics of planar elements, and then, they generalized and improved the so-called Direct Stiffness Method, in order to study aircraft structures. During WCCM V, Clough also reported an important circumstance that had given a positive turn to the history of the FEM (see Clough [27]):

> «A 'red letter' event that occurred during this very early history of FEM was my visiting Northwestern University [in 1958] to give a seminar lecture on finite elements. When I received this invitation from Olek Zienkiewicz, who was teaching at Northwestern at that time, I expected we would have some arguments about the relative merits of finite elements versus finite differences because Olek had been brought up in the tradition of Professor Southwell. It is true that we did have some such discussions, but Olek recognized very quickly the advantages of the finite element approach. In fact I would say that my visit to Northwestern yielded a tremendous dividend in the conversion of Olek from finite differences to finite elements.»

It may also be noted that Zienkiewicz likely was the first author who published a book popularizing the FEM, see Zienkiewicz and Taylor [165]. Thus, he should be considered as one of the main proponents of the consequent "technology transfer", namely the process of exporting FEM concepts from aerospace industry to a wider range of engineering applications. The FEM had then turned out to constitute a fundamental tool to acquire an in-depth knowledge on the behaviour of different physical and engineering systems. Specifically, the effectiveness of the method in describing the dynamic behaviour of mechanical systems and structural elements has been realized and considerably developed.

Up to then, the FEM had been treated without considering an essential ingredient that truly supports the efficiency of the method: *digital computation*. Computers have been considered as a scarce and precious resource, for several years. In fact, during the '50s, only large industrial companies managed to afford mainframe computers, because of the required dedicated and expensive infrastructures. It shall not be a coincidence that all of the mentioned FEM pioneers were likely working in the aerospace industry, at least during a part of their careers. As chronicled by Felippa [43], military companies first moved from desk calculators and punched-tape accounting machines to digital computers during the so-called "cold war". Another barrier to the diffusion of the FEM was probably represented by the secretiveness on the proprietary FEM codes developed by companies. In spite of that, the distribution of early open-source FEM software began, thanks to researchers and universities, but only around

the ’70s. Since then, the FEM approach has become more and more popular, within a wider amount of engineering applications.

The increasing complexity of FEM models has then required faster and faster electronic components, to deliver the demanded processing speed. As predicted by the Co-founder of the Intel Corporation, Moore [114], the number of components per integrated circuit has rapidly grown for years, inducing an increase in the available computational resources. Otherwise, this same positive trend may not last forever. Nowadays, it may be getting harder and harder to achieve further performance improvements at the same pace, because of present physical and technological limits induced by the design of micro- or nano-components. Despite that, the computational time may significantly be reduced by introducing different efficient coding strategies such as *parallelism*, also allowing to reach and take benefit of modern distributed resources of High Performance Computing (HPC).

1.3 Paper Scope

For the above briefly mentioned contextualization reasons, the main subject of the present work is constituted by setting a FEM structural dynamic analysis of a single slender finite Euler–Bernoulli elastic beam excited by a transverse concentrated ML, traveling at a high constant velocity along the beam. The beam is assumed to lie on either a distributed continuous (visco)elastic foundation (linear or nonlinear) or on firm discrete supports.

In particular, within the context of a small displacement (geometrically linear) theory, various numerical analyses are performed, focusing on the numerical determination of the physical response of the mechanical system, and specifically of the critical ML velocities, leading to large beam deflections, by revealing all the associated kinematic characteristics.

The scope of the paper is to investigate the physical response of such a beam-support structural system and to identify all its peculiar features, by first developing a literature framing on the several involved aspects of the relevant modelization approach and then producing a campaign of numerical simulations relying on a parallel computing paradigm. The latter philosophy is briefly introduced, as a first guideline to readers that may be involved in similar parallelization projects. Then, numerical results are obtained through an autonomous object-oriented C++ FEM implementation, specifically coded within a parallel programming environment, aiming at drastically reducing the computational cost of possible parametric analyses of the ML mechanical dynamical problem under consideration.

1.4 Work Summary

The basics of FEM and its applications to the field of ML dynamical problems are first presented, after a general discussion about competent analytical and numerical approaches developed in the last decades for studying the response of railway tracks or bridges under ML. Then, the FEM is employed to derive the characteristic matrices and vectors of a beam finite element, with or without modelling an underlying distributed foundation. Moreover, the HTT-α method is considered, as an efficient numerical integrator for determining the transient structural response, within both linear and nonlinear contexts.

FEM applications frequently require a fine, often multi-dimensional, discretization, resulting into several degrees of freedom and attached mechanical system unknowns. Thereby, the opportunity of exploring efficient parallel computing environments in FEM calculations has been investigated. First of all, main generalities and strategies of parallelization are analyzed, for a useful guideline. Then, an algorithm that could be employed in a generic FEM analysis is here developed and described, in order to examine the potential advantages of employing distributed computing systems. Subsequently, the algorithm is implemented within an object-oriented C++ FEM formulation, and its fundamental features are sketched, to start and to compare with a previous sequential MatLab FEM coding of the same ML modelization. Furthermore, some benchmark performance results are reported, to illustrate achieved *speed-up*, *efficiency* and *scalability* of the devised C++ parallel computing implementation.

At a first validation stage, the implemented parallel code is employed to carry out the analysis of the transient response of a simply supported finite beam lying on a (visco) elastic foundation subjected to a transverse concentrated ML, with either a constant or a harmonic-varying amplitude in time. A successful validation with previous recent outcomes by a standard FEM approach is demonstrated, with a much superior computational performance. Then, the dynamic response of a 1D multi-span FEM model of the upper continuous beam of a historic railway riveted iron arch bridge, the Paderno d’Adda Bridge (1889), is evaluated, by considering the structure subjected to a constant-magnitude and constant-velocity ML, referring to the passage of a locomotive along the upper continuous beam. Numerical tests allow to reveal the effects induced by the bending stiffness and the damping of the beam on the critical velocity of the ML, and to observe the whole dynamic response of the structure in space and time.

The present investigation shows, after an appropriate literature framing, that an original, efficient C++ parallel computing FEM implementation devoted to analyze ML problems allows to derive representative numerical

results, which truly reveal the physical characteristics of the dynamic response of the structural system and become apt to consistently quantify the amount of vibration, in view of assessing the structural performance of the mechanical system in such a challenging dynamical context.

The literature framing and the combination of a new object-oriented C++ parallel computing FEM implementation with the mechanical and engineering implications derived from the numerical tests on beam ML problems shall represent the main contributions of this work. Furthermore, the present approach shall reveal to be feasible not only for inspecting the dynamic response of those physical systems subjected to MLs but also for representing an efficient tool to carry out complex FEM analyses, by distributed computing, independently from the specific field of application, as here considered.

1.5 Manuscript Organization

The presentation in the manuscript is organized into seven main sections, as it is outlined below, as a structural guideline toward a convenient reading. State-of-the-art reviews will be presented along the next three sections, concerning the relative topics.

Section 2 presents a discussion about several approaches developed in the last decades for studying the response of railway tracks or bridges under the passage of a fast travelling vehicle. These kinds of analyses usually concern the response of *finite* beams under ML and the investigation of their resonance conditions. Several criteria for classifying research works in the field of ML problems may be selected, but, in the following exposition, a methodological classification is basically adopted, based on the solution strategy employed for solving various ML problems. Therefore, literature contributions are analyzed and presented by organizing them into two main categories: *analytical* and *numerical* approaches. Accordingly, the section is subdivided into two corresponding main parts. Both subsections focus onto two different kinds of finite beam models, namely beams without underlying foundation, constrained only at the extremes, and beams also supported by a (visco)elastic foundation, with either a linear or a nonlinear behavior.

Section 3 deals with the basics of the FEM and its specific applications to the field of ML dynamical problems of planar one-dimensional beams. After a brief introduction to the Weighted Residuals Method (WRM) and, in particular, to the Galerkin approach, the specific application of the FEM to ML structural problems is described, together with the derivation of the matrices and vectors of a beam finite element, with or without underlying foundation. Furthermore, the HTT-α method is presented, as an efficient numerical integrator scheme for determining the transient structural dynamic response, within both the linear and nonlinear contexts.

Section 4 examines the opportunities for exploiting parallelism in FEM calculations. Three different approaches to achieve a parallel implementation of the FEM are herein presented, together with the models of computation that shall allow to select the most suitable parallel computing architecture. In the end, an algorithm to be employed in a generic FEM analysis is described, in order to take advantage of the potentialities of distributed computing systems.

Section 5 reports how the previous algorithm could be devoted to the analysis of the considered ML beam problem (and related ones). First of all, the fundamental features of an implemented object-oriented C++ parallel FEM code are briefly sketched, in order to introduce the comparison of performances between a previous sequential MatLab FEM coding, earlier developed by Froio et al. [50, 51, 54, 56], and the current C++ parallel version of the program. Moreover, the numerical transient response of a simply supported beam lying on a (visco)elastic foundation and subjected to a ML, with either a constant or a harmonic-varying amplitude in time, is considered. In order to validate the devised C++ code, the obtained numerical outcomes in terms of critical velocities are depicted, and compared with recent reference results from the literature (Castro Jorge et al. [19], Froio et al. [56]). Finally, some benchmark performance results of the achieved parallel computing strategy are reported, to illustrate the achieved *speedup*, *efficiency* and *scalability* of the present object-oriented C++ parallel implementation.

Section 6 outlines the final adoption of the implemented algorithm to evaluate the dynamic response of a 1D multi-span FEM model of the upper continuous beam of a historical railway riveted iron arch bridge, namely the Paderno d'Adda Bridge (1889), which has recently been made the target of a wide research project concerning the determination of its present structural capacity (see Ferrari et al. [44–46]), in both static and dynamic mechanical contexts. The technique originally considered by designer engineer Jules Röthlisberger to conceive and calculate the bridge is briefly introduced and some technical aspects are summarized, to provide an introductory view about the architecture and structural functioning of such a monumental railway bridge. The upper box-beam of the Paderno d'Adda Bridge has been modeled as a 1D multi-span continuous beam on nine firm supports. Finally, the results of the numerical tests carried out by varying the ML velocity display the effects induced by the bending stiffness and the damping of the beam on the resulting critical velocities of the physical system. Moreover, the time history of the system subjected to a moving locomotive is depicted, through a sequence of frames and figures reporting the full dynamic response in terms of displacements, velocities and accelerations.

Section 7, eventually, delivers the most prominent conclusions of the present work, by outlining its main thesis. Also, potential future developments are briefly mentioned.

1.6 Further Information

Concerning notation, terminology and adopted computational tools and resources, the following specific information may be reported, on behalf of the interested reader, possibly helping for independent reproductions of the present developments.

Statements similar to those of a typical structured programming language, such as C++, are employed in outlining the developed algorithms. Examples of such statements include: **if**…**then**…**else**…**end if**, **loop**…**end loop**, **for each**…**do**…**end for**. Algorithms will also here be reported as *pseudocodes*, which constitute an informal high-level description of the underlying operating principle of a computer program. In fact, those algorithms have been sketched during the coding phase, to assemble the present C++ parallel implementation, but they could also be followed in creating a new computational program, independently from the selected programming language and implementation platform.

All plots in this work have been generated by `matplotlib` [84], which is an open-source Python 2D plotting library, producing publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Instead, illustration sketches have been created by LatexDraw [9], a graphical drawing editor interface for LaTeX, developed in Java.

The C++ parallel FEM code has been implemented into a Unix-like environment (Ubuntu LTS 16.04) using Vim. All libraries (Eigen [62], PETSc [10], Open MPI [115]), compilers (GNU/G++, mpic++) and external programs (GMSH [67], METIS [92]) linked to the current implementation are open-source and freely available in the Internet.

The final parallel implementation has been compiled and run on GALILEO and MARCONI HPC platforms of the Cineca consortium, Italy.

2 State of the Art on Moving Load Analysis

This section presents a brief review on the subject of the dynamic response of beams under "Moving Load" (ML), and attempts to provide a useful introductory information to this topic, by focussing on some specific aspects involved in the modelization and the relevant treatment, as detailed below.

It is common to classify a "ML problem" as that particular mechanical problem occurring when the point of application of concentrated or distributed loads acting on a certain structural element changes along time, at a constant or variable velocity. This characteristic makes the ML problem a fundamental topic of structural dynamics. Timoshenko [149] reported that in the middle of the nineteenth century there seemed to be no agreement among engineers about the effect of a ML on a beam. It likely was the collapse of Stephenson's Bridge in 1847 that accelerated the experimentation on dynamical problems involving MLs. The first approach was to build a railroad track and a carriage with two axles, in order to simulate the force acting on the rails, as reported and sketched in Timoshenko [149]. Such a test was conducted for the first time by Willis et al. [156], with the purpose of discovering the dynamic deflections of a beam under a given ML. They discovered that displacements were larger than those produced by a static load with an equal magnitude. It was pointed out that the bars could be critically damaged by a ML, while structural integrity was not compromised by the same static load. It was probably Stokes [144] the first researcher who proposed an exact solution and then provided a possible physical interpretation of the ML problem.

Since that time, many different approaches to the ML problem and more accurate modelizations have been developed. ML problems, which specifically arose in the railway framework, have found their application within several various engineering contexts, such as in the analysis of the interaction between a vehicle and a road ground, the simulation of a tool acting on a rotating craft or the examination of the interaction between brake calipers and discs. In the present computational analysis, only those cases which can be modeled as a beam subjected to a ML will be considered, and the possible presence of different kinds of underlying foundation will be taken into account.

Concerning the analysis of *infinite* beams, namely beams of an unbounded spatial extension, one of the most common approaches has been that of considering the *steady-state* vibration response of the mechanical system, by virtue of its great theoretical and practical appeal. In fact, during the analysis and design stages, a railway track may be well modeled as an infinite beam under support, in order to correctly represent the large extension of the rail and the corresponding arising wave propagation phenomena. When a steady-state approach is considered, the mathematical model may lead to a deeper understanding of the wave propagation occurrence within the analyzed structure. By supposing that the effects of transient motion due to the initial conditions have vanished, the result of such a simplification is a beam response that only depends on the action of the ML. In fact, as stated by Frýba [59], the dynamic response, in terms of transverse displacement, is depicted as a traveling wave, moving at the load velocity. Thus, by further considering a moving coordinate system, called "convected coordinate", attached to the ML position, the dynamical problem

becomes a time-invariant one, and may be analyzed as in statics (see e.g. Froio et al. [55]).

Many authors have dealt with the steady-state response of an infinite beam under a constant-magnitude ML (see, for instance, Kenney [94], Lu and Deng [103], Mallik et al. [104], Basu and Rao [11] and Froio et al. [55]), while others have focused on a harmonic-varying amplitude of the ML (see Mathews [106, 107], Chonan [26]).

Between the latter, Bogacz et al. [15] proved the existence of four sinusoidal traveling waves gathered into two groups, one ahead and one behind the load position. When the load displays a certain frequency and a velocity equal to one of the two groups of waves, the energy is accumulated in the load vicinity. So, in the ideal undamped case, the response of the beam approaches infinity, for those velocities defined to be as critical, the so-called "critical velocities".

Recently, a new Discontinuous Least-Squares FEM formulation was developed in the modelling of the steady-state response of an infinite supported taut string under ML (2nd-order differential problem), see Froio et al. [57], and then of an infinite beam under ML (4th-order differential problem), see Froio et al. [58]. These works also properly consider the effects of absorbing boundaries in terms of an efficient Perfectly Matching Layer (PML) implementation, displaying results that are fully consistent with available analytical solutions, and resolving several issues of inconsistency in existing traditional FEM formulations for steady-state problems, as in overcoming numerical instabilities that may be linked to high-velocity propagating loads and in correctly handling the far-field conditions.

Of course, the steady-state approach turns out to be unsuitable, in principle, for analyzing the response of a *finite* beam. In fact, an infinite beam may display a time-unlimited response while, in the case of a finite beam, the solution shall always be bounded, since, at a certain time instant, the travelling load has to leave the beam, then setting to zero the external excitation. Despite the absence of an unbounded response, in the analysis of the resonance condition, the critical velocity of a finite beam may still be defined. Two different definitions may be found in the literature. According to Chen and Huang [23], the critical velocity is then defined as the lowest between the modal resonant velocities of the system. On the other hand, it may also be defined as the load velocity inducing the maximum transverse displacement, as stated by Dimitrovová and Varandas [39].

Another characteristic feature of the dynamic response of finite beams under ML is the phenomenon of wave reflection, which is absent for the wave propagation within unbounded media. The reflection of waves induces important increments of beam displacements. As explained by Steele [143], the head wave front reflects from the far end before the ML reaches the beam extreme. So, at a certain instant of time, near the beam ends, interference phenomena occur between the incident and the reflected waves. The possibility of a constructive (build-up) interference among such waves implies that the maximum displacement for a finite beam shall be higher than that for the steady-state maximum response of an infinite beam.

During the sequel of the present section, the variety of solution approaches adopted by different authors in dealing with the analysis of the response of *finite* beams under ML and the investigation of their resonance conditions will be illustrated. Several criteria for classifying the research works in the field of ML problems may be adopted, see for instance the review information already included in Froio et al. [55, 56]. In the following exposition, a methodological classification is chosen, namely that based on the solution strategy employed toward solving the ML problem. Therefore, literature contributions will be analyzed and presented by organizing them into two main categories:

- *Analytical* approaches;
- *Numerical* approaches.

Due to the large amount of publications on the topic, the interested reader may as well consider dedicated comprehensive literature reviews, as proposed e.g. by Wang et al. [155], Ouyang [123] and Beskou et al. [13].

Accordingly, the present section is divided into two main subsections: Sect. 2.1 presents the analytical formulations developed to solve ML problems involving finite beams, while Sect. 2.2 discusses the numerical approaches adopted for the solution of the same ML dynamical problem. Further, both sections will focus onto two different kinds of *finite* beam ML problem modelizations:

- Beam without foundation, constrained at its ends;
- Beam resting on a (visco)elastic foundation, constrained at its ends.

2.1 Analytical Approaches

As above mentioned, dynamic analyses on ML problems started in the nineteenth century, while during the 1980s, the first numerical experimentations began. Analytical investigations continued because of the crucial need of gaining a better understanding of the problem and making a benchmark reference for numerical tests. A collection of analytical solutions about several different ML problems may be found in the classical monograph by Fryba [59], while some others are presented in the following discussion.

2.1.1 Analytical Finite Beam Without Foundation

Out of many alternatives adopted in the modelization of road and railway bridges, a widespread one is the simply supported beam model, under a ML travelling with a constant velocity, as in Fig. 1; though, other well-posed types of boundary conditions, even elastically compliant ones, may similarly be considered.

This model was probably first solved by Krylov [99] and, shortly after, by Timoshenko [147], who applied the method of harmonic analysis. Other solutions worthy to be mentioned are those by Jeffcot [89], Inglis [85], Lowan [102], Kolousek [98]. Some years later, Fryba [59] obtained the dynamic response of the beam by using the Finite Fourier sine transformation and dwelling on the concept of critical velocities, as those values leading to a resonance effect recorded within the structure.

Such pioneering works encouraged further investigations on the subject, such as the simulation of vehicular traffic loads on a bridge. Iwankiewicz and Sniady [87] developed a technique that was able to predict the variability of the beam deflection from a new stochastic viewpoint. Force arrivals were assumed to be described as a Poisson process of events. They argued about the validity of the Poisson process to idealize the traffic load and sought for an explanation of the apparent induced effects. In fact, they found that the value of beam deflections decreases with an increasing force speed. It was clear that the average of the total loads on the beam decreased with an increasing speed. Actually, it was well-known yet that the dynamic response to a single ML is amplified by an increasing velocity.

Olsson [121] discussed the dynamical problem of a simply supported beam subjected to a constant-amplitude force moving at a constant speed. Throughout that paper, an analytical solution based on the separation of variables was presented and compared with those numerically obtained by the FEM. He noticed that the ratio between the transverse deflection, $w(x, t)$, and the same displacement evaluated in the middle of the beam, $w(L/2, \mathrm{t})$, depends on three non-dimensional parameters: x/L, t/τ and α, where τ is the traversing time of the moving force and $\alpha = T_1/(2\tau)$ in which T_1 is the period of the lowest vibration mode of the beam. Typically $\alpha = 1$ corresponds to a vehicle velocity of $v = 400 \div 1500\,\mathrm{km/h}$, depending on the structural flexibility/rigidity, while $\alpha = 0$ corresponds to the static case. Time histories for mid-span displacement and moment were displayed, highlighting a similar behavior. However, the moment histories seemed to be more irregular, and a noticeable negative mid-span moment was obtained, when the force entered the beam. In the end, the size of discretization errors introduced by a FEM approach were pointed out, in contrast with the exactness of the analytical solution.

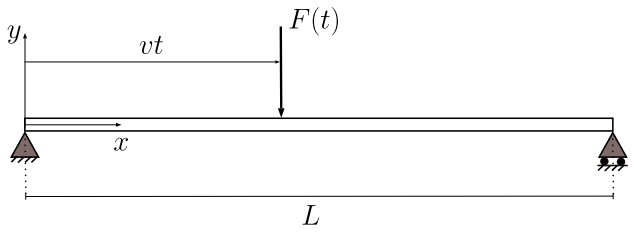


Fig. 1 Simply supported beam model under transverse concentrated moving load

The vibration of simply supported beams subjected to the passage of high speed trains was investigated by Yang et al. [161]. The analytical solution was obtained by modeling the train as the composition of two subsystems, traveling with a constant speed, one for the front and the other for the rear assembly of the wheels. A closed-form solution for the dynamic response of the beam was derived, the resonance condition was found and some criteria were given to select the span length and the cross-section of the beam.

Abu-Hilal and Zibdeh [2] analyzed an elastic homogeneous isotropic beam with general boundary conditions traversed by a ML, with accelerating, decelerating and constant velocity types of load motion. The dynamic response in a closed-form solution employing the modal form of the transverse displacement was obtained. Different cases of boundary conditions and damping were presented. It was shown how transverse vibrations were amplified by a uniformly accelerated or decelerated ML. Moreover, the influence of boundary conditions on the value and position of the maximum displacement along the beam was highlighted.

In a next article, Abu-Hilal and Mohsen [1] introduced a harmonic ML, to study how its frequency affected the maximum dynamic deflection. When the frequency was equal to the first natural frequency of the beam, displacements reached their maximum values. Also, the velocity of the load turned out to be significant: the maximum response decreased by an increasing load velocity, because the acting time of the harmonic load became shorter.

For the purposes of track maintenance, the phenomenon of resonance was also investigated by Yau and Yang [162]. The vertical acceleration response was taken into account, in case of a simply supported beam, under a series of MLs at a high constant speed. Considering a train as a series of MLs with regular intervals, a closed-form solution was developed, by a superposition method. The results indicated that the contributions of the higher modes to the beam acceleration could not be neglected, for beams with a light structural damping. In particular, the maximum acceleration of the beam did not occur at the mid point, when resonance was excited by the higher mode frequencies, which corresponded to the characteristic frequency of occurrence of the MLs.

Museros et al. [116] studied not only the beam free vibrations but also the effects of resonance and cancellation

phenomena. If the resonance phenomenon occurs because of the continuous build-up of the modal responses of the bridge, the cancellation phenomenon implies that waves may annihilate each other. A new closed-form expression was provided, for the cancellation speed of a generic mode of a simply supported beam. It was of a great interest to a priori know the speeds of the maximum free vibrations or cancellations, in order to conduct experimental tests on real structures. In fact, with this information, it would be made possible to obtain refined measurements of the dynamic properties of bridges, such as for estimating the amount of inherent structural damping.

A closed-form solution for evaluating the dynamic behavior of a general multi-span Euler–Bernoulli beam was derived by Johansson et al. [90]. A bridge was modeled as a certain number of simply supported beams whose behavior was controlled by a Partial Differential Equation (PDE), connected to adjoining beams by means of boundary conditions. The PDE was first transformed into a modal equation, assuming it to be a linear combination of normal modes, thanks to the expansion theorem. Then, the natural frequencies and the normal modes were determined. Furthermore, the modal equation was solved within the frequency domain, using a Laplace transform, to arrive at a closed-form solution. Finally, the displacement, velocity and acceleration responses were obtained within the time domain by an inverse Laplace transform, and subsequent derivation with respect to time.

Xia et al. [159] investigated the mechanism of vibration resonance and cancellation for a simply supported bridge subjected to a ML series. The analysis was theoretically led, by using free vibrations, and then verified through a FEM simulation. It was explained that the superposition of vibrations induced by a series of MLs could cause a resonant vibration, when the natural period of the bridge turns out equal to the time interval between the adjacent loads. Two types of cancellation effects were detected in the observed case. The first one dealt with the free vibration of the bridge induced by a single moving force. This free vibration may cancel out by itself, whether the load speed satisfies a relationship between the span length and the natural frequencies of the bridge. The second one implied that, under a certain relationship between the load speed, the load interval and the natural frequencies of the bridge, the free vibrations of the beam induced by a series of MLs may cancel out. Furthermore, an analysis of the case in which both resonance and cancellation occur was provided. Proofs were produced about the fact that the cancellation plays a dominant role in getting rid of the resonance condition and then giving some important parameters toward bridge design.

Kumar et al. [100] studied the cancellation phenomenon characterizing the free vibration response of a simply supported beam, after the transition phase of a single concentrated ML, providing a simple closed-form expression for both lightly and heavily damped beams. Expressions for the cancellation speeds were derived, finding a relationship between the free-vibration amplitude, the phase angle and the initial conditions.

Di Lorenzo et al. [32] dealt with the vibration response under ML of uniform Euler–Bernoulli beams, with translational supports and rotational joints obeying to a Kelvin-Voigt viscoelastic behavior. The aim of that paper was that of developing an analytical method that could handle loads traveling on multi-span beams. A novel modal superposition approach was introduced, to provide an analytical expression of the beam response. The discontinuities due to the supports and joints were handled using the theory of generalized functions. Two examples of application were presented by the authors: a three-span beam, held by two viscoelastic translational supports, assuming clamped-clamped boundary conditions, with rotational dampers at the ends of the beam; a simply supported four-span beam, with Kelvin-Voigt viscoelastic translational supports and rotational joints. For both tests, the deflection profile of the beam at different time instants was determined.

An original approach to the ML problem on an Euler–Bernoulli beam with Kelvin-Voigt viscoelastic translational supports and rotational joints was proposed by Adam et al. [3]. Additionally, the structure was equipped with viscoelastic Tuned Mass Damper (TMD) control devices (see, e.g., Salvi and Rizzi [134–136], in the context of non-stationary and seismic excitation; Pioldi et al. [126] and Salvi et al. [133], in the field of Soil-Structure Interaction and modal dynamics system identification within that; Salvi et al. [137, 138], in the realm of controlling the mechanical response to pulse-like excitations), as a mean of reducing the vibration amplitude induced by MLs. In that paper, a closed-form analytical response was achieved, within the time domain, for those mechanical systems subjected to MLs travelling with a constant velocity, in the presence of any number of TMDs, supports and joints. In order to treat the discontinuities given by the rotational joints, the theory of generalized functions was used; so, the exact complex eigenvalues and eigenfunctions of the beam were obtained, by imposing an orthogonality condition. In that way, all response variables were expressed by time-domain convolution integrals. Furthermore, no numerical integration was required, as instead performed in other FEM approaches. The proposed exact solution may be employed as a benchmark one for applications with standard FEM codes.

2.1.2 Analytical Finite Beam with Foundation

The models without foundation could be enriched by the insertion of an underlying (visco)elastic supporting medium, as portrayed in Fig. 2.

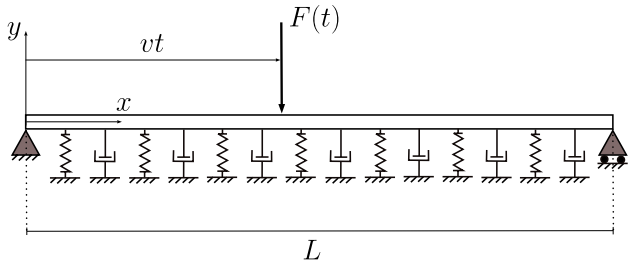


Fig. 2 Simply supported beam-(visco)elastic foundation model under transverse concentrated moving load

Among the existing mechanical models, such as those of Pasternak or of Kerr, or that of the underlying elastic continuum, the Winkler foundation one owns a leading position among engineers and researchers, because of its mathematical simplicity and applicability to several practical situations. As described in Winkler [157], Winkler's assumption was to consider the foundation as an array of infinitely closed linear elastic springs with a symmetric behavior under tension and compression (see also competent review information provided in Froio and Rizzi [52]). Winkler's elastic stiffness coefficient $k(x)$ (N/m^2), also known as foundation modulus, was introduced as

$$k(x) = \frac{r(x)}{w(x)} \tag{1}$$

where $r(x)$ (N/m) is the transverse reaction force acting on the beam, $w(x)$ (m) the deflection of the beam and x (m) a longitudinal spatial coordinate. Thanks to this model, it was possible to investigate the interaction between a beam and the underlying foundation, as elastically deformed by different kinds of static and dynamic loadings.

Regarding the Winkler model, the stiffness of the elastic springs is usually assumed to be constant but, in the literature, some examples of a variable stiffness coefficient could be found, either in the static or in the dynamic analysis. A static analytical solution was presented by Froio and Rizzi [52], for a simply supported Euler–Bernoulli elastic beam lying on a *variable* Winkler elastic foundation. The analytical solution, in a nonlinear case describing the Winkler foundation modulus as

$$k(x) = (c_0 + c_1 x)^{-4} \tag{2}$$

was presented in closed form, and a complete parametric study was carried out to inspect the influence of the mechanical properties on the behavior of the beam-foundation system. The analytical solution was in perfect agreement with numerical predictions but the advantage of owning an exact solution shall become evident, when a numerical method shall need to be validated or also in view of practical applications.

Similarly to such a previous work, Froio and Rizzi [53] later considered the analytical static bending response of a finite uniform free-free Euler–Bernoulli beam resting on a Winkler elastic foundation with a linear variation of the support coefficient, as e.g. resembling the configuration of a vertical pile embedded into the ground. The stiffness coefficient was then described through the following relation

$$k(x) = k_0 + \frac{k_L - k_0}{L}\, x \tag{3}$$

in which k_0 and k_L (N/m^2) are the point-wise values of the spring stiffness, respectively, at the top and bottom ends of the beam. That paper dealt with the explicit closed-form analytical solution of a beam-foundation system loaded by a force and a moment applied at one beam's end.

Pavlovic and Wylie [125] investigated the natural vibration response of a beam restrained by an elastic foundation of a linearly varying modulus along the beam span, by using the infinite power series method. A completion of the previous method for an arbitrary variation of the foundation modulus, even non-analytical, if the singularity is of a regular type, was presented by Foyouzat et al. [49], by using the Frobenius theorem.

Dealing with the analytical investigation of the ML problem with underlying foundation, Dimitrovová and Varandas [39] implemented two analytical approaches, in order to study a beam, supported by a foundation presenting a stiffness change, subjected to a force moving with a constant velocity. The analysis of that case shall become fundamental, because of the constant diffusion growth of high-speed lines. A strong variation in the stiffness of the track-foundation system causes excessive ground vibration, which may compromise vehicle stability and cause passenger discomfort. When the supporting structure changes, additional waves are generated and the dynamic response may significantly become much severe. Two different ways of analytically solving the problem are collected in that paper: a generalized method of integral transformations and a combination of the differential equations used to describe the two half beams. The analysis was performed with a parametric approach and it displayed that the amplitudes of the maximum displacements are highly amplified passing from the softer to the harder region.

The response of a simply supported beam on an elastic foundation subjected to repeated moving concentrated loads was obtained by Amiri and Onyango [6], by means of a Fourier sine transformation. The procedure and the effects of some parameters were illustrated with numerical examples, adopting two concentrated loads moving along the beam. It could be seen that the results were affected by the foundation stiffness, the traveling speed and the magnitude of

the MLs. In particular, an increase in speed could lead to amplify the dynamic deflections, for a characteristic range of velocities, and an increment of foundation stiffness could bring to higher values, either the natural frequencies or the critical speed of the mechanical system.

Dimitrovová [33] meant to deepen the effect of changes in stiffness of the foundation. For instance, the effects on the dynamic response of the beam were analyzed, to achieve a better understanding on tunnel transitions or on phenomena related to entering or leaving bridges. In that paper, the transverse vibration was examined, as induced by a load moving at a constant speed along a finite or almost infinite (very long) beam resting on a piece-wise homogeneous visco-elastic foundation. The changes in stiffness foundation were considered, together with the discontinuities in the supporting structure created by track degradation or alteration of structural design. The governing equations were solved by normal-mode analysis and Laplace-Carson integral transform. Using the dynamic stiffness matrix method introduced by Chen et al. [24], the natural frequencies were determined and the solution was then established, adding an intermediate region of adaptable foundation stiffness.

Dimitrovová and Rodrigues [38] carried out an investigation on a finite or almost infinite (very long) simply supported beam possibly composed of two subdomains. The governing equations were solved by a modal expansion technique, both considering Euler–Bernoulli and Timoshenko beam theories. Moreover, the critical velocity and the influence of damping on it were taken into account. It was demonstrated that the critical velocities of a finite beam constitute an overestimation of those for an infinite one, because of the effects of wave reflections. For the finite beam composed of two sub-domains, the harder sub-domain acquires another displacement peak, which corresponds to the critical velocity of the softer sub-domain. Finally, it was stated that also a low level of damping is enough to smoothen the resulting peaks, for both the upward and downward displacements.

Investigating the behavior of another type of foundation, different from the Winkler one, Dimitrovová [34] derived a new formulation for the critical velocity related to a beam supported by a foundation of a finite depth, instead of a continuously supported beam resting on a uniform layer of closely spaced, mutually independent linear elastic springs. A simplified plane model of the foundation was built and some results were presented, either for the finite beam or for the infinite one, using also some different values of damping. Thanks to a parametric analysis and to the introduction of three dimensionless parameters (velocity ratio, shear coefficient and mass ratio), the new formulation was validated and it was found that the mass ratio, defined as the square root of the fraction of foundation mass to beam mass, strongly influences the resulting critical velocity.

2.2 Numerical Approaches

2.2.1 Numerical Finite Beam Without Foundation

To the Authors' knowledge, the FEM was first applied to a ML problem by Yoshida and Weaver [163]. An analysis technique was developed to predict the dynamic response of lightweight highway bridges due to the passage of semi-trucks and other large vehicles. In that paper, the FEM was used to model continuous beams, with different support conditions, traversed by constant-velocity and constant-acceleration MLs, with or without the influence of a mass associated with the load. The moving-force problem was converted into a finite number of differential equations of motion, for the discretized structure, and then the normal-mode method was employed to obtain the displacements of the nodes in terms of a Duhamel integral. The numerical integration of the second-order equations was carried out with the linear acceleration method based on the assumption that the nodal accelerations linearly vary during a small time interval.

A general bridge-vehicle finite element was defined by Olsson [120], in order to investigate the arising interaction phenomenon. Assuming a linear structural behavior, that specific finite element displayed time-dependent and unsymmetric matrices and it was used to obtain a modal formulation of the bridge motion. The advantage of such a technique was that of reducing the number of the motion equations, truncating the higher vibration modes. So, although the modal equations were coupled, the set of linear equations with a time-dependent coefficient was made solvable through the Newmark method.

Rieker et al. [130] studied the discretization for FEM models related to a ML acting on an elastic beam. That work investigated the relationship between model accuracy and number of elements used to represent the beam. The discretization turned out to be important in establishing the size of the computational error and the elapsed time in the developed simulation. It was stated that the interpolation error potentially affects the calculation of the equivalent nodal reactions, of the transmitted force, from a moving sprung mass, and of the overall system response. A comparison was made, in order to establish the number of finite elements required to achieve an acceptable static or dynamic deformation analysis. The results showed that for a ML-type problem, the mesh density must be increased, when the traveling speed increases. Moreover, adaptive meshing may be required in the area around the moving force, to ensure high accuracy, within a reasonable computational cost.

An exact dynamic stiffness element under the framework of a FEM approximation was presented by Henchi et al. [75]. That paper was about the dynamic response of multi-span structures under a convoy of MLs. Using the virtual work expression for the transverse vibrations of Euler–Bernoulli

beams, the weak formulation for a discretized structure was obtained. Instead of employing the traditional Hermite-type polynomial functions, they introduced a characteristic nodal approximation, which led to an exact evaluation of the frequencies and the modes. The dynamic response of multi-span beam-structures could be reduced to a numerical solution, using the discrete Fourier transform implemented as a FFT algorithm.

Wu et al. [158] studied how to improve the control of mobile gantry cranes used in ports and rail-head freight yards. A one-dimensional model of a gantry crane was developed, employing a simply supported beam subjected to a single ML or a pair of forces. A computer program was written, to calculate the time-variant external forces on the whole structure, providing the equivalent loads that move around it. In that way, a model was assembled and it was able to reveal how the forces acted on the structure, in order to correctly size the crane.

A dynamical simulation of multi-span bridges modeled as simply supported beams was developed by Ju and Lin [91]. The three-dimensional model represented a vehicle-bridge system and it analyzed the passage of a high-speed train. Three-dimensional beam elements were adopted and the Least-Squares method was employed to calculate the stiffness, the damping and the mass matrices. A SKS-700 high-speed train transit was simulated, using the non-linear Newmark direct integration method, which converged, averagely, within two Newton-Raphson iterations. Then, the resonance effect was investigated; so, an analysis was performed to find the bridge natural frequencies. It was proven that, to avoid resonance effects, the dominating train frequencies should be as much different as possible from the bridge natural frequencies. In particular, if the two first frequencies are similar, bridge resonance may cause serious damages.

Martínez-Castro et al. [105] presented a semi-analytical solution of the ML problem applied to multi-span uniform and non-uniform beams. The beam was spatially discretized using conventional Euler–Bernoulli finite elements with two nodes, individually owning two degrees of freedom. Linear variations of the area and depth of the cross-section were assumed, leading to a linear variation of the mass per unit length and a cubic variation of the second moment of inertia of the cross-section. When the element equations of motion were built, the assembly of each contribution led to a system of linear differential equations with constant coefficients and an analytical right-hand term. In fact, in that method, the load term is a vector, with an analytical polynomial expression, defined on a temporal interval equivalent to the time spent by the ML on a single element. Then, the differential equation was defined for the nth mode and the analytical expression was obtained in closed form, piecewise defined for every different time interval. So, the only approximation introduced in the procedure came from the spatial FEM discretization of the beam, while the solution was expressed in terms of ten coefficients per element and per mode, the values of whose were independent from the travelling speed of the ML. This semi-analytic procedure was applied to the analysis of a set of simply supported as well as continuous multi-span beams, by implementing a FORTRAN computer code.

Salcher and Adam [132] proposed a three-dimensional mechanical model for analyzing the dynamic interaction of a train passing on a railway bridge at high speed. For the complexity of the system, the equations of motion were separately derived, for the bridge and the vehicle subsystems. Subsequently, a coupling of the bridge with the train was achieved through a Component Mode Synthesis (CMS) methodology, which is an efficient sub-structuring technique based on modal analysis. The full FEM bridge model was built using 3D elements with six degrees of freedom per node, while Euler–Bernoulli finite beams were chosen to represent the rails. They considered the train made up of a certain number of wagons, modeled in five different parts: the primary and the secondary suspension systems and the passenger, the bogie and the wheel stages. Once the equations of motion for each subsystem were obtained by employing the Lagrange equations, the CMS method was applied. It consisted of three different steps. In the first step, the equations of motion of the subsystems were separately derived, in nodal or generalized spaces. Then, the component mode coordinate transformation was formulated, according to the compatibility conditions between the train and the bridge. Finally, the coupling was fulfilled through a coordinate transformation, imposing the equilibrium conditions defined at the interface. Moreover, using an ABAQUS FEM model, they displayed the dynamic response of a simply supported single-span steel railway bridge, taking into account also the irregularities of the rail profile. Additionally, the maximum absolute deflection was presented at different locations of the bridge, as a function of the travelling speed; also, they displayed some mode shapes of the bridge model.

2.2.2 Numerical Finite Beam with Foundation

A FORTRAN program was implemented by Thambiratnam and Zhuge [145], for the dynamic analysis of a beam on an elastic foundation subjected to MLs with a constant propagating velocity. Using the Lagrange's equation, the element matrices and vectors were obtained, and the equations of motions were integrated within the time domain, by the Newmark method. Some parameters such as the length of the beam, the speed of the ML, and the magnitude of the foundation stiffness were studied, to investigate their influence on the recorded dynamic response. The procedure was applied to the analysis of railway tracks, approximating the response of an

ideal infinite beam with a (long) finite beam FEM structure. Furthermore, the effects of two consecutive MLs were taken into account. The reported beam deflections were similar to those detected for a single ML, even though the time duration of the beam response was significantly increased.

Chang and Liu [22] performed a deterministic and random vibration analysis on a nonlinear beam on an elastic foundation, excited by a ML travelling with a constant velocity. In order to obtain the dynamic response of the beam, the Galerkin method was employed, together with the FEM. The nonlinear system of differential equations was linearized, by using the incremental method and it was solved by the implicit form of the Newmark time-integration algorithm. Using a Monte Carlo approach, the randomness of the beam profile, in terms of beam axis variability was studied. Then, the responses coming from the linear and nonlinear beam models were compared: it was displayed that the standard deviation of the midpoint deflection of the beam for the linear model was always the largest among those of the nonlinear models.

The dynamic analysis of a pre-stressed Euler–Bernoulli beam resting on a two-parameter elastic foundation was described by Kien and Hai [95]. The FEM was used to investigate the response of the beam under a moving harmonic load. The finite beam stiffness matrix was found, by employing cubic Hermitian polynomials, as interpolation functions for the deflection. Then, the expression of the potential energy due to beam bending, the foundation stiffness and the potential due to the axial load were considered. The foundation stiffness was characterized by either the stiffness of Winkler springs or the contribution of a shear layer. So, that model showed a greater accuracy in modelling the effect of the foundation support on the structure. The dynamic response of a simply supported beam was obtained, with the aid of Newmark direct integration method. Using the above-mentioned formulated element, the dynamic analysis of the beam with different values of axial force, ML frequencies, foundation parameters and velocity was performed, in order to investigate the changes in terms of natural frequencies. Two years later, Kien [96] carried out the same kind of analysis but employing a Timoshenko beam model.

Ansari et al. [7] treated the vibration of a finite Euler–Bernoulli beam supported by a nonlinear viscoelastic foundation traversed by a ML. The Galerkin method was used to discretize the system, by obtaining a local expression of the transverse displacements. The nonlinear foundation presented in that article followed a constitutive law different from classical linear Winkler's one. It was expressed as

$$r_{nl}[w(x,t)] = k_l w(x,t) + k_{nl} w^3(x,t) \tag{4}$$

where $r_{nl}[w(x,t)]$ is the force generated by displacements $w(x, t)$ from the equilibrium configuration, k_l is the linear part of the foundation stiffness, while k_{nl} is the nonlinear one. The dynamic response was obtained for different harmonics using the Multiple Scales Method (MSM), as one of the perturbation techniques. Then, the effects of the damping, of the ML magnitude and of the nonlinear stiffness on the achieved solution were examined through a parametric analysis. Like that, the resonance condition and the frequency responses of different harmonics were displayed.

Ding et al. [40] dealt with the convergence of the Galerkin method as applied to the dynamic response of an elastic beam resting on a nonlinear dissipating foundation. The latter was represented by a constitutive equation that was similar to that in Eq. (4) but with an additional parameter, in order to introduce a coupled presence of damping:

$$r_{nl}[w(x,t)] = k_l w(x,t) + k_{nl} w^3(x,t) + c\frac{w(x,t)}{t} \tag{5}$$

in which c is the damping coefficient of the foundation. So, using Hamilton principle and Euler–Bernoulli theory, the governing differential equation of motion was written for the beam. Then, adopting the Galerkin truncation method, the system was discretized. By a fourth-order Runge–Kutta algorithm, the spatially discretized equation of motion was solved, for different types of beam constraints: clamped-clamped, free-free and simply supported. Considering the UIC60 European high-speed rail for their tests, the effects of foundation stiffness, beam length and damping coefficient on the dynamic response were investigated. It was found that the effects of the boundary conditions became smaller, when longer span lengths of the beam were chosen. Moreover, the convergence of the method increased with the growing modulus of elasticity of the beam, nonlinear foundation parameter k_{nl} and span length, while decreased with greater k_l and c parameters.

Castro Jorge et al. [19] focused on the dynamics of simply supported beams on a non-uniform nonlinear foundation acted upon by a moving concentrated force. The effects of the load's intensity and velocity, of the damping and of the foundation stiffness on the critical velocities were investigated. The analysis was based on a FEM formulation of the problem, built on the Lagrange equations. Considering the elastic strain energy of the beam and the foundation, and then the kinetic energy of the finite beam, the element matrices were built. Dealing with a nonlinear foundation governed by Eq. (4), the potential due to the foundation forces acting on a finite element was found. It was possible to describe the nonlinear behavior of the system, from the equations of motion, which contained the nonlinear force vector expressed as a function of the generalized displacements of the finite element. Viscous damping was taken into account by considering classical Rayleigh damping. The governing differential equations of motion were solved by means of the HHT α -method, implemented within a MatLab environment [146].

Some meaningful conclusions were derived, for both uniform and non-uniform foundations, which consisted of two sub-domains, with different stiffness constants. A reduction of the maximum displacements and a higher critical velocity could be observed if the foundation stiffness increased. Instead, the damping had no apparent effect on the critical velocity, even though it reduced the maximum values of the resulting displacement. Furthermore, in case of a two sub-domain foundation, independently from the position of the stiffer region, the maximum displacement took place when the load was on the second sub-domain. Then, it was noticed that the second sub-domain exhibited two critical velocities: its critical speed and that of the first sub-domain, as shown by Dimitrovová and Rodrigues [38]. The important aspect of the presence of friction and its effects were further considered by Toscano Corrêa et al. [151, 152].

Froio et al. [51] investigated the transient response of a simply supported Euler–Bernoulli beam resting on a homogeneous in space Winkler elastic foundation. Two types of foundation constitutive law, linear and cubic, were taken into account. Moreover, a beam subjected to a transverse concentrated load moving at a constant velocity with a harmonic-varying magnitude was considered

$$F(t) = F\cos(\Omega t) \tag{6}$$

where F is the reference amplitude, Ω the angular frequency of the ML variation and t the generic time instant. For the purpose of solving the governing linear/nonlinear partial differential equation of motion, a FEM approach was employed, and the system was discretized in space using cubic Hermitian polynomials, as interpolation functions for the unknown transverse displacements. Finally, the dynamic response was obtained by a HTT-α algorithm. A code was implemented within MatLab and the output was validated using the results earlier reported by Castro Jorge et al. [19]. Furthermore, the effects given by the velocity and the frequency of the load were reported, portraying the results for the damped and the undamped cases. It was noticeable that the critical velocities might be one or two, in case of a harmonic ML. The lower critical velocity approaches zero at a very high load frequency, while the higher critical velocity moves toward higher values. Then, using appropriate analytical bifurcation curves, the relationship between the ML frequency and the critical velocity of the beam was derived. The resulting curves showed a very good agreement to those provided by Chen et al. [25], who employed an analytical approach.

Froio et al. [54] further extended the results earlier obtained by Froio et al. [51], through a parametric analysis on the dynamic response of a simply supported elastic beam resting on a spatially homogeneous nonlinear cubic (visco)elastic foundation. Thanks to extensive numerical simulations implemented into an autonomous FEM code, the dependency of the critical velocities on the mean value of a harmonic ML was explored. This time, the harmonic force was described as

$$F(t) = F_0 + F\sin(\Omega t) \tag{7}$$

where the mean value of the force was defined as $F_0 = \alpha F$, F being the reference amplitude of the force and α a parameter employed within the parametric analysis. First of all, the linear and nonlinear foundation results were compared. It was revealed that, for a constant-magnitude ML ($\Omega = 0$), for the linear model, the peak corresponding to the critical velocity was proven to be independent from ratio α; however, for the nonlinear foundation, it was recorded that the peak moves to higher values of critical velocities, at increasing α. On the other hand, for a zero-mean harmonic ML ($\alpha = 0$), the peak bifurcates into two distinct peaks, considering either the linear or the nonlinear model. Then, the much salient result was the manifestation of three different critical velocities, when mean value F_0 of the acting ML was different from zero ($\alpha \neq 0$). It was also noticed that the lowest and highest critical velocities tend to separate, as the load frequency increases, while the central one remains stationary for every Ω, at the value corresponding to $\Omega = 0$.

Up to now, the cited papers were concerning beams lying on a foundation with a symmetric constitutive law, so with the same behavior under tension and compression. In the following, a bilinear foundation is further presented, in order to obtain more realistic models, which not only rely on the amount of displacements induced by the forces on the beam but also on their versus. A bilinear foundation exhibits two distinguished linear force-displacement branches, for the upward and downward displacements, obeying to the law

$$\begin{aligned} r[w(x,t)] = {} & k^{+}\, H[w(x,t)]\, w(x,t) \\ & + k^{-}\, H[-w(x,t)]\, w(x,t), \quad 0 \le k^{+} \le k^{-} \end{aligned} \tag{8}$$

where $w(x, t)$ is the transverse displacement of the beam, $r[w(x, t)]$ the reaction of the foundation, $H(\cdot)$ the Heaviside function, k^{+} the tension stiffness and k^{-} the compression stiffness, as shown in Fig. 3.

Very few researchers seem to have addressed the problem of a bilinear foundation. Between them, Castro Jorge et al. [18] extended the analysis presented in Castro Jorge et al. [19], to the beam lying on a bilinear elastic foundation. Using a previous FEM approach, it was explained how to take into account the contribution of the foundation to the internal forces, so as to generate the correct tangent stiffness matrix, employed within the direct integration of the governing equations. Several analyses were performed, testing different values of load velocity and foundation stiffness. It was observed that when the stiffness to the upward motion decreases, the upward displacements of the beam increase, while the critical velocity decreases.

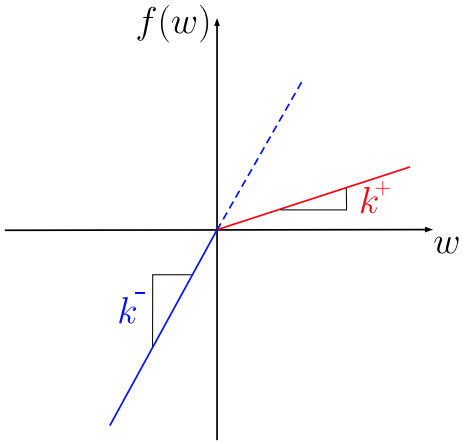


Fig. 3 Constitutive law of a bilinear elastic foundation

Froio et al. [56] concerned with the numerical modelization of the transient dynamic response of a simply supported Euler–Bernoulli beam resting on a bilinear Winkler-type elastic foundation. In that paper, the effect of a transverse concentrated load with a zero-mean harmonic-varying magnitude in time was displayed. Employing a FEM approach coupled with a direct integration algorithm, extensive numerical analyses were performed, to investigate the influence of the ML frequency and of the foundation moduli on the transverse deflections of the beam. Several numerical simulations were carried out, after convergence studies designed to establish the proper space and time discretizations. Analyzing the system response to a frequency variation, it was found that if the ML displays a constant magnitude, only one critical velocity appeared. Increasing the value of the load frequency, two critical velocities were instead observed. The two critical velocities tended to separate: the higher one increased, starting from the value obtained for a constant-magnitude ML, while the second one decreased, until it reached zero for a ML frequency equal to the first natural frequency of the beam.

3 FEM Formulation in the Context of Moving Load Problems

The Finite Element Method (FEM) is one of the most common and widespread numerical methods for determining useful approximate solutions to Partial Differential Equations (PDEs), in various fields of engineering and mathematical physics. Indeed, when intricate geometries, complicated loading conditions and complex material constitutive laws, often nonlinear, may be involved, the derivation of analytical solutions may turn out prohibitive or impossible. Such difficulties become even more accentuated if the sought quantities, which could be a temperature, a displacement or a stress field, may display a multi-dimensional space distribution and a temporal variation.

The FEM is based on the concept of approximating the unknown solution as a linear combination of given "shape functions", thus traducing the weak form of an initial-boundary value problem for a given PDE, or a system of PDEs, into an ordinary differential/algebraic system of equations (linear or nonlinear), becoming amenable to effective numerical solutions. Based on a convenient approximation, the real power of this method is the ability to transform a continuous differential problem into a discrete algebraic one, endowed with a finite set of unknowns.

In this section, the basics of the FEM, as related to its application to the field of ML dynamical problems, with specific reference to the context of planar one-dimensional beams are outlined. A brief introduction to the Weighted Residuals Method (WRM) is first provided in Sect. 3.1. Then, the general FEM formulation is overviewed in Sect. 3.2, through a classical Galerkin's approach. The specific application of the FEM to ML structural problems is described in Sect. 3.3, where matrices and vectors related to a beam finite element, with or without underlying foundation, are derived. Finally, in Sect. 3.4, the HTT-α method is introduced, as an efficient numerical integrator, useful for determining the transient beam dynamic response, within both the linear and nonlinear contexts, with pertinent algorithm representations.

3.1 A Brief Introduction to the Weighted Residuals Method

With the purpose of deriving a FEM formulation for a given system of differential equations, a variational or weak statement of the problem is needed. In order to obtain a weak formulation, different approaches could be followed. In that sense, the Virtual Work Principle shall constitute a powerful tool, endowed of a physical appeal, providing a convenient framework for producing a general FEM approximation. Other approaches for deriving a weak form are based on variational principles, according to which specific *functionals* are minimized (for instance, stationary of the total potential energy, Euler–Lagrange equations, stationary of Least-Squares functionals, see e.g. Froio et al. [57, 58]). Furthermore, the Weighted Residual Method (WRM) may be applied. This latter mathematical framework is adopted and discussed, through the classical concept of *weight* functions, in order to derive the weak governing equations from which FEM matrices and vectors may be derived, as illustrated in the following subsections.

As e.g. introduced by Cook [28], a mathematical statement of a physical problem in its *strong* form may be formulated as

$$\begin{cases} Du(x) - f(x) = 0 & \text{in } \Omega \\ Bu(x) - g(x) = 0 & \text{on } \partial\Omega \end{cases} \tag{9}$$

where D and B are two differential operators, x is the independent variable or a set of them, u is the unknown solution of the problem (for instance, a displacement field), and f and g are two given functions of x. The domain of definition of the problem has been labeled as Ω, while $\partial\Omega$ marks the frontier of Ω, on which boundary conditions are set. In the context of transverse bending of a beam, Eq. (9) becomes

$$\begin{cases} EI\frac{\partial^4 w(x)}{\partial x^4} = -q(x) & \text{in } \Omega \\ EI\frac{\partial^2 w(x)}{\partial x^2} - M(x) = 0 & \text{on } \partial\Omega \\ EI\frac{\partial^3 w(x)}{\partial x^3} - V(x) = 0 & \text{on } \partial\Omega \end{cases} \tag{10}$$

in which $q(x)$ is a distributed transverse load applied onto the beam, $w(x)$ is the beam's bending deflection and $M(x)$ and $V(x)$ are the prescribed values of bending moment and transverse shear force at the ends of the beam.

Using the appropriate boundary conditions and finding the correct expression of $u(x)$, the differential relation in Eq. (9) shall exactly be satisfied at each point of the physical domain. Let instead $\tilde{u}(x)$ be an approximation of $u(x)$. Approximate solution $\tilde{u}(x)$ could be conceived as a linear combination of appropriate "shape functions" $\Psi_i(x)$, with coefficients $a_1, a_2, \ldots, a_n$ to be determined. So, $\tilde{u}(x)$ turns out to be a linear combination of n terms, where the ith term is multiplied by coefficient a_i:

$$\tilde{u}(x) = a_1\Psi_1(x) + a_2\Psi_2(x) + \cdots + a_n\Psi_n(x) = \boldsymbol{\Psi}^{\mathrm{T}}\,\mathbf{a} \tag{11}$$

The effect of such an approximation implies that the left hand side of Eq. (9), evaluated for $\tilde{u}(x)$, may result in nonzero *residuals*

$$\begin{cases} R_D(\tilde{u}, f) = D\tilde{u}(x) - f(x) \neq 0 \\ R_B(\tilde{u}, g) = B\tilde{u}(x) - g(x) \neq 0 \end{cases} \tag{12}$$

When such residuals approach zero, the approximation improves its quality, as $\tilde{u} \simeq u$.

According to the WRM, the best solution satisfies the governing equations in their following weak form:

$$\int_\Omega v_i(x)\, R\, d\Omega = 0 \quad \text{for} \quad i = 1, 2, \ldots, n \tag{13}$$

where each $v_i(x)$ is a *weight* or *test function*, which may be defined as

$$v_i = \frac{\partial \tilde{u}}{\partial a_i} \tag{14}$$

Indeed, in the Bubnov–Galerkin method, commonly known as Galerkin method (see e.g. Bathe [12]), weight functions $v_i(x)$ come to coincide with shape functions $\Psi_i(x)$ in Eq. (11), so that Eq. (13) turns out to be

$$\int_\Omega \Psi_i(x)\, R\, d\Omega = 0 \quad \text{for} \quad i = 1, 2, \ldots, n \tag{15}$$

Otherwise, the Petrov–Galerkin method may be considered, where test functions $v_i(x)$ differ from shape functions $\Psi_i(x)$. Equation (13) means that, over a region of interest, the weighted residual takes a vanishing average value, because the best approximate solution is chosen from trial family $\tilde{u} = \tilde{u}(a_i, x)$.

3.2 FEM Problem Formulation

It is here shown how Eq. (15) could be solved through a standard Galerkin FEM formulation. As e.g. reported by Zienkiewicz and Taylor [165], the FEM is based on a discretization of the physical domain (Ω) into appropriate (finite) subdomains (Ω_e) called *finite elements*.

As Fig. 4 ideally illustrates, the original physical domain is represented by discretized domain $\hat{\Omega}$, defined as the assembly of elements Ω_e

$$\Omega \simeq \hat{\Omega} = \sum_{e=1}^{\mathrm{Ne}} \Omega_e \tag{16}$$

where Ne is the overall number of finite elements in the discretization. Similarly, domain boundary $\partial\Omega$ is approximated as

$$\partial\Omega \simeq \partial\hat{\Omega} = \sum_{e=1}^{\mathrm{Nb}} \partial\Omega_e \tag{17}$$

where Nb is the total number of elemental edges belonging to the boundary.

The quality of the approximation depends on the type of employed finite elements and their overall number. In the limit, it shall be possible to provide an exact representation of a continuum by using an infinite number of finite elements.

Now, suppose to handle the following differential problem, stated in general matrix form

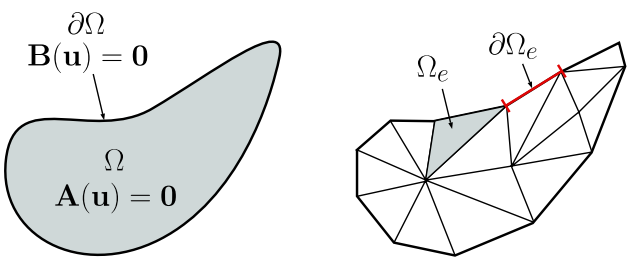


Fig. 4 Discretization of the domain into finite subdomains

$$\mathbf{A}(\mathbf{u}) = \begin{Bmatrix} A_1(\mathbf{u}) \\ A_2(\mathbf{u}) \\ \vdots \\ A_n(\mathbf{u}) \end{Bmatrix} = \mathbf{0} \quad \text{in} \quad \Omega \tag{18}$$

where $\mathbf{u}$ is the sought function, which may be a scalar quantity or a vector of variables, and A_i is a linear or linearized differential operator. The boundary conditions of Eq. (18) may also be stated as

$$\mathbf{B}(\mathbf{u}) = \begin{Bmatrix} B_1(\mathbf{u}) \\ B_2(\mathbf{u}) \\ \vdots \\ B_n(\mathbf{u}) \end{Bmatrix} = \mathbf{0} \quad \text{on} \quad \partial\Omega \tag{19}$$

where B_i is a generic differential operator for the boundary conditions. The FEM representation seeks the approximate solution as

$$\mathbf{u} \simeq \tilde{\mathbf{u}} = \sum_{i=1}^{n} \Psi_i \, a_i = \mathbf{\Psi}^{\mathsf{T}} \, \mathbf{a} \tag{20}$$

where $\mathbf{\Psi}$ is the vector of the *shape functions*, usually locally defined on the elements or subdomains, and $\mathbf{a}$ is the vector of the unknown coefficients. Using the WRM presented in Sect. 3.1 and considering a single element (e), the scalar product between $\mathbf{\Psi}_e$ and Eq. (18) has to vanish at each point of Ω_e. So

$$\begin{aligned} &\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \mathbf{A}(\mathbf{u}) d\Omega_e \\ &= \int_{\Omega_e} \left[\Psi_{e,1} A_1(\mathbf{u}) + \ldots + \Psi_{e,n} A_n(\mathbf{u}) \right] d\Omega_e = 0 \end{aligned} \tag{21}$$

The boundary conditions in Eq. (19) simultaneously need to be satisfied; then, it shall be required that

$$\begin{aligned} &\int_{\partial\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \mathbf{B}(\mathbf{u}) d\partial\Omega_e \\ &= \int_{\partial\Omega_e} \left[\Psi_{e,1} B_1(\mathbf{u}) + \ldots + \Psi_{e,n} B_n(\mathbf{u}) \right] d\partial\Omega_e = 0 \end{aligned} \tag{22}$$

Actually, the previous system of equations is satisfied if the following relation turns out to be fulfilled:

$$\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{A}(\mathbf{u}) d\Omega_e + \int_{\partial\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{B}(\mathbf{u}) \, d\partial\Omega_e = 0 \tag{23}$$

Remembering the previous discretization of the domain stated in Eqs. (16)–(17), Eq. (23) has to be solved on each subdomain; so, it could globally be rewritten as

$$\sum_{e=1}^{\mathrm{Ne}} \left(\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{A}\left(\mathbf{u}_e\right) d\Omega_e + \int_{\partial\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{B}\left(\mathbf{u}_e\right) \partial\Omega_e \right) = 0 \tag{24}$$

where subscript e means that $\mathbf{\Psi}$ and $\mathbf{u}$ are restricted to a single finite element. So, for each finite element, Eq. (20) becomes

$$\mathbf{u}_e \simeq \hat{\mathbf{u}}_e = \sum_{i=1}^{n} \Psi_{i,e} \, a_{i,e} = \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{a}_e \tag{25}$$

Thus, the FEM problem consists of the search for $\tilde{\mathbf{u}}_e$ such that it becomes the solution of Eq. (24), for all $\Psi_{i,e}$ in the space of shape functions V_h defined as

$$V_h = \left\{ \Psi_{i,e} \mid \Psi_{i,e} \in W(\Omega) \right\} \tag{26}$$

where $W(\Omega)$ is a certain Sobolev space such that

$$\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}} \, \mathbf{A}(\mathbf{\Psi}_e) \, d\Omega_e < \infty \tag{27}$$

3.3 On the FEM Application to Moving Load Problems

3.3.1 Problem Formulation

The ML problem could be formulated in the simplest case as follows: a transverse concentrated force, $F(t)$, moves with a constant speed v along a 1D beam without foundation, for instance a simply supported beam, of a finite length L, as earlier shown in Fig. 1.

The following hypotheses are assumed throughout the present formulation:

1. The beam behavior is described by Euler–Bernoulli's theory. It has been deduced within small deformation theory, Hooke's law, Navier's hypothesis and de Saint-Venant's principle;
2. The beam displays a constant cross section and uniform mass per unit length;
3. Only gravitational effects associated with the moving object are considered;
4. The load moves, pointing downward, at a constant speed, e.g. from left to right;
5. The deflection and the bending moment are zero at both ends of the beam (simply supported case).
6. The beam is considered at rest until first force arrival.

The first solution to this problem was likely found by Krylov [99]. Under the above assumptions, such beam ML problem is described by the following 4th-order differential equation of motion:

$$\begin{aligned} &EI \frac{\partial^4 w(x,t)}{\partial x^4} + \mu \frac{\partial^2 w(x,t)}{\partial t^2} + c \frac{\partial w(x,t)}{\partial t} \\ &= -F\delta(x - vt) \end{aligned} \tag{28}$$

where the symbols used in Eq. (28) and throughout the present section display the following meaning:

- x length coordinate, with the origin at the left end of the beam;
- t time coordinate, with the origin at the instant of first force arrival on the beam;
- $w(x, t)$ vertical deflection of the beam at point x and time instant t, measured from the static equilibrium position of the beam subjected to its own weight, acting downward;
- E Young's modulus of the beam;
- I constant area moment of inertia of the beam cross section;
- μ constant mass per unit length of the beam;
- c beam damping coefficient; in the present simplified model, dissipating forces per unit length of the beam are assumed as directly proportional to its transverse velocities;
- F magnitude of the concentrated ML;
- v constant speed of the ML;
- δ Dirac delta function; it expresses the concentrated load as a distribution taking unbounded magnitude at the point of application and zero elsewhere, its resultant being equal to ML intensity F.

Moreover, a beam lying on a (visco)elastic smeared foundation could be considered (Fig. 2). If the balance of the forces acting on an infinitesimal chunk of the beam is carried out, as in Fig. 5, a differential equation with an additional term is found, and it could be used to further explore the ML problem:

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} + \mu\frac{\partial^2 w(x,t)}{\partial t^2} + c\frac{\partial w(x,t)}{\partial t} + k_l w(x,t) = -F\delta(x - vt) \tag{29}$$

In Eq. (29), k_l is introduced as the constant coefficient of a classical linear elastic Winkler foundation (see for instance the discussion, with historical perspective, in Froio and Rizzi [52]). The third term on the left hand sides of Eqs. (28) and (29) represents the contribution of structural damping, coming from either the beam itself and/or the external damping due to an underlying foundation.

Now that the equation of motion of the mechanical system has been defined, it is possible to proceed through the WRM, as earlier introduced in Sect. 3.2.

3.3.2 Application of the WRM

Consider the system earlier shown in Fig. 2, consisting of a finite 1D beam lying on a Winkler (visco)elastic foundation under the action of a transverse concentrated force moving along with a constant velocity.

Suppose to abandon the idea of achieving an exact analytical solution and think about a discretized beam made of finite elements of length h, interconnected at the nodal points. First of all, it is essential to define generic beam element e. It could be seen as a finite piece of a beam, obeying to Euler–Bernoulli beam theory, lying between two nodes (i and j), as depicted in Fig. 6.

This finite element displays constant cross section area A, constant mass density per unit length μ and constant bending stiffness EI along its length. Each node is endowed of two degrees of freedom: the transverse translations (q_1 and q_3) and the rotations about an axis normal to the plane of the sheet (q_2 and q_4). The resulting vector of such four generalized coordinates is defined as follows:

$$\mathbf{q}_e(t) = \left\{q_1(t) \;\; q_2(t) \;\; q_3(t) \;\; q_4(t)\right\}^{\mathsf{T}} \tag{30}$$

Providing four generalized coordinates, the simplest approximation of the field of transverse displacements $w_e(x, t)$ is given in cubic form:

$$w_e(x,t) = a_0(t) + a_1(t)\,x + a_2(t)\,x^2 + a_3(t)\,x^3 = \left\{1 \quad x \quad x^2 \quad x^3\right\}\begin{Bmatrix} a_0(t) \\ a_1(t) \\ a_2(t) \\ a_3(t) \end{Bmatrix} = \mathbf{A}(x)\,\mathbf{a}(t) \tag{31}$$

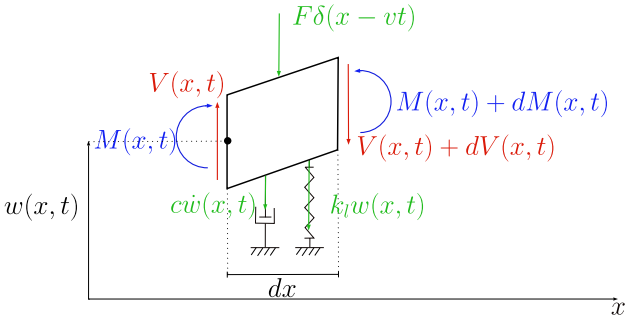


Fig. 5 Acting generalized forces on an infinitesimal chunk of beam

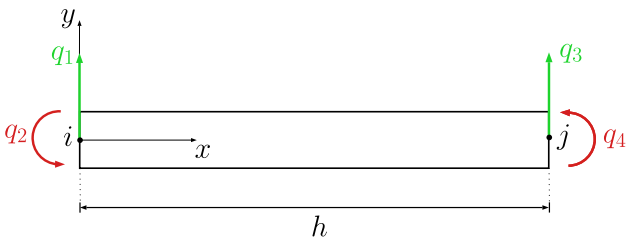


Fig. 6 Euler–Bernoulli beam finite element

In order to find an expression for each coefficient a_i, four boundary conditions have to be imposed for all time instants t:

$$
\begin{aligned}
&q_1(t) = w(0,t) = a_0(t),\\
&q_2(t) = w'(0,t) = a_1(t),\\
&q_3(t) = w(h,t) = a_0(t) + a_1(t)\,h + a_2(t)\,h^2 + a_3(t)\,h^3,\\
&q_4(t) = w'(h,t) = a_1(t) + 2\,a_2(t)\,h + 3\,a_3(t)\,h^2
\end{aligned}
$$

These expressions can be re-written in matrix form as

$$
\begin{Bmatrix} q_1(t)\\ q_2(t)\\ q_3(t)\\ q_4(t) \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 1 & h & h^2 & h^3\\ 0 & 1 & 2\,h & 3\,h^2 \end{bmatrix} \begin{Bmatrix} a_0(t)\\ a_1(t)\\ a_2(t)\\ a_3(t) \end{Bmatrix} \tag{32}
$$

or, in a more compact form:

$$
\mathbf{q}_e(t) = \mathbf{D}\,\mathbf{a}(t) \tag{33}
$$

Using Eq. (33), the expression of the transverse displacements along the beam element, Eq. (31), can be stated in matrix form as

$$
w_e(x,t) = \mathbf{A}(x)\mathbf{D}^{-1}\mathbf{q}_e(t) = \mathbf{\Psi}_e(x)\mathbf{q}_e(t) \tag{34}
$$

where $\mathbf{\Psi}_e(x)$ is the vector of the shape functions. Hence, the displacement field along the beam element can be approximated by the linear combination of these shape functions with the generalized coordinates:

$$
\begin{aligned}
w_e(x,t) =& \begin{Bmatrix} \psi_1(x) & \psi_2(x) & \psi_3(x) & \psi_4(x) \end{Bmatrix} \begin{Bmatrix} q_1(t)\\ q_2(t)\\ q_3(t)\\ q_4(t) \end{Bmatrix}\\
=& \mathbf{\Psi}_e(x)\,\mathbf{q}_e(t)
\end{aligned} \tag{35}
$$

where the interpolating shape functions finally display the following classical cubic expressions:

$$
\begin{aligned}
&\psi_1(x) = 1 - 3\left(\frac{x}{h}\right)^2 + 2\left(\frac{x}{h}\right)^3,\\
&\psi_2(x) = x - 2h\left(\frac{x}{h}\right)^2 + h\left(\frac{x}{h}\right)^3,\\
&\psi_3(x) = 3\left(\frac{x}{h}\right)^2 - 2\left(\frac{x}{h}\right)^3,\\
&\psi_4(x) = -h\left(\frac{x}{h}\right)^2 + h\left(\frac{x}{h}\right)^3
\end{aligned} \tag{36}
$$

Going back to the equation of motion (Eq. (29)), it is now possible to apply the WRM. First of all, that equation has to be transformed into its weak form, as in Eq. (13). Following the Galerkin method, the weight functions to be employed are given by Eq. (36). Then, the scalar product between the test function vector $\mathbf{\Psi}_e(x)$ and Eq. (29) has to be computed on each single element. Additionally, the whole expression has to be integrated over domain Ω_e. Using subscripts x and t for the spatial and temporal derivatives, respectively, and remembering that

$$
M(x,t) = EI\,\frac{\mathrm{d}^2 w_e}{\mathrm{d}x^2} = EI w_{e,xx}(x,t) \tag{37}
$$

Equation (29) becomes

$$
\int_{\Omega_e} \left(\mathbf{\Psi}^{\mathsf{T}} M_{,xx} + \mathbf{\Psi}^{\mathsf{T}}\,\mu\,w_{e,tt} + \mathbf{\Psi}^{\mathsf{T}}\,c\,w_{e,t} + \mathbf{\Psi}^{\mathsf{T}}\,k_l\,w_e + \mathbf{\Psi}^{\mathsf{T}}\,F\delta(x-vt)\right) d\Omega_e = 0 \tag{38}
$$

Analyzing each term in Eq. (38), the equation of motion can be obtained in the following way. The first terms, integrated twice by parts, become

$$
\begin{aligned}
&\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}}(x)\,M_{,xx}(x,t)\,d\Omega\\
&\quad= \left[EI\mathbf{\Psi}_e^{\mathsf{T}}\,w_{e,xxx}\right]_{\partial\Omega} - \left[\mathbf{\Psi}_{e,x}^{\mathsf{T}} EI\,w_{e,xx}\right]_{\partial\Omega}\\
&\qquad+ \int_{\Omega} \mathbf{\Psi}_{e,xx}^{\mathsf{T}} EI w_{e,xx}\,d\Omega\\
&\quad= -\mathbf{Q}_e + \int_{\Omega} \mathbf{\Psi}_{xx}^{\mathsf{T}}(x)\,EI\,\mathbf{\Psi}_{xx}(x) d\Omega\,\mathbf{q}_e(t)
\end{aligned} \tag{39}
$$

where $\mathbf{Q}_e$ is the contribution given by the external forces or by the connection of the element with other parts of the structure. If a single beam element (e) is considered, domain Ω_e is $x = [0, h]$. So, the stiffness matrix is given by

$$
\begin{aligned}
\mathbf{K}_b^{(e)} &= \int_0^h \mathbf{\Psi}_{e,xx}^{\mathsf{T}}(x)\,EI\,\mathbf{\Psi}_{e,xx}(x)dx\\
&= EI\begin{bmatrix} \frac{12}{h^3} & \frac{6}{h^2} & -\frac{12}{h^3} & \frac{6}{h^2}\\ \frac{6}{h^2} & \frac{4}{h} & -\frac{6}{h^2} & \frac{2}{h}\\ -\frac{12}{h^3} & -\frac{6}{h^2} & \frac{12}{h^3} & -\frac{6}{h^2}\\ \frac{6}{h^2} & \frac{2}{h} & -\frac{6}{h^2} & \frac{4}{h} \end{bmatrix}
\end{aligned} \tag{40}
$$

From the second term, the mass matrix of the element is found:

$$
\begin{aligned}
&\int_{\Omega_e} \mathbf{\Psi}_e^{\mathsf{T}}(x)\,\mu\,w_{e,tt}(x,t)\,d\Omega_e\\
&\quad= \int_0^h \mathbf{\Psi}_e^{\mathsf{T}}(x)\,\mu\,\mathbf{\Psi}_e(x)\,dx\,\mathbf{q}_{e,tt}(t) = \mathbf{M}_b^{(e)}\,\mathbf{q}_{e,tt}
\end{aligned} \tag{41}
$$

where $\mathbf{M}_b^{(e)}$ is

$$
\mathbf{M}_b^{(e)} = \frac{\mu}{420}\begin{bmatrix} 156\,h & 22\,h^2 & 54\,h & -13\,h^2\\ 22\,h^2 & 4\,h^3 & 13\,h^2 & -3\,h^3\\ 54\,h & 13\,h^2 & 156\,h & -22\,h^2\\ -13\,h^2 & -3\,h^3 & -22\,h^2 & 4\,h^3 \end{bmatrix} \tag{42}
$$

The third term in Eq. (38) introduces the damping of the system and its contribution will be included into global damping matrix **C**. For each element:

$$\int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}(x)\, c\, w_{e,t}(x,t)\, d\Omega = \int_0^h \boldsymbol{\Psi}_e^{\intercal}(x)\, c\, \boldsymbol{\Psi}_e(x)\, dx\, \mathbf{q}_{e,t} \tag{43}$$

From the fourth term, the foundation stiffness matrix associated to the beam element (e) is found:

$$\begin{aligned} &\int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}(x)\, k_l\, w_e(x,t)\, d\Omega_e \\ &\quad = \int_0^h \boldsymbol{\Psi}_e^{\intercal}(x)\, k_l\, \boldsymbol{\Psi}_e(x)\, dx\, \mathbf{q}_e(t) = \mathbf{K}_f^{(e)}\, \mathbf{q}_e \end{aligned} \tag{44}$$

in which

$$\mathbf{K}_f^{(e)} = k_l \begin{bmatrix} \frac{13\,h}{35} & \frac{11\,h^2}{210} & \frac{9\,h}{70} & -\frac{13\,h^2}{420} \\ \frac{11\,h^2}{210} & \frac{h^3}{105} & \frac{13\,h^2}{420} & -\frac{h^3}{140} \\ \frac{9\,h}{70} & \frac{13\,h^2}{420} & \frac{13\,h}{35} & -\frac{11\,h^2}{210} \\ -\frac{13\,h^2}{420} & -\frac{h^3}{140} & -\frac{11\,h^2}{210} & \frac{h^3}{105} \end{bmatrix} \tag{45}$$

The last term deals with the nodal forces generated by a moving concentrated force placed at $x_c = vt$. Remembering the translation property of the Dirac delta function (sometimes referred to as either the shifting or the sampling property), the contribution of the moving force on a finite beam can be evaluated as:

$$\begin{aligned} \int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}(x)\, F\, \delta(x-vt)\, d\Omega_e &= F \int_0^h \boldsymbol{\Psi}_e^{\intercal}(x)\, \delta(x-vt)\, dx \\ &= F\boldsymbol{\Psi}_e^{\intercal}(vt) = F\boldsymbol{\Psi}_e^{\intercal}(x_c) \end{aligned} \tag{46}$$

The analysis is complete and now the equations of motion can be displayed. Remembering the previous steps, and expressing for convenience structural damping in classical Rayleigh form, instead of in the general form given by Eq. (43), by using a linear combination of the mass and stiffness matrices

$$\mathbf{C} = a_0\, \mathbf{M} + a_1\, \mathbf{K} \tag{47}$$

where a_0 and a_1 are constants, the next equation is valid for each element:

$$\begin{aligned} &\mathbf{M}_b^{(e)}\, \mathbf{q}_{e,tt} + \mathbf{C}^{(e)}\, \mathbf{q}_{e,t} + \left[\mathbf{K}_b^{(e)} + \mathbf{K}_f^{(e)}\right] \mathbf{q}_e \\ &\quad = -F\boldsymbol{\Psi}_e^{\intercal}(x_c) + Q_e \end{aligned} \tag{48}$$

Then, by assembling the contributions from all the finite elements by means of the *direct stiffness method* and imposing the boundary conditions at the two extreme nodes of each finite element, the global equations of motion are obtained in the following form:

$$\mathbf{M}\mathbf{q}_{,tt} + \mathbf{C}\mathbf{q}_{,t} + \mathbf{K}\mathbf{q} = -F\boldsymbol{\Psi}^T(x_c) \tag{49}$$

where **M**, **C** and **K** are, respectively, the global mass, damping and stiffness matrices of the structure; $\mathbf{q}_{tt}$, $\mathbf{q}_t$ and **q** are the global unknown vectors of generalized accelerations, velocities and displacements; $\boldsymbol{\Psi}$ is the vector defined as

$$\boldsymbol{\Psi}^{\intercal} = \left\{0\,0\,0\,0 \,|\, \ldots \,|\boldsymbol{\Psi}_{1,\hat{e}}\, \boldsymbol{\Psi}_{2,\hat{e}}\, \boldsymbol{\Psi}_{3,\hat{e}}\, \boldsymbol{\Psi}_{4,\hat{e}}| \ldots |0\,0\,0\,0\right\} \tag{50}$$

where $\hat{e}$ is the element containing the ML at instant t.

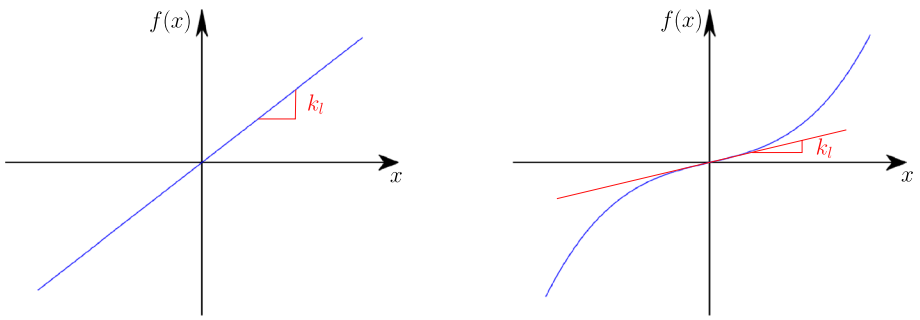
3.3.3 Beams Lying on a Nonlinear Elastic Foundation

Up to now, a linear foundation has been considered, where the elastic restoring force and the displacement are related by a direct proportionality law, as shown in Fig. 7a.

To account for a nonlinear elastic foundation, a cubic term may be added. The law of the foundation describing reaction f_{nl} per unit length (see Fig. 7b), may be taken as in Castro Jorge et al. [19]:

$$f_{nl}(x) = k_l\, w(x) + k_{nl}\, w(x)^3 \tag{51}$$

Recalling the WRM, the nonlinear reaction force exerted by the foundation can be obtained from



(a) *Linear law* **(b)** *Cubic law*

Fig. 7 Different kinds of constitutive law for the stiffness of the foundation

$$\mathbf{R}_{nl} = \int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}\, k_l\, w_e(x,t)\, d\Omega_e + \int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}\, k_{nl}\, w_e^3(x,t)\, d\Omega_e \tag{52}$$

and inserting the FEM approximation given by Eq. (35) it results that

$$\int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}\, k_l\, \boldsymbol{\Psi}_e\, d\Omega_e\, \mathbf{q}_e(t) + \int_{\Omega_e} \boldsymbol{\Psi}_e^{\intercal}\, k_{nl}\, \left(\boldsymbol{\Psi}_e\, \mathbf{q}_e(t)\right)^3 d\Omega_e = \mathbf{K}_l\, \mathbf{q}_e(t) + \mathbf{Q}_{e,nl}[\mathbf{q}_e(t)] \tag{53}$$

where $\mathbf{Q}_{nl,e}[\mathbf{q}(t)]$ is the vector of the nonlinear forces. Subsequently, the global equations of motion are obtained following the same procedure used for the linear foundation, described in Sect. 3.3.2. The final system of nonlinear equations is

$$\mathbf{M}\mathbf{q}_{,tt} + \mathbf{C}\mathbf{q}_{,t} + \mathbf{K}_l\mathbf{q} + \mathbf{Q}_{nl}[\mathbf{q}(t)] = -F\boldsymbol{\Psi}^T(x_c) \tag{54}$$

The numerical integration of Eq. (54) adopted in the present analysis requires the definition of a *tangent stiffness matrix* at each time instant, which will be shown in the next section.

3.4 Numerical Integration

The analytical procedures for solving systems of Ordinary Differential Equations (ODEs), such as Eqs. (49) and (54), are very demanding for transient problems. It is seldomly possible to analytically solve a nonlinear differential equation. That is why, in a FEM implementation, a *direct integration approach* may be adopted to resolve a problem in the time domain. It consists of finding a sequence of solutions at a discrete number of time steps (time discretization); so, the equations of motion are not exactly satisfied at each time instant, but the time evolution of the field quantities is traced, in an approximate form.

Several direct integration methods have been developed. Here, the HHT-α method (Hilber–Hughes–Taylor method) is described, since it is going to be employed in the subsequent ML analysis.

3.4.1 Linear Systems

Consider the equations of motion in Eq. (49). Mathematically, they represent a system of linear second-order differential equations with constant coefficients. An initial-value problem has to be solved, and its solution is $\mathbf{q}(t)$, where $t \in [0, \tau]$, $\tau > 0$. The initial conditions are given as

$$\begin{aligned} \mathbf{q}(0) &= \mathbf{q}_0 \\ \mathbf{q}_{,t}(0) &= \mathbf{v}_0 \end{aligned} \tag{55}$$

Hilber et al. [80] and Hughes [83] provided the following equation to solve the dynamical problem

$$\mathbf{M}\mathbf{a}_{n+1} + (1+\alpha)\mathbf{C}\mathbf{v}_{n+1} - \alpha\mathbf{C}\mathbf{v}_n + (1+\alpha)\mathbf{K}\mathbf{d}_{n+1} - \alpha\mathbf{K}\mathbf{d}_n = \mathbf{F}(t_{n+\alpha}) \tag{56}$$

$$\mathbf{d}_{n+1} = \mathbf{d}_n + \Delta t\mathbf{v}_n + \Delta t^2\left[\left(\frac{1}{2} - \beta\right)\mathbf{a}_n + \beta\mathbf{a}_{n+1}\right] \tag{57}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \Delta t[(1-\gamma)\mathbf{a}_n + \gamma\mathbf{a}_{n+1}] \tag{58}$$

in which the various symbols stand for

n	number of the time step
Δt	size of the time step, which is uniformly taken as τ/N
$t_n, t_{n+\alpha}$	$t_n = n\Delta t$ and $t_{n+1+\alpha} = (1+\alpha)t_{n+1} - \alpha t_n = t_{n+1} + \alpha\Delta t$
$\mathbf{d}_n, \mathbf{v}_n, \mathbf{a}_n$	approximations of $\mathbf{q}, \mathbf{q}_t, \mathbf{q}_{tt}$ at time step t_n
$\mathbf{F}(t_{n+\alpha})$	$-F\boldsymbol{\Psi}^{\intercal}(x_c)$ with $x_c = vt_{n+1+\alpha}$
α, β, γ	constants of the HTT-α method

Coefficients α, β, γ are numerical parameters that govern the stability, accuracy and numerical damping of the algorithm. In order to achieve an unconditionally stable and second-order accurate intregration scheme, these parameters have to be chosen through the following rules:

$$-\frac{1}{3} \le \alpha \le 0 \tag{59a}$$

$$\beta = \frac{1}{4}(1-\alpha)^2 \tag{59b}$$

$$\gamma = \frac{1}{2} - \alpha \tag{59c}$$

Decreasing α increases the amount of numerical dissipation. For $\alpha = 0$ the method reduces to the classical Newmark method, with $\gamma = 1/2$, stating for no numerical dissipation.

Thus, when numerical dissipation is required, the HTT-α method provides a numerical damping, without degrading the accuracy order.

Assume that in Eqs. (56)–(58) vectors marked with subscript n are known from the previous step or given by the initial conditions (Eq. (55)). Otherwise, $\mathbf{d}_{n+1}$, $\mathbf{v}_{n+1}$ and $\mathbf{a}_{n+1}$ are the unknowns that have to be determined at the nth step. An implicit formulation of the HHT-α method is reported in the following. Rearranging Newmark Eqs. (57), (58) for $\mathbf{a}_{n+1}$ and $\mathbf{v}_{n+1}$ gives:

$$\mathbf{a}_{n+1} = \frac{1}{\beta\Delta t^2}\left(\mathbf{d}_{n+1} - \mathbf{d}_n - \Delta t\mathbf{v}_n\right) - \left(\frac{1}{2\beta} - 1\right)\mathbf{a}_n \tag{60}$$

$$\begin{aligned}\mathbf{v}_{n+1} &= \mathbf{v}_n + \Delta t(1-\gamma)\mathbf{a}_n + \frac{\gamma}{\beta\Delta t}\left(\mathbf{d}_{n+1} - \mathbf{d}_n - \Delta t\mathbf{v}_n\right)\\ &\quad - \Delta t\,\gamma\left(\frac{1}{2\beta} - 1\right)\mathbf{a}_n\end{aligned} \tag{61}$$

Substituting these equations into Eq. (56) leads to:

$$\begin{aligned}&\left\{(1+\alpha)\left(\mathbf{K} + \frac{\gamma\mathbf{C}}{\beta\Delta t}\right) + \frac{1}{\beta\Delta t^2}\mathbf{M}\right\}\mathbf{d}_{n+1}\\ &= \left\{\alpha\mathbf{K} + (1+\alpha)\frac{\gamma\mathbf{C}}{\beta\Delta t} + \frac{1}{\beta\Delta t^2}\mathbf{M}\right\}\mathbf{d}_n\\ &+ \left\{\frac{1}{\beta\Delta t}\mathbf{M} + \left[-1 + (1+\alpha)\frac{\gamma}{\beta}\right]\mathbf{C}\right\}\mathbf{v}_n\\ &+ \left\{\left(\frac{1}{2\beta} - 1\right)\mathbf{M} + \Delta t(1+\alpha)\left(\frac{\gamma}{2\beta} - 1\right)\mathbf{C}\right\}\mathbf{a}_n\\ &+ \mathbf{F}(t_{n+1+\alpha})\end{aligned} \tag{62}$$

This can be rewritten in a more compact form as:

$$\tilde{\mathbf{K}}\mathbf{d}_{n+1} = \tilde{\mathbf{F}} \tag{63}$$

where $\tilde{\mathbf{K}}$ is the effective stiffness matrix and $\tilde{\mathbf{F}}$ is the effective load vector within the time integration method.

Algorithm 1 below shows how the HTT-α direct time integration method may be implemented within a FEM code.

Algorithm 1 HTT- α linear solver

if it is the first step **then**
  initialize $\mathbf{q}(0)$ and $\mathbf{q}_{,t}(0)$
  initialize $\mathbf{d}_0$ and $\mathbf{v}_0$
  find $\mathbf{a}_0$ using $\mathbf{d}_0$ and $\mathbf{v}_0$: $\mathbf{a}_0 = \mathbf{M}^{-1}(\mathbf{F}_0 - \mathbf{K}\mathbf{d}_0 - \mathbf{C}\mathbf{v}_0)$
  calculate effective stiffness matrix $\tilde{\mathbf{K}}$
  compute effective force $\tilde{\mathbf{F}}$
  solve linear system $\tilde{\mathbf{K}}\mathbf{d}_1 = \tilde{\mathbf{F}}$
  find $\mathbf{a}_1$ and $\mathbf{v}_1$ using Eqs. (60)-(61)
  update $\mathbf{q}(t_1)$, $\mathbf{q}_{,t}(t_1)$ and $\mathbf{q}_{,tt}(t_1)$ with $\mathbf{d}_1$, $\mathbf{v}_1$, $\mathbf{a}_1$
end if
for each step n **do**
  compute effective force $\tilde{\mathbf{F}}$
  solve linear system $\tilde{\mathbf{K}}\mathbf{d}_{n+1} = \tilde{\mathbf{F}}$
  find $\mathbf{a}_{n+1}$ and $\mathbf{v}_{n+1}$ using Eqs. (60)-(61)
  update $\mathbf{q}(, t_{n+1})$, $\mathbf{q}_{,t}(t_{n+1})$ and $\mathbf{q}_{,tt}(t_{n+1})$ with $\mathbf{d}_{n+1}$, $\mathbf{v}_{n+1}$ and $\mathbf{a}_{n+1}$
end for

3.4.2 Nonlinear Systems

The HTT-α method was adapted by Geradin [66], in order to study the transient response of the solution for a nonlinear system. It is possible to rewrite Eq. (54) in a more general form:

$$\begin{aligned}&\mathbf{M}\mathbf{q}_{,tt}(t) + \mathbf{f}(\mathbf{q}, \mathbf{q}_{,t}) = \mathbf{g}(\mathbf{q}, t)\\ &\mathbf{q}(t_0) = \mathbf{d}_0\\ &\mathbf{q}_{,t}(t_0) = \mathbf{v}_0\end{aligned} \tag{64}$$

where

$\mathbf{f}(\mathbf{q}, \mathbf{q}_{,t})$ states for the internal forces linked to displacements and velocities

$\mathbf{g}(\mathbf{q}, t)$ are the external forces imposed onto the system, being possibly a nonlinear function of $\mathbf{q}$

Similarly to Eqs. (56)–(58), the same finite differences formulas of the Newmark method are used but the equation of motion, discretized in time, becomes:

$$\begin{aligned}&\mathbf{M}\mathbf{a}_{n+1} + \mathbf{C}\left[(1+\alpha)\mathbf{v}_{n+1} - \alpha\mathbf{v}_n\right] \\ &+ \mathbf{K}\left[(1+\alpha)\mathbf{d}_{n+1} - \alpha\mathbf{d}_n\right] \\ &- \alpha\mathbf{Q}_{nl}(\mathbf{d}_n) + (1+\alpha)\mathbf{Q}_{nl}(\mathbf{d}_{n+1}) = \mathbf{F}(t_{n+1+\alpha})\end{aligned} \tag{65}$$

Looking at Eq. (65), at each time step, either $\mathbf{d}_n$ or $\mathbf{d}_{n+1}$ has to be known. A possible technique is to employ the *Newton-Raphson method* that, thanks to an iterative process, allows to find $\mathbf{d}_{n+1}$, despite for the presence of unknown nonlinear term $\mathbf{Q}_{nl}(\mathbf{d}_{n+1})$.

This iterative approach is based on the definition of a residual vector, $\mathbf{r}(\mathbf{d}_{n+1})$, that is:

$$\begin{aligned}\mathbf{r}(\mathbf{d}_{n+1}) =& \mathbf{M}\mathbf{a}_{n+1} + (1+\alpha)\mathbf{C}\mathbf{v}_{\mathbf{n+1}} - \alpha\mathbf{C}\mathbf{v}_n \\ &+ (1+\alpha)\mathbf{K}\mathbf{d}_{n+1} - \alpha\mathbf{K}\mathbf{d}_n \\ &+ (1+\alpha)\mathbf{Q}_{nl}(\mathbf{d}_{n+1}) - \alpha\mathbf{Q}_{nl}(\mathbf{d}_n) - \mathbf{F}(t_{n+1+\alpha})\end{aligned} \tag{66}$$

The aim of the iterations is to find out the value of $\mathbf{d}_{n+1}$ that makes vanishing the residual in Eq. (66). Now, replacing Eqs. (60), (61) into Eq. (66), one obtains that

$$\begin{aligned}\mathbf{r}(\mathbf{d}_{n+1}) =& \left\{(1+\alpha)\left(\mathbf{K} + \frac{\gamma\mathbf{C}}{\beta\Delta t}\right) + \frac{1}{\beta\Delta t^2}\mathbf{M}\right\}\mathbf{d}_{n+1} \\ &+ (1+\alpha)\mathbf{Q}_{nl}(\mathbf{d}_{n+1}) - \alpha\mathbf{Q}_{nl}(\mathbf{d}_n) - \mathbf{F}(t_{n+1+\alpha}) \\ &- \left\{\alpha\mathbf{K} + (1+\alpha)\frac{\gamma\mathbf{C}}{\beta\Delta t} + \frac{1}{\beta\Delta t^2}\mathbf{M}\right\}\mathbf{d}_n \\ &- \left\{\frac{1}{\beta\Delta t}\mathbf{M} + \left[-1 + (1+\alpha)\frac{\gamma}{\beta}\right]\mathbf{C}\right\}\mathbf{v}_n \\ &- \left\{\left(\frac{1}{2\beta} - 1\right)\mathbf{M} + \Delta t(1+\alpha)\left(\frac{\gamma}{2\beta} - 1\right)\mathbf{C}\right\}\mathbf{a}_n\end{aligned} \tag{67}$$

In order to identify the number of the present iteration, apex (i) is employed. So, at each time step, $\mathbf{d}_{n+1}^{(i)}$ should be known, possibly using a minimum number of iterations.

Before starting to iterate, the Jacobian matrix has to be defined as

$$\mathbf{S}(\mathbf{d}_{n+1}^{(i)}) = \left.\frac{\partial\mathbf{r}}{\partial\mathbf{d}}\right|_{\mathbf{d}_{n+1}^{(i)}} \tag{68}$$

If Eq. (67) is substituted into Eq. (68), the expression of the Jacobian matrix for the α-method is obtained

$$\mathbf{S}(\mathbf{d}_{n+1}^{(i)}) = (1+\alpha)\left(\tilde{\mathbf{K}}_{n+1}^{T\,(i)} + \mathbf{K} + \frac{\gamma\mathbf{C}}{\beta\Delta t}\right) + \frac{1}{\beta\Delta t^2}\mathbf{M} \tag{69}$$

$$\tilde{\mathbf{K}}_{n+1}^{T\,(i)} = \left.\frac{\partial\mathbf{Q}_{nl}(\mathbf{d}_{n+1})}{\partial\mathbf{d}}\right|_{\mathbf{d}_{n+1}^{(i)}} \tag{70}$$

The Jacobian matrix in Eq. (69) constitutes the *global tangent stiffness matrix* of the mechanical system. Then, Eq. (71) shows how a converging series of possible solutions can be created

$$\mathbf{d}_{n+1}^{(i+1)} = \mathbf{d}_{n+1}^{(i)} - \mathbf{S}^{-1}(\mathbf{d}_{n+1}^{(i)})\,\mathbf{r}(\mathbf{d}_{n+1}^{(i)}) \tag{71}$$

Equation (71) is derived from a Taylor series expansion of the residual vector, truncated to the first order:

$$\mathbf{r}(\mathbf{d}_{n+1}^{(i+1)}) \approx \mathbf{r}(\mathbf{d}_{n+1}^{(i)}) + \mathbf{S}(\mathbf{d}_{n+1}^{(i)})(\mathbf{d}_{n+1}^{(i+1)} - \mathbf{d}_{n+1}^{(i)}) \tag{72}$$

Finally, an appropriate rule has to be chosen in order to stop the iterative process. For instance, the maximum number of iterations or an appropriate norm of the residual vector could be established. Moreover, an appropriate tolerance (ε) could be set as in the following relation

$$\frac{\left\|\mathbf{d}_{n+1}^{(i+1)} - \mathbf{d}_{n+1}^{(i)}\right\|}{\left\|\mathbf{d}_{n+1}^{(i+1)}\right\|} < \varepsilon \tag{73}$$

The solution will converge to the exact one with a quadratic rate, as stated in Bathe [12]. Chang [21] proves that the solver is unconditionally stable if α, β and γ are chosen as suggested in Eq. (59).

At the end of the section, Algorithm 2 for nonlinear HTT-α time integration, implemented inside the developed FEM code, is listed.

Algorithm 2 HTT-α nonlinear solver

if it is the first step **then**
  initialize $\mathbf{q}(0)$ and $\mathbf{q}_{,t}(0)$
  initialize $\mathbf{d}_0$ and $\mathbf{v}_0$
  find $\mathbf{a}_0$ using $\mathbf{d}_0$ and $\mathbf{v}_0$: $\mathbf{a}_0 = \mathbf{M}^{-1}(\mathbf{F}_0 - \mathbf{Q}_{nl}(\mathbf{d}_0) - \mathbf{K}\mathbf{d}_0 - \mathbf{C}\mathbf{v}_0)$
  calculate effective stiffness matrix $\tilde{\mathbf{K}}$
  compute effective force $\tilde{\mathbf{F}}$
  solve linear system $\tilde{\mathbf{K}}\mathbf{d}_1 = \tilde{\mathbf{F}}$
  find $\mathbf{a}_1$ and $\mathbf{v}_1$ using Eqs. (60)-(61)
  update $\mathbf{q}(t_1)$, $\mathbf{q}_{,t}(t_1)$ and $\mathbf{q}_{,tt}(t_1)$ with $\mathbf{d}_1$, $\mathbf{v}_1$, $\mathbf{a}_1$
  evaluate force vector $\mathbf{Q}_{nl}(\mathbf{d}_1)$
end if
for each step n **do**
  while stopping criteria are not satisfied **do**
    evaluate $\mathbf{Q}_{nl}(\mathbf{d}_{n+1}^{(i)})$
    assemble $\mathbf{S}(\mathbf{d}_{n+1}^{(i)})$
    compute residual with Eq. (67)
    solve Eq. (71) to find $\mathbf{d}_{n+1}^{(i+1)}$
    if the maximum number of iterations is reached **then**
      interrupt the algorithm
    end if
  end while
  save $\mathbf{d}_{n+1}^{(i)}$ in $\mathbf{d}_{n+1}$
  compute $\mathbf{v}_{n+1}$, $\mathbf{a}_{n+1}$ using Eqs. (60)-(61)
  update $\mathbf{q}(t_{n+1})$, $\mathbf{q}_{,t}(t_{n+1})$ and $\mathbf{q}_{,tt}(t_{n+1})$ with $\mathbf{d}_{n+1}$, $\mathbf{v}_{n+1}$ and $\mathbf{a}_{n+1}$
end for

4 Parallel Implementation Approach of the FEM

Several types of computational simulations may require fast computers and High Performance Computing (HPC) resources, to process a vast amounts of data or to perform a large number of calculations within a reasonable time frame. Such necessity arises in many different fields of application such as in solid and fluid mechanics, economic planning, biomedical analysis, robotics, cryptanalysis and so forth. The first approach likely adopted on that by the computer industry was to develop faster and faster electronic components and devices, to attempt delivering the demanded processing speed. Nowadays, it may be getting harder and harder to achieve further performance improvements, because of present physical and technological limits.

In any case, at one present stage of development of the hardware components, the computational time may significantly be reduced by introducing different implementation strategies such as the one put forward by *parallelism*. Parallelism consists of simultaneously performing several operations, instead of only sequentially processing the data. Thus, parallelism may be similar to a vision of an organized society in which several people of comparable skills are cooperating, in completing a complex job, in a fraction of the total time that would be taken by just a single individual. The concurrency of the tasks is not meant to speed up the execution of each individual task, but to increase the global byproduct of the whole system.

FEM applications frequently require a fine, often multi-dimensional, discretization, resulting into several dofs and associated unknowns. The efficient solution of such large-scale FEM problems may be accomplished on parallel machines (see e.g. Yagawa et al. [160], Dupros et al. [41], Ullah et al. [154]).

This section aims at framing the exploitation of the parallelism concept into FEM calculations, within the present Moving Load context. Section 4.1 presents three different approaches to achieve efficient parallel FEM implementations. Different models of computation are explained in Sect. 4.2, as a guideline to understand which kind of parallel algorithm may result to be more suitable for an available

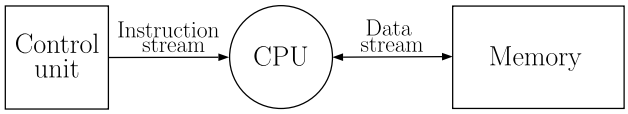


Fig. 8 SISD computer model (adapted from Selim [140])

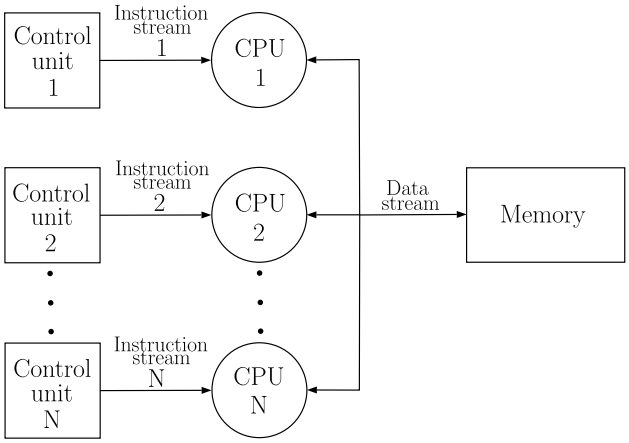


Fig. 9 MISD computer model (adapted from Selim [140])

computing architecture. Finally, Sect. 4.3 provides an algorithm developed within the present attempt, for the analysis of ML structural dynamics problems, but it could also be employed in a generic FEM analysis carried out on any parallel distributed computing system.

4.1 Different Strategies for FEM Implementation

From a mere computational viewpoint, a general FEM analysis shall be composed of the following main phases:

1. *Input phase*: problem characteristics, element and nodal data, boundary conditions and geometry information are provided to set up an appropriate model description of a physical problem;
2. *Evaluation phase*: element characteristics are evaluated from the information collected in phase 1;
3. *Assembly phase*: element contributions are assembled, to create the global matrices and vectors that constitute the solving discretized system of governing equations;
4. *Solving phase*: nodal unknowns of the problem at hand are obtained by solving the system built in phase 3;
5. *Post-processing phase*: calculated data are post-processed, in order to display the various outcomes sought for the analysis, toward interpretation and critical evaluation, and possible reiteration, by refinement or adaptation of the model (model updating) and identification of its underlying parameters.

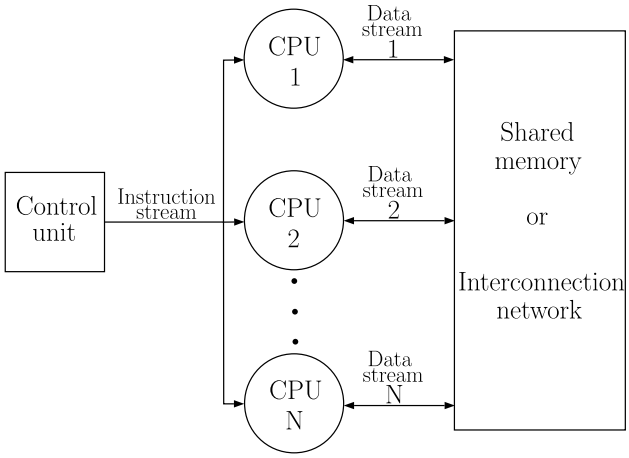


Fig. 10 SIMD computer model (adapted from Selim [140])

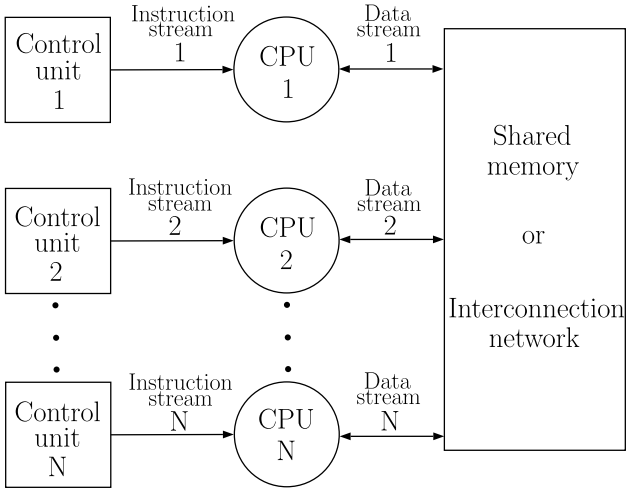


Fig. 11 MIMD computer model (adapted from Selim [140])

It could be seen that the input phase can be parallelized, as well as the evaluation of the element characteristics (e.g. stiffness matrix and load vector), since each element brings in its own information, independently from the other parts of the discretization. Moreover, the most computationally expensive phases are constituted by the assembly and solution of the resulting system of discretized equations; so, parallel solvers may drastically reduce the time required to evaluate the unknowns of the physical problem.

As described by Noor [119], a number of strategies, based on a "divide and conquer" approach, could be used to increase the degree of parallelism within a given FEM implementation. They consist of splitting a large and complex problem into a number of smaller and simpler subproblems, which may independently be solved. Towards that, three main *parallelization strategies* are presented in the following:

1. *Domain decomposition and sub-structuring*: the fundamental idea of this approach is to subdivide the physical domain into a number of non-overlapping regions. Thus, a new initial-boundary value problem is stated, for every subdomain. Since the nodal values on the boundaries of each region are unknown, the solution could iteratively be found, to satisfy the governing equations, over the whole domain.
2. *Operator splitting*: this technique achieves the solution of a complex problem from that of a sequence of simpler problems. It may be employed in partitioning the computational task into a number of subtasks that are either independent or loosely coupled, as explained below. Thus, computations could be carried out on distinct processes, with little need of communication among them, which may reduce the speed of computation. For instance, an operator splitting may be employed, to deal with multidimensional problems, by converting them into a sequence of one-dimensional problems.
3. *Element-by-element solution*: when the scale of the problem becomes very large, the requirement of memory necessary to store and generate the global matrices and vectors greatly intensifies. The basic idea of an element-by-element approach is to perform a calculation at the element level, thus avoiding to create global matrices and vectors.

4.2 Models of Computation

Before implementing a parallel code, it is worthwhile to design algorithms toward achieving a good understanding of performance and efficiency. Such design phase should be focused on the available resources and facilities, which could be described through different models of computation, as suggested by Selim [140].

Any computer, either sequential or parallel, works by executing instructions on data. An algorithm, thanks to a stream of instructions, tells the machine what to do at each step. The instructions dictate to the Central Processing Unit (CPU) how to process the input of the algorithm, which could be seen as a stream of data. Depending on the number and arrangement of these streams, computers may be distinguished among four main classes:

1. *Single Instruction stream, Single Data stream* (**SISD**): a SISD computer consists of a single CPU receiving a single stream of instructions that operates on a single stream of data, as shown in Fig. 8. The control unit sends an instruction to the CPU, which executes it on a data obtained from the memory. An algorithm designed for a SISD computer is defined as *sequential* or *serial*.
2. *Multiple Instruction stream, Single Data stream* (**MISD**): a MISD computer consists of a certain number of CPUs, each with its own control unit. A common shared memory provides a unique datum on which the CPUs have to simultaneously work, each performing different operations, as displayed in Fig. 9.
3. *Single Instruction stream, Multiple Data stream* (**SIMD**): in this class of calculators, as depicted in Fig. 10, a parallel computer consists of N identical CPUs, all controlled by a central control unit.

   At each step of the algorithm, all CPUs synchronously execute the same instruction on different data. The memory providing data to the processors could be organized in two different ways. The first option is that the CPUs shall share a common memory, which they could simultaneously access. Multiple-read access to the same memory address should in principle endow no problem, because each processor makes a local copy of the content. Instead, multiple writing requires a deterministic rule to achieve memory access, in order to avoid conflicts. The second solution is to divide the memory into blocks in which only a processor could be accessed. A data could be employed by an external process only using an explicit request of communication.
4. *Multiple Instruction stream, Multiple Data stream* (**MIMD**): this class of computers turns out to be the most general and powerful, among the parallel architectures. As sketched in Fig. 11, this parallel system is made up of N CPUs ruled by N streams of instructions, sent by their dedicated control units, in order to elaborate N streams of data. As for SIMD, the communication between processors is performed through a shared memory or an interconnecting network. In the first case, processors are linked to a globally available memory, while the second alternative requires that each processor displays its own individual memory location. It is possible to refer to the latter as multicomputers or *distributed systems* because of their physical distance separating the processors. The main difference between SIMD and MIMD computers is the presence of N control units, instead of a shared one. This allows for MIMD computers to asynchronously execute different programs on different data.

Table 1 Minimal set of MPI routines [69]

MPI_Init	Initializes MPI
MPI_Finalize	Terminates MPI
MPI_Comm_size	Determines the number of processes
MPI_Comm_rank	Determines the label of the calling process
MPI_Send	Sends a message
MPI_Recv	Receives a message

The MIMD model is employed to build *clusters*. Clusters are sets of machines (also called *nodes*) working together as a unified resource and connected over a high-bandwidth network. These computational architectures are based on the absolute and incremental scalability of the system. This means that clusters could be made up with a very large number of nodes and they could be expanded, if necessary. Furthermore, the ratio between price and performance becomes favorable, because great results may be achieved even if cheap commodity nodes (i.e. workstations) would be employed. Moreover, if each node were equipped with a multiprocessor, a great amount of computational power could be made available.

It is crucial to distinguish between the notion of *processor* and that of *process*, to clarify terms that will be used in the following. A process is a computational task and a collection of them builds up an algorithm. Many processes may simultaneously be executed on a number of available processors. A parallel algorithm starts its execution on an arbitrarily chosen processor, which is responsible to create other computational processes and to distribute them between processors. A processor is said to be free when it is waiting for a process. So, if all processors are busy, a process is queued and waits for a free processor.

In order to exploit the performance of a cluster, an efficient mean of communication between nodes has to be developed. Processes owned by different nodes could exchange data and perform synchronizing actions using messages; hence, the denomination *message passing*. In the most general form, message-passing paradigms are able to coordinate the execution of several programs distributed between nodes. Different Application Programming Interfaces (APIs) (e.g. Message Passing Interface (MPI), Parallel Virtual Machine (PVM)) support four basic operations. The interactions between nodes are accomplished by sending and receiving messages calling `send` and `receive` functions. Additionally, the send and receive operations must specify a target address, so, a unique identification has to be assigned to every process, through a function such as `whoami`. Typically, the number of processes participating to the ensemble is provided by `numprocs`, in order to carry out collective computations, which are operations involving all the available processors.

Among the *protocols* allowing for communication between nodes, MPI has become widely used for its:

- *Universality*: the message-passing model fits well on separate processors connected by a fast or slow communication network. Thus, it matches the hardware of today parallel supercomputers, as well as the workstation networks and the dedicated PC clusters that are beginning to compete with them. Furthermore, MPI runs either on shared or distributed memory architectures;
- *Portability*: no need to modify the source code, when an application has to be ported to a different platform, and a variety of implementations may be available (i.e. Open MPI [115], MPICH [70]);
- *Functionality*: many built-in routines are available and collective calls are implemented, yet to simplify communication and synchronization between machines;
- *Scalability*: the protocol allows to obtain a speed up, by simply increasing the number of processes that execute a program.

As stated by Grama et al. [69], the MPI library contains over 125 routines, but it is possible to write fully functional message-passing programs by using only six routines, as gathered in Table 1.

These routines are able to initialize and terminate the use of a MPI library, to get information about the parallel computing environment and to send and receive messages. So, using a few routines, it is possible to achieve a great performance boost. The interested reader may deepen some other aspects on MPI architecture, for instance, in Snir and Gropp [141] and Gropp et al. [71].

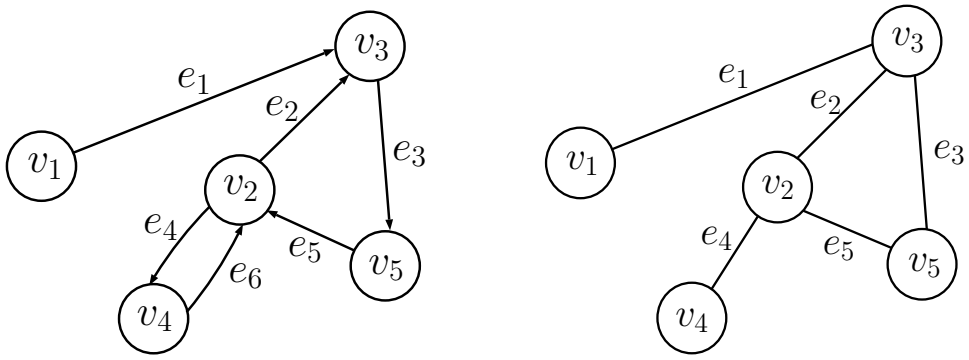


Fig. 12 Two examples of a graph

(a) *Directed Graph* **(b)** *Undirected Graph*

4.3 A Parallel FEM Algorithm for Distributed Systems

In this section, a general parallel algorithm employed to implement the FEM, for the targeted ML dynamical problem context described in Sect. 3.3.2, by adopting an operator splitting approach on distributed systems (MIMD), is presented.

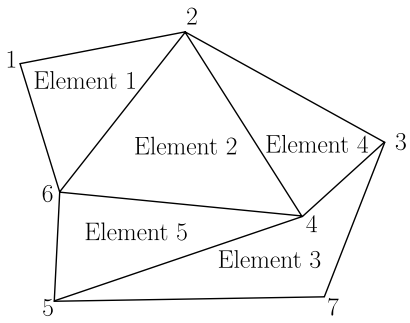
The general rule that should be followed to build a well-performing parallel code is to split a job into smaller tasks, and to distribute them between the available processes. This does not mean that a process shall run only a little part of the code, but that simultaneous processes concurrently work together to achieve the same aim.

The most important task of a FEM code is to efficiently assemble the global matrices and vectors, and then to solve the obtained system of equations using optimized parallel solvers. It is for this reason that dedicated libraries, such as, for instance, PaStiX [78] and PETSc [10], have been built.

During the assembly phase, some local matrices have to be computed from each finite element and their components have to be inserted at specific positions of the global matrices, according to the numbering of the degrees of freedom. By using a sequential code, element matrices are evaluated and collocated, one by one.

On the contrary, a parallel program should be able to split the mesh information between the various processes; so, the information related to the geometry of the problem could simultaneously be read and processed. For instance, if a problem is discretized by ten finite elements and one second would be spent to read each of them, the elapsed time would be ten seconds, for a serial code, while two seconds should instead be needed by a parallel program executed through five concurrent processes.

An essential preprocessing operation is to consider a FEM mesh and to create a partition of it, with the purpose of teaching the processes which elements they have to take care of. Since the finite elements are connected to each other at the nodes of the mesh, it results unavoidable to assign adjacent finite elements to different processes. Consequently,



Graph of the mesh

1	1, 2, 6
2	2, 4, 6
3	3, 4, 5, 7
4	2, 3, 4
5	4, 5, 6

Fig. 13 The resulting graph from a mesh

for handling such "process interfaces" during a parallel FEM solution procedure, communications between the processes are needed.

The problem of partitioning elements into roughly equal sets, under the condition of minimizing the communication, may be identified as the well-known graph-partitioning problem. For instance, libraries such as METIS [92] are able to elaborate a mesh file and decide how to split it, based on *graph theory*.

In the following, some basic concepts of graph theory are presented, in order to obtain a partitioned mesh.

4.3.1 Creation of Partition

As explained by Koijo [93], in mathematics a graph shall formally be defined as a pair of sets (V, E), where V is the set of *vertices* that represent the extremities of the *edges*, which are the elements belonging to E. An edge could also be seen as the line linking two vertices. In order to briefly illustrate what a graph is, Fig. 12 may be considered.

In such drawn graphs, $V = \{v_1, v_2, v_3, v_4, v_5\}$ and $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ are, respectively, the sets of vertices and edges. An edge can also be written as $e_1 = (v_1, v_3)$. In general, two types of graph exist. The first one is the category of *directed* graphs. Their edges display an orientation so that (v_1, v_3) is different from (v_3, v_1). In this case, edges display an arrow specifying their direction, as show in Fig. 12a. On the other hand, as depicted in Fig. 12b, the second class is represented by an *undirected* graph, for which the order of the vertex of the pair is not influent.

Weights could be associated to both kinds of graph. Weights are numerical values associated to each edge or node, with a different meaning depending on the studied case. For instance, if vertices represent cities, edges may stand for the existence of connecting roads. So, by using weights on edges, it is possible to specify the length of each road.

An undirected graph of a FEM mesh may be built, starting from its connectivity information and may be used for the creation of mesh partitions. This means that it has to be known which nodes belong to each element. For instance, two node numbers have to be provided to define a beam element, three for a triangular element and so forth. The graph may be represented as a cell structure in which the index of each cell states for the element number and each cell is filled with the numbers of the nodes owned by the element. An example may be visualized in Fig. 13.

The next step is to set weights, in order to ensure that the computed graph partition satisfies some specified constraints. A proportional weight could be assigned, depending on the local communication volume and the amount of computations performed by each task. In fact, if two edges straddle partitions, data exchanges are required. As a matter

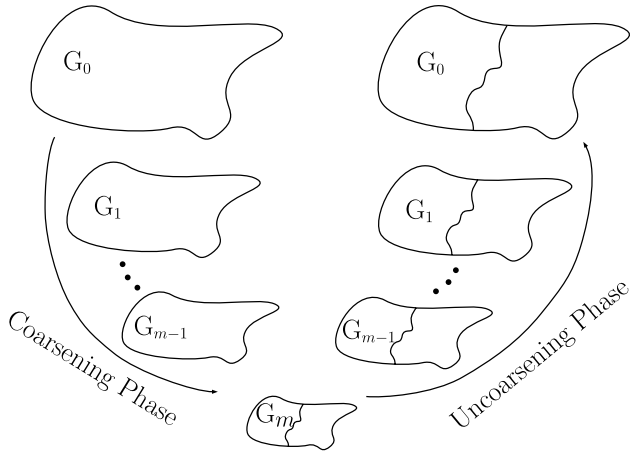
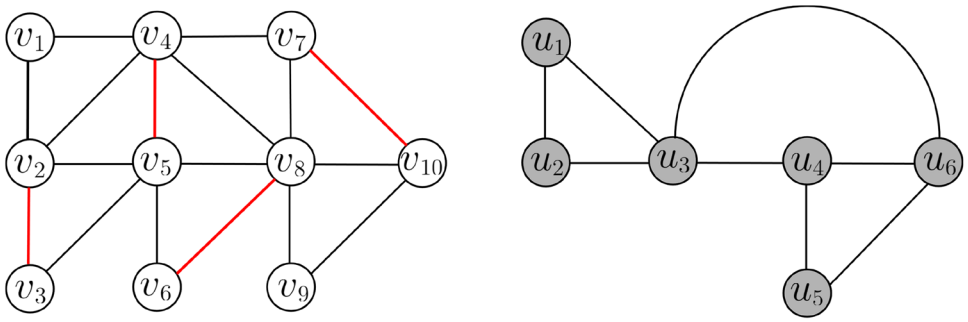


Fig. 14 The three steps of graph partitioning: coarsening, bisectioning and uncoarsening phase (adapted from Karypis and Kumar [92])

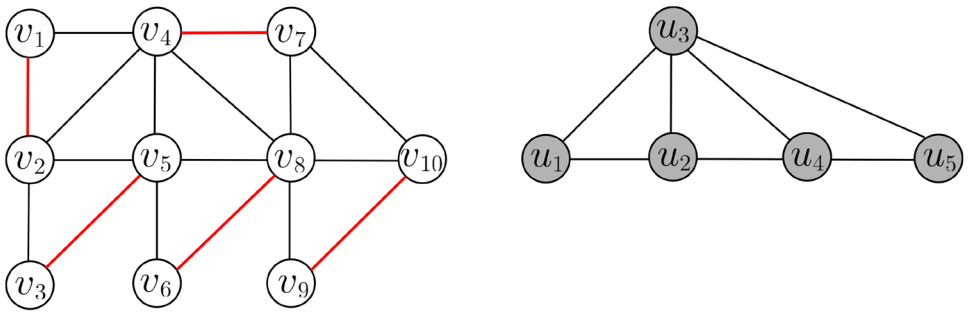
of fact, physical quantities associated to nodes and degrees of freedom should be shared between partitions, when elements belonging to different partitions share, at least, a node. Thus, requiring the same data in more than one process, some kind of communication is demanded between different nodes, in order to guarantee that information is locally stored. Therefore, communication is a time-consuming part of the program, due to the low performance of the means that allow communication between processes. Nowadays, it is recommended to use less messages as possible, to achieve the desired speedup. This is the reason bringing the need of attaching importance to partitioning.

As explained by Karypis and Kumar [92], the graph partitioning problem, known also as *k-way*, could be formulated as follows:



$$M_1 = \{[v_2, v_3]\,[v_4, v_5]\,[v_6, v_8]\,[v_7, v_{10}]\}$$

(a) *Coarse graph using a generic matching*



$$M_2 = \{[v_1, v_2]\,[v_3, v_5]\,[v_4, v_7]\,[v_6, v_8]\,[v_9, v_{10}]\}$$

(b) *Coarse graph using the maximum matching*

Fig. 15 Two examples of a coarse graph. The red line indicates matching edges, white circles simple vertices and grey circles multinodes. M_1 and M_2 are two matchings of the graph. It is shown that the coarse graph in **b** is smaller than the graph in **a**, because M_2 is the maximum matching (adapted from Papadimitriou and Steiglitz [124]). (. (Color figure online)Color figure online)

k-way graph partitioning problem

Given a graph $G = (V, E)$ with $|V| = n$ the number of vertices, the k-way problem consists of partitioning V into k subsets, $V_1, V_2, \ldots, V_k$ such that $V_i \cap V_j = \emptyset$, $\forall i \neq j$, $|V_i| = n/k$ and $\bigcup_i V_i = V$, and the number of edges of E whose incident vertices belong to different subsets is minimized (minimization of the *edge-cut*).

For those graphs that display weights associated to the vertices and edges, the partition goal is to find k disjoint subsets such that the sum of the vertex-weights in each subset is the same, and the sum of the edge-weights whose incident vertices belong to different subsets is minimized. The final result of a *k-way* partition of V is commonly depicted as a vector P of length n, such that for every vertex $v \in V$, $P[v]$ is an integer between 1 and k, indicating the partition at which vertex v belongs.

The *k-way* graph partitioning problem is classified as NP-complete, where NP stands for the Nondeterministic Polynomial time required to verify the correctness of a given solution. This means that no efficient algorithm has been found yet for this class of problems, but nobody has proven that NP-complete problems are unsolvable (see Garey and Johnson [63], to deepen the argument). However, many algorithms have been developed to find reasonably good graph partitions. Three different kinds of partitioning algorithms could be found in the literature (see Karypis and Kumar [92]):

1. *Spectral methods*: they rely on the computation of the eigenvalues of the Laplacian matrix related to a graph. Then, the eigenvectors associated to the sought eigenvalues may be used to compute good separators in graphs. These methods, above all the multilevel spectral bisection methods, provide good partitions for a wide class of problems, but they turn out to be very computationally expensive, requiring a large amount of time. Pothen et al. [127] and Hendrickson and Leland [76] presented two different ways of interpreting the spectral method and they could also be adapted as introductory references on the topic.
2. *Geometric methods*: these approaches use the geometric information of the graph to find a good partition, and for this reason, they are applicable only if coordinates are available for the vertices of the graph. Due to the randomized nature of this class of algorithms, multiple trials are often required, and the obtained solution may display a lower quality, as compared to that of spectral partitions. One of the most prominent algorithms has been described by Miller et al. [113].
3. *Multilevel methods*: the algorithms belonging to this class reduce the size of the graph by collapsing vertices and edges, partition a smaller graph, and then uncoarsen it to construct partitions for the original graph. A multilevel algorithm has been developed by Hendrickson and Leland [77].

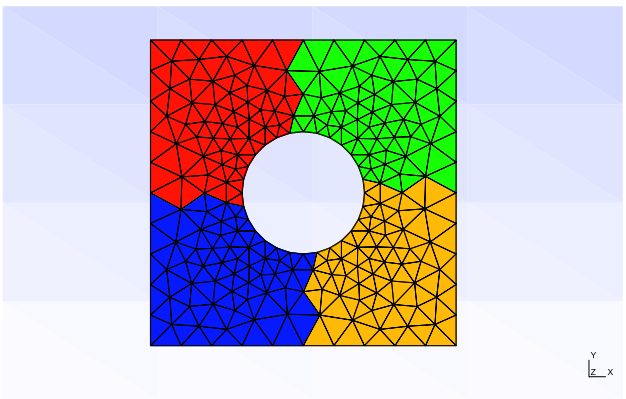


Fig. 16 Example of a partitioned mesh: four different colors are employed to distinguish the partitions. The triangular mesh has been obtained through GMSH and the partitions have been generated by employing METIS

In order to figure out the graph partitioning problem, the *multilevel graph* algorithm by Karypis and Kumar [92] is exposed. The knowledge of this method is deepened because it has been employed during the development of the FEM code presented in the next section. The basic concept of this algorithm is to coarsen a graph G_0 down to a few hundred of vertices, to partition the new smaller graph and then to project it back toward the original finer graph, as displayed in Fig. 14. During the coarsening phase, a sequence of smaller graphs G_i is constructed. Each of them displays a decreasing number of vertices, such that

$$G_0, G_1, G_2, \ldots, G_m \rightarrow |V_0| > |V_1| > |V_2| > \cdots > |V_m| \tag{74}$$

A set of adjacent vertices of G_i is combined to form a single vertex, called *multinode* of coarser graph G_{i+1}. The edge between two vertices is collapsed using the *matching* procedure. A matching of a graph is said to be a set of edges, no two of which share a vertex. Thus, coarser graph G_{i+1}

is built by finding a matching M of G_i and collapsing the vertices belonging to M into multinodes. The unmatched vertices are simply copied over to G_{i+1}. The size of a coarse graph is minimized when a *maximal matching* is found. It means that any edge in the graph that is not matching M has at least one of its endpoints matched. Some algorithms to discover maximal matchings are presented by Papadimitriou and Steiglitz [124]. An example of coarsening may be found in Fig. 15.

The second phase of a multilevel algorithm computes a partition P_m of coarsest graph $G_m = (V_m, E_m)$ by splitting V_m into k parts, each containing $1/k$ of vertices belonging to original graph G_0. This could be achieved by employing 2-way algorithms, such as the spectral bisection, the geometric bisection and combinatorial methods, which are able to split V_m into two partitions. Instead, if k partitions are desired, the k-way problem could be solved by recursively applying the 2-way algorithms on each partition already created through the bisection approach.*The final result is that the amount of time spent to split coarse graph G_m turns out to be much smaller than the time spent for the original graph.*

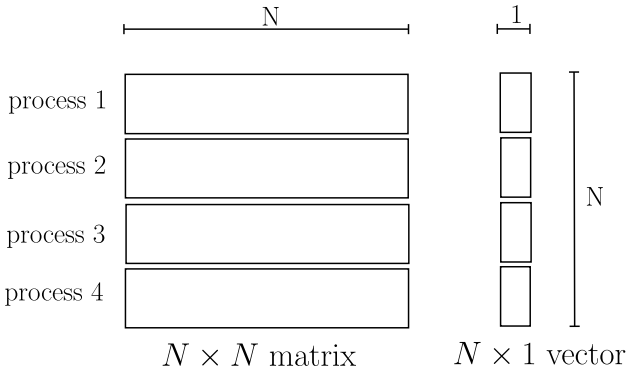


Fig. 17 CRS storage format for distributed sparse matrices and vectors adopted by libraries such as PETSc, toward parallel computing

To sum up the partitioning procedure, Algorithm 3 is presented.

When the partitioning phase ends, it is known which elements of the mesh are owned by a process, thanks to vector P. A possible intuitive result can be seen in Fig. 16, in which four different colors discriminate the element ownership. Notice that the four partitions are not divided along the two diagonals of the external square, because that configuration would lead to a larger number of shared nodes among the partitions.

Algorithm 3 Mesh (graph) partitioning

```
for each element do
    read its number
    read its connectivity
    update graph with the numbers of owned nodes
end for
find maximum matching of graph
coarsen graph
split graph using a k-way algorithm
uncoarsen graph
```

The last step is to uncoarsen graph G_m by projecting it back to original one G_0, through a series of graphs $G_{m-1}, G_{m-2}, \ldots, G_1$, as shown in Fig. 14.

Partition P_i could be obtained from P_{i+1}, by assigning vertices v collapsed in a multinode $u \in G_{i+1}$ to partition $P_{i+1}[u]$. So, $P_i[v] = P_{i+1}[u]$ for all v in multinode u. A smaller edge-cut could be obtained by implementing a heuristic refinement algorithm.

4.3.2 Reading the Input Files

Ideally, in a parallel FEM code, it may be supposed to dispose of three main input files: one containing the information about the element partitioning, one gathering the physical parameters of the simulation, and one including the nodal coordinates and the connectivities of the various elements.

$$\begin{bmatrix} 11 & 12 & 13 & 14 & 0 & 0 \\ 0 & 22 & 23 & 0 & 0 & 0 \\ 0 & 0 & 33 & 34 & 35 & 36 \\ 0 & 0 & 0 & 44 & 45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 56 \\ 0 & 0 & 0 & 0 & 0 & 66 \end{bmatrix}$$

$$\begin{aligned} \texttt{values} &= \begin{bmatrix} 11 & 12 & 13 & 14 & 22 & 23 & 33 & 34 & 35 & 36 & 44 & 45 & 56 & 66 \end{bmatrix} \\ \texttt{row\_indx} &= \begin{bmatrix} 1 & & & & 5 & & 7 & & & & 11 & 13 & 14 & 15 \end{bmatrix} \\ \texttt{col\_indx} &= \begin{bmatrix} 1 & 2 & 3 & 4 & 2 & 3 & 3 & 4 & 5 & 6 & 4 & 5 & 6 & 6 \end{bmatrix} \end{aligned}$$

Fig. 18 Compressed row storage format example for a sparse matrix (symbolic representation)

First of all, a data structure has to be built, in order to contain information about the type, the mechanical properties and the number of nodes that characterize each element. The coordinates and the connectivity of each node have to be provided, to fulfill the description of each element. This information may be communicated to each process in two different ways. A unique copy of the same input files may be available for each process. Those files contain the information of the whole system; so, a process may spend time to read unnecessary data that have to be rejected. Otherwise, a specific file should be written, storing only the parts of the mesh information that are demanded by the process during the FEM simulation.

Dealing with the unknowns of the problem, it is fundamental to allocate the correct amount of local memory to store the nodal values that, in typical displacement-based formulations in solid mechanics, are nodal displacements (and velocities and accelerations). Additionally, each node has a specified number of degrees of freedom, depending on the element type. Each degree has to be labeled with a *unique* number (see the procedure described in Sect. 4.3.3), so that the unknowns of the problem could be defined and enumerated.

Finally, the boundary conditions have to be set, to complete this phase. Some degrees of freedom are marked by the boundary conditions to be tagged as constrained (e.g. with -1, instead as with a positive number). Those degrees of freedom are kept out from the search of the nodal unknowns, because their value is imposed from the user, at the initial definition of the mathematical problem.

Algorithm 4 describes how input files could be read in parallel.

Algorithm 4 Reading the input files

```
for each process do
    read partitioning file
    save numbers of elements owned by process
    open mesh file
    if process is reading element connectivity then
        look at element number
        if it is owned by process then
            save connectivity information
        end if
    else if process is reading node coordinates then
        if node is owned by a locally stored element then
            save data
        end if
    end if
    open file with information about physical problem
    for each owned element do
        save element type
        allocate memory to store degrees of freedom
        save mechanical properties
    end for
    for each owned node do
        allocate memory for nodal values
        set boundary conditions
    end for
end for
```

4.3.3 Allocating Memory

The exact amount of memory to store global matrices and vectors should be allocated, in order to obtain a good performance from a FEM code, without wasting computational resources. So, the number of the unknowns of the physical problem needs to be determined.

Seeking the global dimension of the problem, every degree of freedom must be enumerated. Some troubles may arise in their numbering, since some nodes shall be shared among partitions. It has to be remembered that every machine can only read information saved on its local memory; so, no data is shared without communication in distributed systems. Actually, a degree of freedom must display the same number in all the processes, even though it is shared between them. Differently from a serial code in which nodes are numbered one by one, another strategy must be sought, for a parallel algorithm but first the matrix and vector storage format has to be introduced. The matrices and vectors could be stored in parallel, by employing the operator splitting approach, so that each process only owns a part of these structures.

Suppose to have four processes, so global matrices and vectors may be split into four different parts, each of them locally stored by a process, as displayed in Fig. 17. Every row or column index matches with the number of a degree of freedom; so, if the dimension of a matrix were $N = 80$, the first process would own the rows and degrees of freedom from 1 to 20, the second from 21 to 40, and so forth.

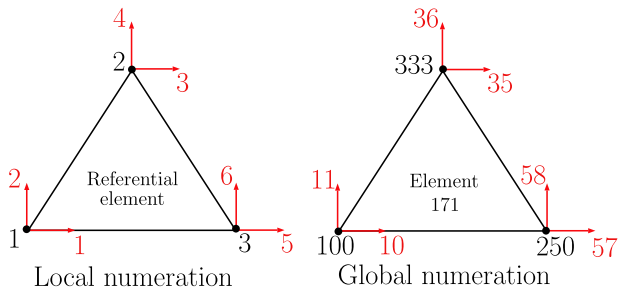
This means that the ownership of the degrees of freedom is set among processes, also for those at the interface between cells. Conventionally, if a node is shared between partitions, its degrees of freedom may be assigned to the processor with the smaller index; so, communication is required to acquire the global numeration on all the other processes. Algorithm 5 describes a possible implementation based on that suggested by Heister et al. [74].

These steps turn out fundamental, to find out the global dimension of the problem and the nonzero patterns. When the enumeration of the degrees of freedom is terminated, size N of the vector of unknowns represents the characteristic dimension of the global matrices. Then, the nonzero pattern is sought, because, usually, matrices related to a FEM analysis are sparse; so, most of their entries are zeros.

The first approach is to store sparse matrices using *triplets*, which are sets of three numbers representing the nonzero entries by the row and the column indices and a numerical value. So, if nonzeros are `nnz`, `nnz`^{3} values have to be stored, for each global matrix.

The Compressed Row Storage (CRS) format could be adopted, decreasing the amount of required memory. The CRS format saves in memory only 2 `nnz` $+ N + 1$ numbers. This storing format puts the subsequent nonzeros of the matrix rows in contiguous memory locations and save their indices using two vectors, as below explained and depicted in Fig. 18:

1. `values`: it is a vector used for floating-point numbers that represent the values of the nonzero entries of the matrix;
2. `col_indx`: it is a vector which stores the column indices of the nonzeros;
3. `row_indx`: each element of this vector represents the position in `values` of the first nonzero entry belonging to each line of the matrix. Conventionally, `row_indx[N+1]` is set equal to `nnz`.



Node numeration

Local	Global
1	100
2	333
3	250

Dof numeration

Local	Global
1	10
2	11
3	35
4	36
5	57
6	58

Fig. 19 Mapping from local to global numeration

Algorithm 5 Global enumeration

```
Define p and q as indeces of two different processes
for each process p do
    loop over locally owned elements
        for each degree of freedom do
            if it is not constrained then
                mark it as invalid (i.e. with −2)
                if a node is shared between processes p and q then
                    if p < q, ∀q then
                        mark degree of freedom as valid again (i.e. with 0)
                    end if
                end if
            end if
        end for
    end loop
    for each valid degree of freedom do
        give it a number starting from 0
    end for
    find maximum local values n_p among degrees
    send it to other processes
    shift local indices by ∑_{q=0}^{p−1} n_q
    if some degrees are still marked as invalid then
        search their numbers in processes marked with q < p
    end if
end for
```

In order to allocate the right amount of memory, for a CRS matrix, it is enough to know how many nonzeros are present in each row. So, it is necessary to correctly resize the `values` vector, because deleting or adding single entries can take an amount of time proportional to the number of nonzeros. In fact, since no gaps are existing between two nonzero entries, for instance, it is required to shift up by one the position all the stored numbers of `values`, when a new nonzero is inserted in the first cell of the vector.

Finally, the pattern may be expressed by employing the indices of the nonzero entries. These indices could be found by performing a fake assembly of the matrices, without actually inserting values in them. When it is not possible to discover the exact number of the nonzeros, remember that extra space is cheaper than an inadequate amount of memory. So, it is better to allocate extra resources, which should be able to avoid the above-mentioned unnecessary copy and paste operations.

4.3.4 Assembling Global Matrices and Vectors

Once the necessary amount of memory is allocated, the global matrices and vectors could be initialized. For each element, the stiffness, damping and mass matrices of a dynamical problem have to be built, together with the nodal force vector. Some information is necessary to complete this task: the type of the elements (usually stated with a code or an acronym), the physical properties of the element (e.g. Young's modulus, mass density, area moment of inertia, stiffness and damping coefficients), the coordinates of the nodes and the numbers of every degree of freedom that the element owns. Now, it is requested to link every dof, from the local numeration to the global one, as Fig. 19 shows. Each dof owned by an element has to be mapped to those related to the referential element.

This step is carried out to correctly insert the values associated with the element matrices and vectors into their global counterparts. In this way, each row/column index of the element matrix corresponds to a row/column index in the global matrix; similarly, the location of a value in the local vector matches with one in the global vector. So, each value can be stored in a precise cell of memory and, possibly, if the cell is not empty, the quantity will be added up to the previous value.

Remembering how matrices and vectors are distributed among processes (see Fig. 17), the contributions to the dofs are locally computed by the process owing the dofs, but other contributions may also come from other processes by

communications. This is the case when the dofs are referred to nodes shared by elements ascribed to different processes. An advice to optimize the implementation of the code may be to generate most of the entries on the right process, in order to avoid communication. Note that, however, it is fine to produce some entries on another "wrong" process, because this can lead to cleaner, simpler and less buggy codes, thus accepting a certain amount of communication between processes.

Once structural dynamical FEM matrices **M**, **C** and **K** are assembled with the contributions from all the elements of the mesh, load vector **F** has to be completed yet. First of all, external concentrated forces applied to the nodes should be considered, reading the proper input file in which their direction and magnitude should be written. The value of force magnitude could simply be added to the right position of **F**, depending on the degree of freedom on which the force is imposed. Finally, **F** is completed when the contributes of the distributed forces on corresponding element sets are taken into account.

The assembly procedure is summarized in Algorithm 6.

Algorithm 6 Assembling global matrices and vectors

```
for each process do
    for each element owned do
        read type of element
        read properties of material
        read global numeration
        create element stiffness matrix
        create element damping matrix
        create element mass matrix
        create nodal force vector
        for each value in element matrices and vectors do
            map degree of freedom from local to global numeration
            if degree of freedom is owned by process then
                add nonzeros in a local cell of memory
            else
                send values to correct process
            end if
        end for
    end for
    evaluate external nodal forces
    evaluate distributed forces on physical sets of elements
    update F
end for
```

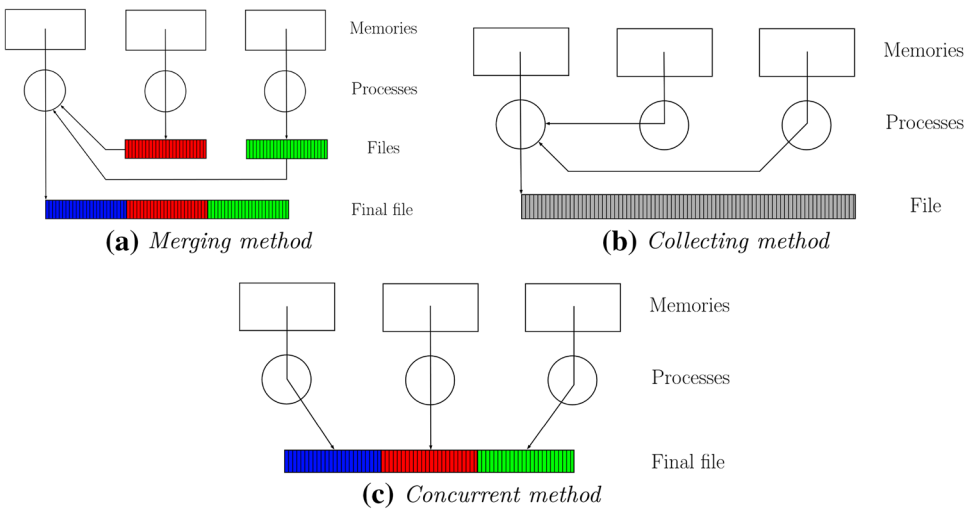


Fig. 20 Three different ways of writing a file for distributed systems

4.3.5 Time-Marching Solution

A dynamic solution should be obtained during a simulation, in order to describe the system evolution in time. It is the aim of a FEM code to solve the governing system of equations, as previously presented in Eq. (49):

$$\mathbf{M}\mathbf{q}_{,tt} + \mathbf{C}\mathbf{q}_{,t} + \mathbf{K}\mathbf{q} = \mathbf{F}, \quad \text{with} \quad \mathbf{q} = \mathbf{q}(t) \tag{75}$$

The response history of the physical system may be obtained by performing a *direct time integration*. The solution may be obtained using a step-by-step time integration, without changing the form of Eq. (75).

Above all, an appropriate time step has to be set, depending on the nature of the numerical simulation. Then, the temporal duration or the total number of the steps of the simulation has to be set. Suddenly, the implemented parallel code should display an overall view of the shared global matrices and vectors. The role of processes is to efficiently and concurrently solve a sequence of linear or nonlinear systems.

The interested reader may consider Davis [30], as an introduction to the direct methods involved in the solution of sparse linear system. The typical steps to solve a generic sparse linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ may be resumed as follows:

1. Find a permutation in order to reduce the fill-in associated with factorization;
2. Factorize permuted matrix **A** using Cholesky, LU, or QR algorithms;
3. Permute right-hand side **b**;
4. Solve for **x** using forward/backsolve;
5. Permute solution **x**.

Several packages perform a parallel factorization: some examples of Cholesky and LU algorithms for distributed systems are provided by Amestoy et al. [5] and Hénon et al. [79].

However, in case of system nonlinearities, an iterative procedure shall be employed to recover the approximate solution at every time step. The SNES library of PETSc provides powerful numerical routines to solve nonlinear problems, by using methods such as the nonlinear Richardson, GMRES, Gauss–Seidel and Krylov algorithms.

The solution of the problem is evaluated at instants separated by time increments Δt; so, nodal values are computed at times Δt, $2\Delta t$, ..., $n\Delta t$. At the nth time step Eq. (75) becomes:

$$\mathbf{M}\mathbf{q}_{n,tt} + \mathbf{C}\mathbf{q}_{n,t} + \mathbf{K}\mathbf{q}_{n} = \mathbf{F}_{n} \tag{76}$$

Discretization in time is accomplished by using Finite Differences approximations of time derivatives. The direct integration computes the unknowns at time step $n+1$ from the equations of motion and the known conditions at one or more of the preceding time steps. Algorithms can be classified as being *explicit* or *implicit*.

An explicit algorithm uses expressions of the general form

$$\mathbf{q}_{n+1} = f\left(\mathbf{q}_{n}, \mathbf{q}_{n,t}, \mathbf{q}_{n,tt}, \mathbf{q}_{n-1}, \ldots\right) \tag{77}$$

so, the solution is generated by the combination of Eq. (77) with Eq. (76).

Instead, an implicit procedure adopts expressions of the general form

$$\mathbf{q}_{n+1} = f\left(\mathbf{q}_{n+1,t}, \mathbf{q}_{n+1,tt}, \mathbf{q}_{n}, \mathbf{q}_{n,t}, \mathbf{q}_{n,tt}, \ldots\right) \tag{78}$$

Fig. 21 Data structure adopted to describe an element

```
struct element
{
  unsigned int elementType;
  unsigned int material;
  Eigen::Matrix<int, Eigen::Dynamic, 1> connectivity;
};
```

Fig. 22 Data structure adopted to describe a node

```
struct node
{
  unsigned int nodeNumber;
  Eigen::Matrix<double, Eigen::Dynamic, 1> coordinates;
  Eigen::Matrix<int, Eigen::Dynamic, 1> dofs;
  Eigen::Matrix<double, Eigen::Dynamic, 1> U;
  Eigen::Matrix<double, Eigen::Dynamic, 1> Ud;
  Eigen::Matrix<double, Eigen::Dynamic, 1> Udd;
  unsigned int isShared;
};
```

which is arranged with Eq. (75) at time step $n+1$.

In practical applications, the choice between explicit or implicit solver is related to the numerical stability and economy of the method.

Dealing with explicit methods, attention should be paid to the selection of the time step. Usually, these approaches reveal to be only *conditionally* stable; so, a stability analysis should be carried out before using them. On the other hand, implicit methods are often *unconditionally* stable, which means that calculations remain stable, regardless of the size of time step Δt.

However, the amount of calculations of each method should be challenged. Simple algebraic operations are necessary for an explicit method to pass from step n to $n+1$, because all data are known at the nth step but the number of time steps increases, since a small Δt may be employed to guarantee stability.

On the contrary, for an implicit method, the number of the time steps usually decreases, thanks to the possibility to employ larger time increments but, at each time step, linear or nonlinear systems have to be solved. So, the computational duty proves to be harder and harder, especially for those problems with a large number of degrees of freedom.

4.3.6 Printing the Solution

When the evolution in time of the solution has to be displayed, it is a good practice to write a proper file during the simulation and then visualize it through a useful external viewer. The solution file should contain the solution values, e.g. of displacements (velocities and accelerations) for each node of the mesh and at each time instant.

As described in Gropp et al. [72] and displayed in Fig. 20, different ways appear to write down the solution for distributed systems:

1. *Merging method*: it consists of creating a file for each process, containing only a part of the solution. Then, the designated process takes over all stored files and rejoins them in the right order in a single file;
2. *Collecting method*: before creating any file, one among the processes collects information from all the members of the distributed system and then writes the solution file;
3. *Concurrent method*: it is based on the creation of a specific format of the file shared between processes, which have the ability to simultaneously write, by specifying the precise location where data have to be stored.

Obviously, the third method turns out to be the most efficient one (but also the most difficult) to be implemented, because no sequential operations are involved and it minimizes the time for data transfer.

Algorithm 7 presents a possible solution to the printing problem using the *concurrent* method.

Algorithm 7 Printing the solution file for distributed systems

```
create a shared file
set position (pos) in file where processes have to write
for each time step t do
    for each process labeled as p do
        open shared file
        allocate a data structure for stream of characters to print
        for each owned node do
            insert displacement values in stream
            insert velocity values in stream
            insert acceleration values in stream
        end for
        compute length of stream (size_p)
        share size_p with all processes
        write stream in file at position pos + sum_{q=1}^{p-1} size_q
    end for
    close shared file
    compute posUpdate = sum_q size_q with q overall number of processes
    compute pos = pos + posUpdate
end for
```

Table 2 Non-dimensional time comparison between different programming languages performing some algorithms

Algorithm	C/C++ (gcc 4.8.5)	Julia (v. 0.6.0)	FORTRAN (gcc 4.8.5)	MatLab (R2017a)	Python (v. 3.5.4)
`iteration_pi_sum`	1.00	1.00	1.00	1.00	20.25
`recursion_fibonacci`	1.00	1.88	0.58	20.29	98.69
`recursion_quicksort`	1.00	0.94	1.30	3.08	37.51
`parse_integers`	1.00	1.35	5.39	238.10	18.77
`print_to_file`	1.00	0.66	3.44	116.70	1.38
`matrix_statistics`	1.00	1.74	1.87	17.68	17.78
`matrix_multiply`	1.00	0.98	1.27	1.18	1.17
`userfunc_mandelbrot`	1.00	0.76	0.75	19.48	142.29

Benchmark times are relative to a C/C++ implementation (C/C++ performance = 1.0)

Table adapted from Julia [14] benchmark web page

These benchmark results were obtained on a single core (serial execution) on an Intel(R) Core(TM) i7-3960X 3.30GHz CPU with 64 GB of 1600 MHz DDR3 RAM, running openSUSE LEAP 42.2 Linux

Table 3 Elapsed time recorded for linear and nonlinear simulations running the MatLab code by Froio et al. [51] and the current C++ implementation, using a single core

Type of implementation	Elapsed time (s)	
	Linear simulation	Nonlinear cubic simulation
MatLab code	46.873	308.280
C++ code	5.910	46.902

Tests were conducted on a 200 elements (599 degrees of freedom) mesh, performing the numerical integration over 1200 time steps

Results were obtained on an Intel (R) Core i3-2370M 2.40 GHz CPU with 8 GB of RAM, running Ubuntu 16.04 LTS

5 Computational Features of the Implemented FEM Code

Throughout the previous sections, the ML problem concerning an elastic Euler–Bernoulli beam on a (visco)elastic foundation, the theoretical bases of its FEM modelization and a strategy to implement it within a parallel computing environment have been presented. In this section, an object-oriented C++ parallel FEM code developed in that framework is assembled and its computational advantages and performances are illustrated.

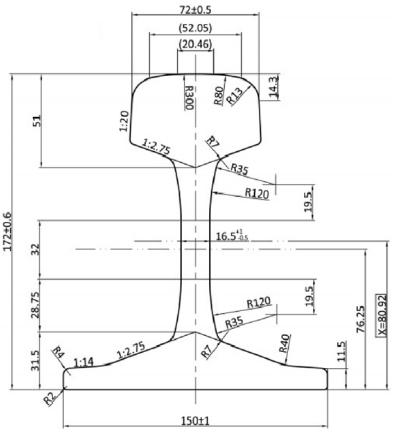
In Sect. 5.1, the fundamental features of the implemented FEM code are briefly sketched. Section 5.2 pertains to the comparison of performance between an earlier sequential MatLab code, implemented by Froio et al. [51], Froio [50] and the current object-oriented C++ parallel version of the algorithm employed in the present ML analyses. In Sect. 5.3, the numerical transient response of a simply supported beam posed on a (visco)elastic foundation and subjected to a ML, with either a constant or a harmonic-varying amplitude is considered. In order to validate the C++ code, the obtained numerical outcomes in terms of critical velocities are successfully compared with corresponding results from the literature [19, 51, 54, 56]. Finally, some benchmark performance results are reported in Sect. 5.4, in order to illustrate the achieved speedup, efficiency and scalability of the proposed C++ parallel code implementation, in the considered ML context.

5.1 Brief Description of the Implemented Code

The implemented code is made up of a set of functions and data structures that allow the user to set a general FEM problem on both a serial and a distributed system, by using some specific routines. The parallel FEM code is based on the following computational tools:

- *C++* [86]: although the conceived algorithm could certainly be implemented using other coding languages, e.g. FORTRAN or Julia [14], C++ offers an object-oriented structure that accommodates the problem into a collection of data structures and operations involving such structures. C++ consists of a core language specification containing basic data types and constructs, and a collection of built-in functions gathered in libraries. In this context, the concept of *functions* has been introduced, to ensure the modularity of such a programming language, and the concept of *class* has been created, to encapsulate data and operations on them.
- *GMSH* [67]: it is a free 3D finite element mesh generator with a built-in CAD engine and a post-processor. Its design goal is to provide a fast, light and user-friendly meshing tool, with parametric input and advanced visualization capabilities.

- *METIS* [92]: it is a set of programs for partitioning graphs, partitioning finite element meshes, and producing fill-reducing orderings for sparse matrices. The algorithms implemented within METIS are based on multilevel recursive-bisection, multilevel k-way, and multi-constraint partitioning schemes.
- *Eigen* [62]: it is a high-level C++ open source library of template headers for linear algebra, matrix and vector operations, geometrical transformations and numerical solvers.
- *Open MPI* [115]: the Open MPI Project is an open source Message Passing Interface (MPI) implementation that is developed and maintained by a consortium of academic, research, and industry partners. Open MPI offers advantages for system and software vendors, application developers and computer science researchers and is installed on many of the TOP500 [150] supercomputers.
- *PETSc* [10]: it is a Portable, Extensible Toolkit for Scientific computation developed at the Argonne National Laboratory. PETSc provides data structures and routines for the scalable parallel solution of scientific applications modeled by PDEs. PETSc employs the MPI standard for all the message-passing communication and is intended for the use within large-scale application projects. Fur-

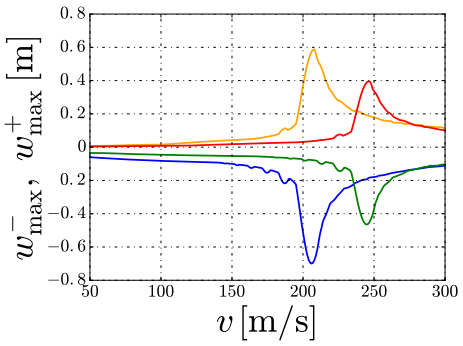


(a) *Rail profile, quotes in millimetres*

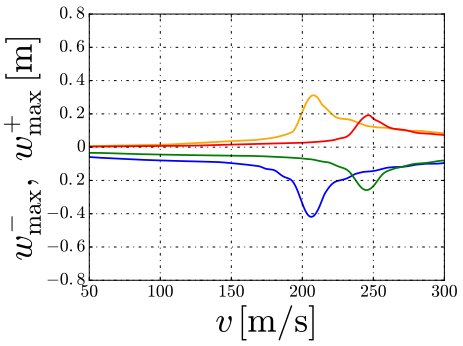
Fig. 23 UIC60 rail profile and its properties

Rail properties			
Young's modulus	E	210	GPa
Poisson's ratio	ν	0.3	–
Cross-section area	A	7684×10^{-6}	m^2
Area moment of inertia	I	3055×10^{-8}	m^4
Mass per unit length	μ	60	kg/m

(b) *Mechanical properties*



(a) $\zeta = 0\%$



(b) $\zeta = 2\%$

Fig. 24 Representation of beam maximum displacements as a function of moving load velocity, for linear (visco)elastic foundations with stiffness of $k_l = 250\,\mathrm{kN/m^2}$ (—upward and —downward displacements) and $k_l = 500\,\mathrm{kN/m^2}$ (—upward and —downward displacements) and two damping ratios. (Color figure online)

ζ (%)	v (m/s)			$w^-_{\max}, w^+_{\max}$ (m)		
	Present work	Ref. [19]	Error (%)	Present work	Ref. [19]	Error (%)
0	206	206	0.00	− 0.6999	− 0.700	− 0.01
	208	208	0.00	0.5873	0.587	0.05
2	206	206	0.00	− 0.4189	− 0.419	− 0.02
	208	208	0.00	0.3117	0.312	− 0.01

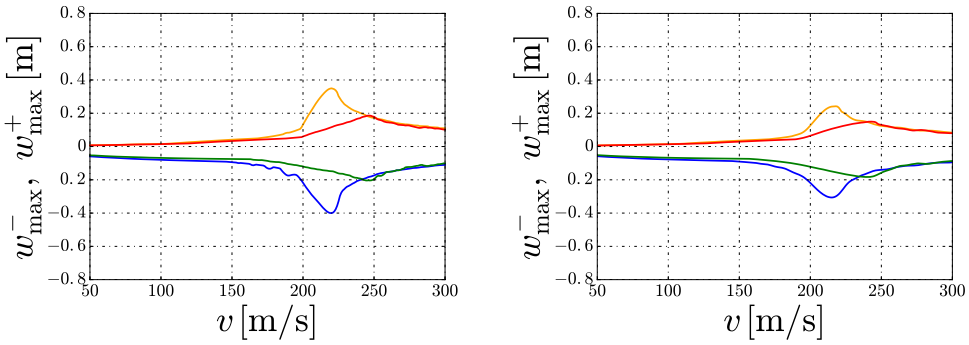
Percentage relative error with respect to Castro Jorge et al. [19]

Table 4 Critical velocities and maximum displacements for a linear foundation with stiffness $k_l = 250\,\mathrm{kN/m^2}$ and two different damping ratios

Table 5 Critical velocities and maximum displacements for a linear foundation with stiffness $k_l = 500\,\mathrm{kN/m^2}$ and two different damping ratios

ζ (%)	v (m/s)			$w^-_{\max}, w^+_{\max}$ (m)		
	Present work	Ref. [19]	Error (%)	Present work	Ref. [19]	Error (%)
0	245	245	0.00	− 0.4649	− 0.465	− 0.02
	246	246	0.00	0.3950	0.395	0.00
2	245	245	0.00	− 0.2582	− 0.258	0.08
	246	246	0.00	0.1922	0.192	0.10

Percentage relative error with respect to Castro Jorge et al. [19]



(a) $\zeta = 0\%$ **(b)** $\zeta = 2\%$

Fig. 25 Representation of beam maximum displacements as a function of load velocity for nonlinear elastic foundations with linear stiffness component $k_l = 250\,\mathrm{kN/m^2}$ and nonlinear stiffnesses of $k_{nl} = 2500\,\mathrm{kN/m^4}$ (—upward and —downward displacements) and $k_{nl} = 25{,}000\,\mathrm{kN/m^4}$ (—upward and —downward displacements) and two damping ratios. (Color figure online)

thermore, this library provides many of the mechanisms needed within a parallel application code, such as distributed matrix and vector assembly routines, which allow to overlap communication and computation.

The implemented code requires data from three entering information files, to arrive at the solution of a FEM problem. The first is a *mesh file* generated using GMSH and includes the nodal coordinates and the finite element connectivities. Furthermore, the description of the elements is fulfilled, with additional information about the acronym (i.e a code or a string), the type and the physical sets created to impose the material properties. The second file is an *input file*, which contains general data about the simulation, such as the number of degrees of freedom per each node, the properties of the materials and several parameters used to control the numerical routines employed to solve the ensuing FEM system. Furthermore, different kinds of boundary conditions are defined, like those on nodes or physical sets of elements. The third file is a *partition file* created by METIS, which reports the information to redirect each element to the correct process, thus performing calculations on a restricted amount of data. These three initialization files allow to initialize, on each process, two main data structures, built using a C++ native data type, together with those provided by Eigen. Similarly to the approach proposed by Bonnet et al. [17], a data structure is built up, to describe the elements (see Fig. 21) and the other one gathers the information about the nodes (see Fig. 22).

Table 6 Critical velocities and maximum displacements for a nonlinear foundation with $k_l = 250\,\mathrm{kN/m^2}$ and $k_{nl} = 2500\,\mathrm{kN/m^4}$ and two damping ratios

ζ(%)	v (m/s)	$w^-_{\max}, w^+_{\max}$ (m)		
	Present work	Present work	Ref. [19]	Error (%)
0	220	− 0.3999	− 0.400	− 0.02
	220	0.3497	0.349	0.20
2	215	− 0.3064	− 0.306	0.10
	217	0.2421	0.242	0.04

Percentage relative error with respect to Castro Jorge et al. [19]

Table 7 Critical velocities and maximum displacements for a nonlinear foundation with $k_l = 250\,\mathrm{kN/m^2}$ and $k_{nl} = 25{,}000\,\mathrm{kN/m^4}$ and two damping ratios

ζ (%)	v (m/s)	$w^-_{\max}, w^+_{\max}$ (m)		
	Present work	Present work	Ref. [19]	Error (%)
0	245	− 0.2042	− 0.204	0.10
	246	0.1861	0.186	0.05
2	241	− 0.1832	− 0.183	0.10
	242	0.1497	0.150	− 0.20

Percentage relative error with respect to Castro Jorge et al. [19]

The data structure called `element` is filled with the details on the type and the material of which an element is made up. Furthermore, it contains the connectivity, which is employed to store the nodes number owned by the element.

Structured data `node` is devised to collect the node number, the spatial coordinates deduced by the mesh file, the global numeration of the degrees of freedom and, above all, the unknown nodal values (in the present case, displacements `U`, and velocities `Ud` and accelerations `Udd`). Additionally, an integer (`isShared`) is declared to mark if a node is shared between two or more partitions.

When all input files have been read, each process initializes the above-mentioned data structures, per each owned element and owned node, as described by the partitioning file. Then, the simulation goes ahead, following the steps described in Sect. 4, by executing implemented Algorithms 3-7. The entire procedure is summarized in Algorithm 8.

Algorithm 8 Procedure to implement FEM on distributed systems

```
Initialize MPI_COMM_WORLD
for each process do
    read partitioning file
    read mesh file
    create data structure to store information about owned elements
    create data structure to store information about owned nodes
    read input file
    for each owned element do
        define properties (i.e. element type, Young's modulus,...)
        set boundary conditions if necessary
        acquire global numeration for each degree of freedom
    end for
end for
define global dimension of problem by counting unknowns
find nonzero pattern of global matrices
allocate memory for distributed matrices and vectors
for each process do
    for each owned element do
        assemble element contribution to global matrices and vectors
    end for
end for
get solution of system of ordinary differential equations
print solution
post-process collected data
finalize MPI_COMM_WORLD
```

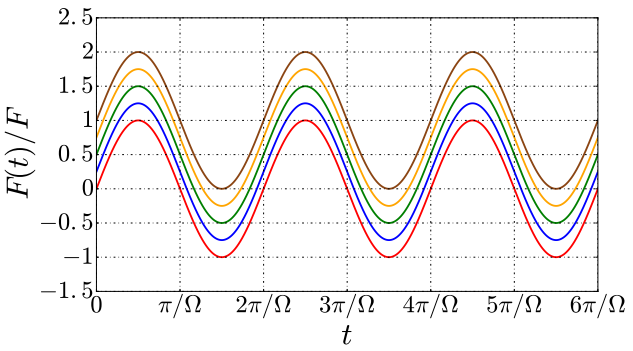


Fig. 26 Harmonic moving load magnitude $F(t)$ within the selected range of values of α ($\alpha = 0$, $\alpha = 0.25$, $\alpha = 0.5$, $\alpha = 0.75$ and $\alpha = 1$). (Color figure online)

5.2 Comparison Between Sequential MatLab and Parallel C++ codes

The present parallel C++ version of the FEM code for ML problems has been based on a previous MatLab [146] implementation [51]. The necessity of porting that project from MatLab to C++ has been derived from the quest of achieving a higher performance, in the demanding solution of the ML dynamical problem, and specifically in order to allow for the development of complete parametric analyses, involving really time-consuming ML simulations. The choice of the programming language has fallen on C++ due to various computational advantages, mainly: execution speed, efficiency, availability of open source implementations (i.e. GCC G++ [64], Microsoft Visual C++ [112]) and widespread exploitation on parallel systems (e.g. workstation clusters, High Performance Computing (HPC) consortia, etc.).

As explained in Fangohr [42], the main difference between C++ and MatLab is that the first one is a low-level compiled programming language, while the second one is a high-level interpreted scripting conceived to develop and to execute programs. This means that C++ needs to be compiled in order to generate an executable program, whereas MatLab reads the source code line by line and translates it on the fly into machine instructions. Interpreted languages usually present degraded execution speeds, due to the dynamic typing of variables, the possibility to have interactive sessions (i.e. the debug mode) and the lack of memory management. Some results that compare the efficiency of interpreted and compiled languages have been discussed, e.g., by Prechelt [128], showing the runtime execution and memory consumption to decrease, by employing C++. On the other hand, the total working time for assembling a C++ program is dilated by the quest of achieving a clean coding. Thus, as exposed by Ousterhout [122], different programming languages may be conceived, as different tools for different tasks. MatLab may be used to rapidly prototype the algorithm, thanks to its built-in toolboxes, routines and facilities for data visualization. However, the need of a High Performance Computing capability may be complied with C++, even though the programmer/user may have to carry out several skillful tasks: choose the correct compiler, include external libraries, allocate and deallocate memory, choose the correct data type and debug the code in the absence of a Graphical User Interface (GUI), such as in many clusters devoted to HPC.

Some benchmark results showing how interpreted languages such as MatLab and Python may turn out to display worse performances than compiled languages like C/C++ and FORTRAN for some standard algorithms are displayed in Table 2.

Similar results have been obtained in the present work, from the comparison of the MatLab and C++ implementations, used to solve linear and nonlinear ML problems. As it could be seen from Table 3, a serial C++ task generates a program that is much faster than a MatLab code.

Tests have been performed on a single core, in order to compare the efficiency of the serial algorithm by only changing the programming environment. The C++ implementation demonstrates to save a considerable amount of time, which is crucial during the numerical tests. Parametric analyses, such as those presented in next Sect. 5.3, consist of a sequence of ML simulations, for which some parameters, like velocity and magnitude frequency of the ML are varied. Thus, saving time on each single simulation allows to complete the entire work in a few hours, instead as of several days.

5.3 Validation of the Implemented FEM Code

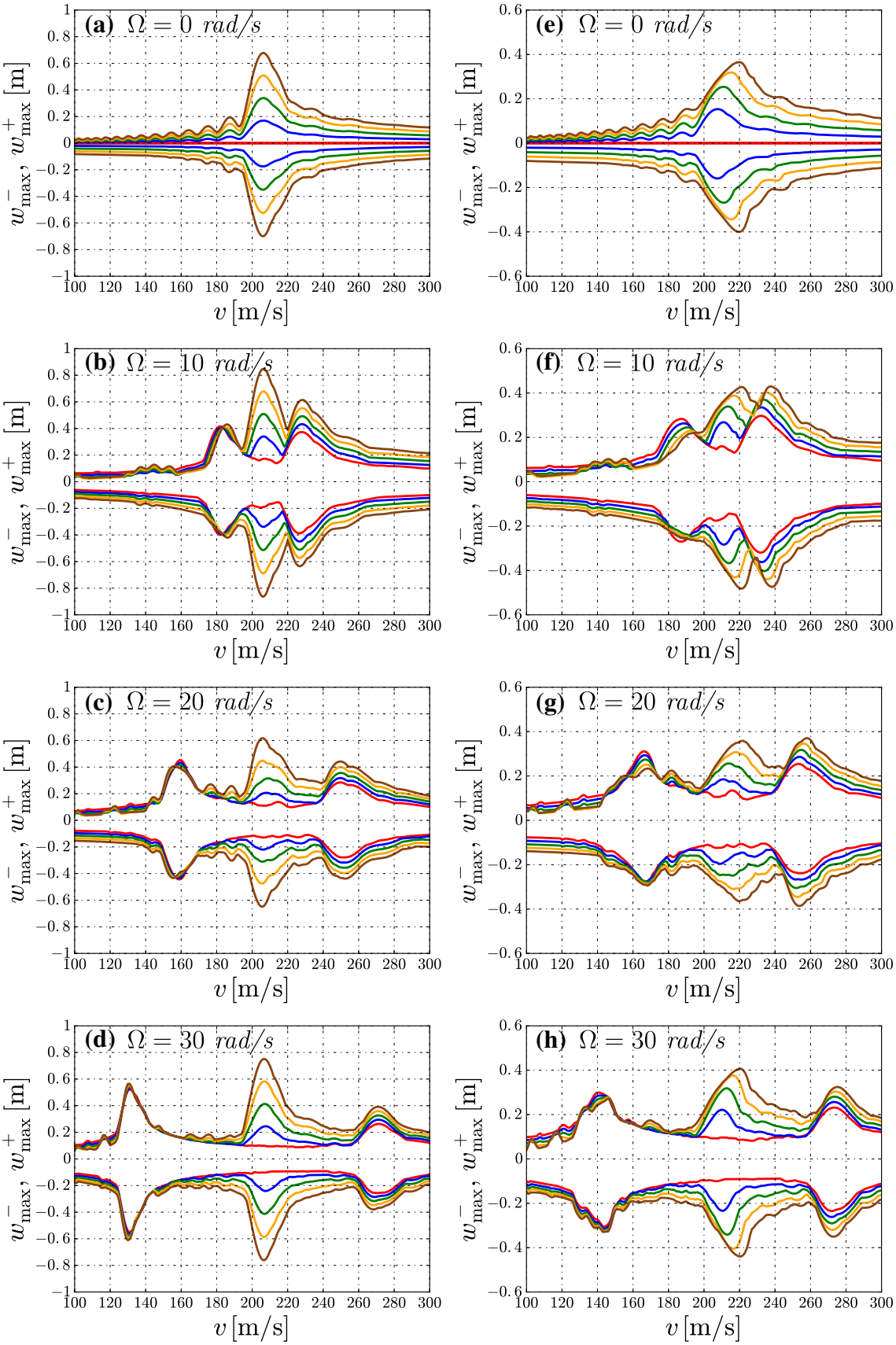
The present general FEM implementation has been developed and employed to solve the dynamical problem of a simply supported beam lying on a (visco)elastic foundation subjected to a ML, as previously described in Sect. 3.3.

Two types of foundation constitutive law have been herein considered: the classical linear law reported in Eq. (1) and the nonlinear cubic law stated in Eq. (4). The validation of the code has been carried out by comparing the maximum and minimum beam displacements and the critical velocities obtained by Castro Jorge et al. [19] and Froio et al. [51, 54, 56], by varying either only the ML velocity or both the velocity and the magnitude frequency of the ML. The critical velocities are figured out by seeking those values of ML velocity corresponding to the maximum attained transverse displacement (see e.g. Dimitrovová and Varandas [39]).

The considered type of beam is a UIC60 Vignole rail, one of the most diffused steel profiles adopted in railway tracks, in order to allow for a comparison with the results from the above-mentioned literature. The rail profile is shown in Fig. 23a, while its mechanical properties are reported in the array in Fig. 23b.

A finite rail length (L) of 200 m has been selected, in order to reasonably approach the limit case of a beam with an infinite length, at least in the subcritical velocity regime. For the analyses, the finite beam has been spatially discretized using 200 finite elements, each 1 m long. The self-weight of the beam has not been taken into account, since it seems not to effect much the dynamic response of the mechanical system and the resulting critical velocities.

The assumed load magnitude is $F = 83.4$ kN, the load acting downward. That value corresponds to a locomotive of the Thalys high-speed train (EU), which displays a total axle mass of about 17,000 kg.


(a) $\Omega = 0$ rad/s
(b) $\Omega = 10$ rad/s
(c) $\Omega = 20$ rad/s
(d) $\Omega = 30$ rad/s
(e) $\Omega = 0$ rad/s
(f) $\Omega = 10$ rad/s
(g) $\Omega = 20$ rad/s
(h) $\Omega = 30$ rad/s
$w^-_{\max}, w^+_{\max}$ [m]
v [m/s]

◂**Fig. 27** Representation of beam maximum displacements versus moving load velocity v and ratio $\alpha = F_0/F$ for a linear (**a**–**d**) and a nonlinear cubic (**e**–**h**) elastic foundation with stiffness $k_l = 2.5 \times 10^2$ kN/m^2 and $k_{nl} = 2.5 \times 10^3$ kN/m^4, in the undamped case. Load magnitude frequency Ω ranges from 0 to 30 rad/s. (—$\alpha = 0$, —$\alpha = 0.25$, —$\alpha = 0.5$, —$\alpha = 0.75$ and —$\alpha = 1$). (Color figure online)

Regarding the aspects of numerical integration, the HHT-α method has been implemented, either for linear or nonlinear systems (see Sect. 3.4). Here, the α parameter expressing the numerical dissipation rate is chosen equal to -0.1, according to Eq. (59).

5.3.1 Constant-Magnitude Moving Load

The first numerical analysis is concerned with the simply supported beam response under a constant-magnitude ML. The beam lies on a (visco)elastic foundation characterized by two different values of Winkler elastic coefficient k_l: 250 kN/m^2 and 500 kN/m^2. Both undamped and damped behaviors are taken into account. Damping factor ζ has been introduced to define constant a_0 in Rayleigh-damping Eq. (47) as

$$a_0 = 2\zeta\sqrt{\frac{2\,k_l}{\mu}} \tag{79}$$

Damping matrix $\mathbf{C}$ in Eq. (47) was evaluated by assuming either $\zeta = 0\%$ or $\zeta = 0.02\%$ and $a_1 = 0$.

Computations have been performed for velocities of the ML varying from 50 m/s up to 300 m/s, with a step variation of 1 m/s. Regarding aspects of numerical integration, the time span taken throughout the process corresponds to the amount of time along which the ML is really acting along the supported beam, by moving along it. The adopted time step corresponds to the time taken by the load to travel a distance of 0.2 m, namely a fifth of a finite element length.

Then, maximum upward $w^+_{\max}$ and downward $w^-_{\max}$ displacements of the beam have been recorded, for each simulation performed at a certain value of ML velocity. These values are subsequently plotted as a function of ML velocity.

The results dealing with the linear foundation are displayed in Fig. 24 and the consistent comparisons between present and previous [19] outcomes are gathered in Tables 4 and 5.

Within these tables, the percentage relative errors upon the values of critical velocity and minimum/maximum displacements are reported. A very good agreement may be noticed, for both the damped and undamped case. In particular, the retrieved small discrepancies are solely due to round-off differences. The results obtained with the present parallel C++ code turn out to be practically the same as those earlier generated within MatLab. These results are expected, since the present code is based on such a previous MatLab implementation, though reformulated within another programming environment and with the insertion of a parallel paradigm, and they demonstrate the reliability of the implementation.

By inspecting Fig. 24, increasing the stiffness of the foundation, it appears clear that a shift in the position of the critical velocities toward higher values and a decrease of the deflection amplitude occur. Moreover, it could be seen that damping does not sensibly affect the values of the critical velocities (peak position along horizontal axis) but it only decreases the amplitude of the beam response (peak elevation along vertical axis) .

Concerning the nonlinear cubic foundation, Fig. 25 presents the response of the beam and depicts the displacement peak corresponding to the critical velocities, whose values are portrayed in Tables 6, 7.

During the numerical tests, the nonlinear foundation has been characterized by two parameters: linear stiffness coefficient k_l, assumed to be constant and set at 250 kN/m^2; nonlinear coefficient k_{nl} considered equal to 2500 or 25,000 kN/m^4.

Then, it has been demonstrated that increasing the nonlinear coefficient, the critical velocities move to higher values, while the maximum displacements of the beam is reduced. Differently from the linear problem, the amount of damping influences either the values of beam maximum response or the critical velocity. In particular, the critical velocities decrease, introducing the effects of damping by using $\zeta = 2\%$.

5.3.2 Harmonic Moving Load

Up to now, the ML acting on the beam-foundation system has been characterized by a constant amplitude. In this section, the concentrated ML travels at a constant velocity along the beam, displaying also a harmonic-varying magnitude in time t, which is described by the following equation:

$$F(t) = F_0 + F\sin(\Omega t) = F[\alpha + \sin(\Omega t)] \tag{80}$$

where F_0 is defined as the mean value of the ML, F and Ω the magnitude and the frequency of oscillation, respectively. During the numerical tests, ratio $\alpha = F_0/F$ has been introduced, to inspect the response of the system to its variation. Parameter α is varied from 0 to 1, with steps of 0.25, in order to create different harmonic ML magnitude trends, as depicted in Fig. 26.

Additionally, the time step used during the numerical integration has been set as $\Delta t = \min\left\{10^{-3}\text{ s},\ h/(5v)\right\}$, where h is the referential size of a beam element in the mesh and v the velocity of the ML. So, adopted Δt ensures a sufficient accuracy of the results, for both the linear and nonlinear

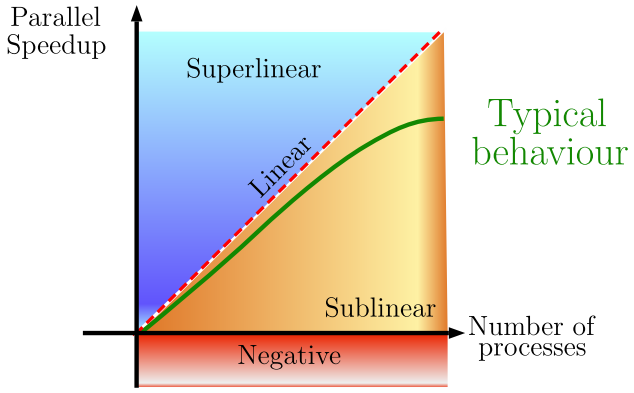
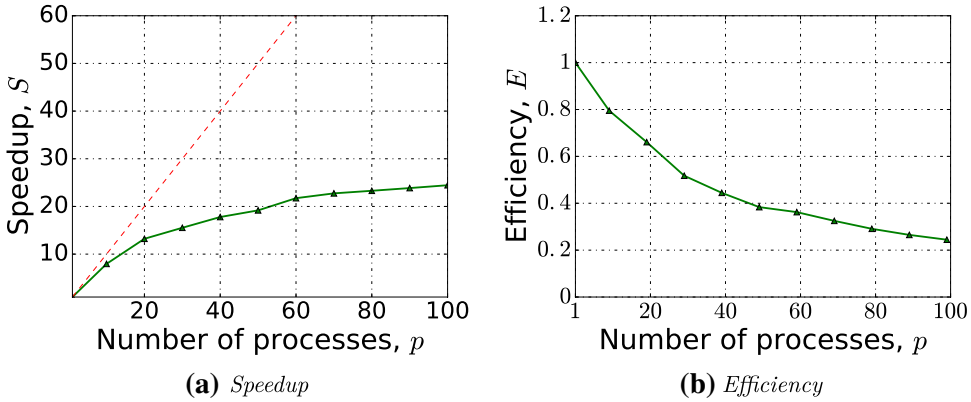


Fig. 28 Typical behaviour of speedup (adapted from Zhang [164])

foundation cases. The resulting maximum displacements are depicted in Fig. 27a–d for the linear case, and in Fig. 27e–h for the nonlinear one.

As explained by Chen and Huang [23], the obtained figures display the existence of two resonant peaks, corresponding to two different critical velocities, in case of a harmonic ML. They collapse into a unique peak, as the frequency of the harmonic amplitude of the ML approaches zero, leading to the results obtained by considering a constant-magnitude ML. Furthermore, the two critical velocities tend to separate, as the load frequency increases, giving rise to a third peak. In particular, the central peak corresponding to the second critical velocity remains stationary, even though the frequency of the ML increases. The growth of the Ω frequency induces the lowest and the highest critical

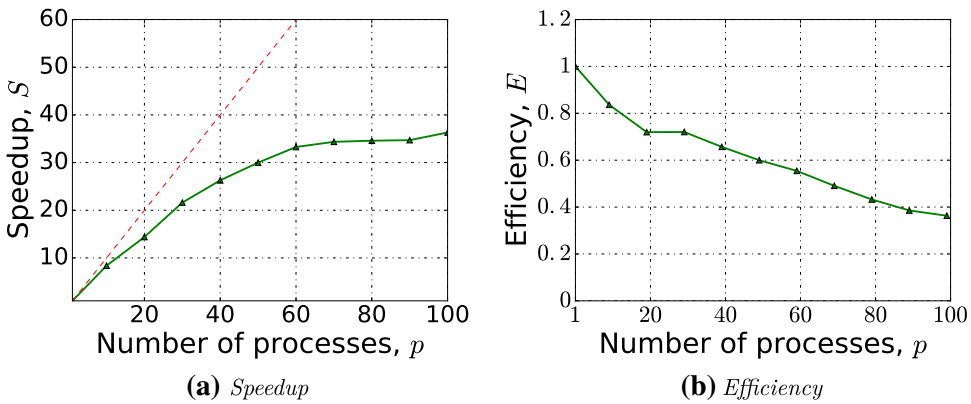


(a) *Speedup* **(b)** *Efficiency*

Fig. 29 Achieved speedup and efficiency for a linear moving load analysis varying the number of processes. The simulation consists of 59,999 degrees of freedom and 1000 time steps. The serial runtime is about $T_S = 98$ s. The green line provides the obtained trend, while the dashed red line represents an ideal linear speedup

Table 8 Scalability of the linear parallel algorithm is proven, presenting values of efficiency E that remain nearly constant, varying the number of processes (`mpiprocs`) and the size of the problem (mesh size)

Problem size	`mpiprocs`	Elapsed time (s)	Efficiency E
10,000 elements (29,999 dofs)	1	42.96	0.78
	6	9.20	
20,000 elements (59,999 dofs)	1	97.81	0.80
	10	12.32	
30,000 elements (89,999 dofs)	1	212.22	0.79
	26	10.31	



(a) *Speedup* **(b)** *Efficiency*

Fig. 30 Achieved speedup and efficiency for a nonlinear moving load analysis varying the number of processes. The simulation consists of 59,999 degrees of freedom and 5 time steps. The serial runtime is $T_S = 979$ s. The green lines provide the obtained trends, while the dashed red line represents an ideal linear speedup

Table 9 Scalability of the nonlinear parallel algorithm is proven, presenting values of efficiency E that remain nearly constant, varying the number of processes (`mpiprocs`) and the size of the problem (mesh size)

Problem size	`mpiprocs`	Elapsed time (s)	Efficiency E
10,000 elements (29,999 dofs)	1	453.91	0.84
	5	107.48	
20,000 elements (59,999 dofs)	1	979.0	0.84
	10	117.05	
30,000 elements (89,999 dofs)	1	1601.0	0.82
	20	97.78	

velocities to move toward the limits of the plot. Moreover, in the case of a nonlinear cubic foundation, the effect of increasing the magnitude mean value through α is to further reducing the lower critical velocity and at the same time to increase the other two critical values.

5.4 Achieved Computational Performance of the Parallel Implementation

The execution time of a sequential code is usually employed to evaluate its computational performance, changing the size of the input data to be processed. The elapsed time of a parallel algorithm relies not only on the input size of the problem but also on the number of processes involved and on their ability to efficiently communicate between themselves. Hence, the performance analysis of a parallel program has to be related to the parallel architecture (see Sect. 4.2), allowing for the execution of the algorithm, in order to evaluate the hardware platforms and to examine the benefits achieved by the scheduled parallelization. Some fundamental concepts have to be introduced, on the purpose to carry out such a kind of benchmark analysis. They are briefly presented in the following; more detailed information could be found in Hill [81], Kumar and Gupta [101] and Grama et al. [69].

Serial runtime T_S of a program is the elapsed time between the beginning and the end of the execution on a sequential computer (SISD). Instead, *parallel runtime* T_P is the amount of time that passes from the moment a parallel computation is launched, to the moment in which the last process finishes the execution of all the instructions given by its control unit.

One of the most intuitive parameters that may be employed to capture the benefit of solving a problem in a parallel form is the so-called *speedup*. Speedup

$$S = \frac{T_S}{T_P} \tag{81}$$

is the ratio of the serial execution time of the fastest known serial algorithm to the runtime of a parallel program that involves a number p of processes. It is well known that the speedup does not keep on increasing with the number of employed processes but it saturates at increasing p, as schematically shown in Fig. 28.

This could be explained by different considerations. If s is the *serial fraction*, defined as the number of components of an algorithm that cannot be parallelized, the speedup is bounded by $1/s$ for all the values of p, as stated by Amdahl [4]. Furthermore, a wide variety of overheads associated with parallelization may be detected. During the execution of a parallel program, processes spend time not only at performing essential computations but also while carrying out interprocess communication or idling, due to required synchronization. The total sum of all the overhead is mathematically defined as

$$T_O = p\,T_P - T_S \tag{82}$$

In practice, each process is not able to devote the whole time to computations. So, *efficiency* E may constitute a measure of the fraction of time for which a process is usefully employed. It could be expressed as

$$E = \frac{T_S}{p\,T_P} = \frac{1}{1 + \dfrac{T_O}{T_S}} \tag{83}$$

The analyses of the implemented parallel code have been led by using the previous parameter, in order to investigate the scalability achieved by the computational algorithm. The *scalability* of a parallel program on a parallel architecture may be defined as the ability to efficiently exploit the resources that are made available by the increasing number of processes. Kumar and Gupta [101] suggested that scalability may be used to select the best algorithm-architecture combination for a given problem. Moreover, it may be employed to predict the performance for a given set of processes or to determine the optimal number of them, to be used to achieve a certain value of speedup.

The performance analyses of the parallel algorithm involved in solving ML problems have been conducted by creating a referential challenge. The simply supported beam lying on a (visco)elastic foundation has been discretized by 20,000 finite elements, for a total amount of 59,999 degrees of freedom and the numerical integration has been performed on 1000 time steps.

The mesh has been refined, in order to challenge the capabilities of the parallel code. In fact, parallelism may completely be exploited when the physical problem presents a great number of unknowns. On the other hand, if the global dimensions of the matrices and vectors are relatively small, the overheads linked to the communication and the synchronization of the processes may kill the resulting performance, nullifying the benefits coming from the parallelization of the code. Further details could be found in the PETSc documentation or in Knepley [97].

The numerical experiments have been carried out on GALILEO, one of the three main clusters owned by Cineca, which is a non-profit consortium, made up of seventy Italian universities, four national research centers, and the Ministry of Education, University and Research (MIUR). GALILEO is composed of 516 nodes, each made up of two 8-cores Intel Haswell 2.40 GHz and 128 GB of RAM running the 64-bit CentOS 7.

Thus, a series of tests has been carried out, by varying the number of processes. As predicted by Amdahl law, the presented algorithm displays a sublinear speedup, which is depicted in Fig. 29a.

Regarding the achieved efficiency of the parallel algorithm, Fig. 29b reports the typical phenomenon that occurs in all parallel systems. Fixing the problem size, the efficiency of the parallel code goes down, due to the increasing overheads. Regarding the current implementation for solving linear ML problems, an important source of overheads comes from the necessity of communication. In fact, the global matrices and vectors are split among an increasing number of processes; so, during the time integration, a growing amount of messages has to be exchanged, in order to solve the series of linear systems.

For the given size of the problem, from Fig. 29 it may be seen that a number of processes greater than about 60 does not bring any further substantial improvements of performance. Furthermore, the maximum achievable speedup tends to be about 25 and, less than 20 processes should be employed to maintain a high level of efficiency.

In many parallel systems, it is possible to observe that an algorithm maintains efficiency E, fixed at a certain value, by simultaneously increasing the number of processes and the size of the problem. These combinations of algorithm and parallel architectures are defined as *scalable* parallel systems. The achieved scalability of the implemented code is reported in Table 8.

The size of the problem has been increased using a refined mesh; so, the number of degrees of freedom grows from 29,999 to 89,999 passing from a mesh with 10,000 elements to that with 30,000 elements. It could be seen that, increasing the size of the problem and the number of processes, the parallel algorithm maintains stable the resulting efficiency. It reflects the ability of the parallel algorithm-architecture combination to efficiently exploit increasing processing resources.

The same speedup and efficiency analyses have also been carried out for the parallel algorithm used to solve ML problems with a nonlinear cubic foundation. The computational resources required to obtain the solution of the nonlinear system increase, comparing to those for the linear case. In fact, the nonlinear HHT-α method (see Sect. 3.4) demands to find an approximate solution per each time step. This is very computationally expensive, due to the internal running of an iterative Newton-Raphson algorithm. So, numerical tests have been performed on a mesh with 20,000 elements, as in the previous case but the number of time steps has been significantly decreased to 5, in order not to waste the available computational resources, without losing accuracy.

GALILEO ended its production phase starting November 2017; so, the following results have been obtained on MARCONI A1, another cluster owned by Cineca. MARCONI A1 is composed of 1512 nodes, each made up of two 18-cores Intel Xeon E5-2697 v4 (Broadwell) 2.30 GHz and 128 GB of RAM running the 64-bit CentOS 7.

The speedup and the efficiency of the nonlinear parallel algorithm are displayed in Fig. 30. It could be seen that the reachable speedup for the nonlinear code is greater than that for the linear one, whereby S tends to about 35. Furthermore, even for this algorithm-architecture combination, the maximum number of processes that reduces the wall-time approaches 60, and a good level of efficiency may be obtained up to 50 processes. Finally, Table 9 demonstrates the achieved scalability of the parallel code, presenting the values of efficiency, for different meshes, with an increasing amount of elements, employing a growing number of processes.

6 Numerical Simulations of Moving Load on a Continuous Bridge Beam

Since the 19th century, the investigation of the dynamic response of structures and bridges subjected to a passing vehicle or a train has much grown in importance. The development of faster and faster transport systems and the discovery of high-strength materials that allow to build bridges with lighter structures and longer spans, has required a detailed consideration of the implemented calculation models, due to the far larger dynamic vibrations and internal stresses.

The construction of high-speed railways has demanded to define appropriate safety standards and to adopt sophisticated technologies to avoid passengers' discomfort. Expertise and know-how on the dynamic analysis techniques have been developed in the field of ML problems, with repercussions also on the European regulations.

In November 1995, the European Rail Research Institute (ERRI) decided to founding a committee of experts (ERRI D-124), in charge of studying problems connected to high-speed trains. Final report drafted by ERRI D-124 [29] explained that the excessive vibration of a bridge deck may cause ballast destabilization, in some bridges along the high-speed railway line connecting Paris to Lyon. That destabilization occurred due to a resonance phenomenon caused by the correspondence between the loading frequency of the train and the natural frequency of the bridge. After further investigations on the bridge response, the Comité Européen de Normalization (CEN) has established the maximum admissible bridge deck acceleration to 3.5 m/s^2, for a ballasted track (see CEN [20]).

Museros [117] upheld the importance of focusing on some specific aspects during the creation of ML models. He pointed out the significance of studying the distribution of the load through the sleepers and the ballast layer, using a train-bridge interaction model. Furthermore, various issues should be taken into account, such as the real value of structural damping during the passage of a train, the boundary restrictions exerted by the rail in the transition over the abutments, the vibration of the ballast layer and the effects of the excitation induced by track irregularities and wheel flats.

Ramondenc [129] argued about the importance of controlling the dynamic behaviour of railway bridges. In fact, without that, bridges and viaducts may experience rupture phenomena by the crossing trains. Some physical models have been developed, in order to investigate how the characteristics of the bridge shall affect the transient response. Possibly, the degradation of the mechanical properties of the structure should be considered, in order to schedule proper maintenance programs. The universal train model, also called High Speed Load Model (HSLM), was incorporated into the Eurocodes, with the aim of elaborating new regulations based on the resistance and structural durability and the serviceability of bridges.

Goicolea-Ruigómez and Castillo [68] reviewed and commented on the analysis methods, from a practical point of view. They explained how some parameters could be introduced during the investigation of the bridge response. *Impact factor* Φ may be exploited, to obtain the dynamic response, by amplifying standard static results. The values of Φ could be found in engineering codes published by CEN, Union International des Chemis de Fer (UIC) and Ferrovie dello Stato [48]. However, the impact factor does not include resonance effects possibly caused by the passing of a train. So, some other models have been developed. The *dynamic train signature* expresses the response of simply supported bridges on a combination of harmonic series involving the *dynamic signature*, namely the distribution of the train axles, and the *dynamic influence line*, which is a parameter depending on the span, the first natural frequency and the damping. Moreover, the consideration of a dynamic vehicle-structure interaction may lead to more realistic predictions of the bridge behavior.

Further insights on the dynamic analyses and practical design of bridges subjected to moving vehicles could be found in the collection of articles edited by Delgado et al. [31]. The interested reader should also be dwelt, e.g., on the works by Bogaert [16], Gabaldón Castillo et al. [61] and Hoorpah [82].

Together with the methods mentioned in this section and in Sect. 2, ML FEM models, even describing the train-bridge interaction and the resonance phenomena, provide a general methodology to obtain the dynamic response of a structure, like that of a bridge. In this section, the FEM algorithm and its implementation developed in Sects. 3 and 5 has further been employed to evaluate the dynamic response of a 1D multi-span model of the upper beam of the Paderno d'Adda Bridge (1889), a historical riveted iron arch bridge built nearby Milano, Italy, for railway and road traffic uses [44–47, 118, 142]. Here, the main scopes are those of providing an estimation of its dynamic response, ideally caused by a ML as traveling along the viaduct even at a theoretically high constant velocity.

Section 6.1 briefly introduces the technique exploited by engineer Jules Röthlisberger to design the Paderno d'Adda Bridge at the Società Nazionale delle Officine di Savigliano (SNOS), Cuneo. Furthermore, some technical aspects are

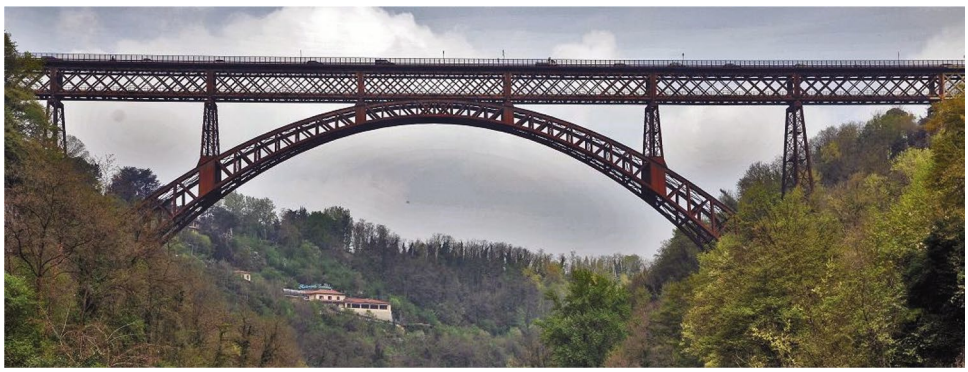


Fig. 31 Down-stream view of the Paderno d'Adda Bridge (1889)

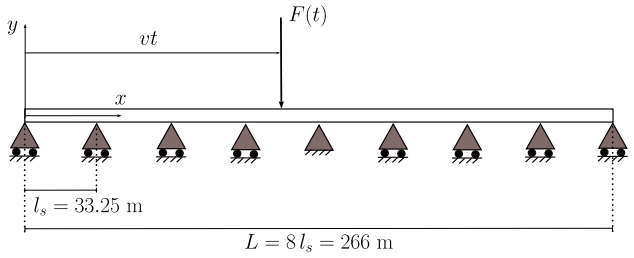


Fig. 32 Multi-span 1D model of the upper continuous beam of the Paderno d'Adda Bridge, with hinge at midspan, as conceived by the SNOS builder, and roller supports

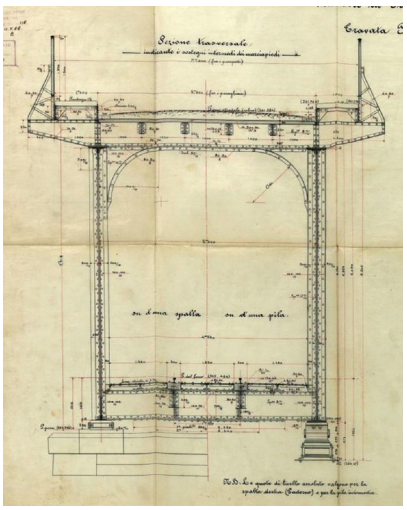
summarized, to provide a brief description of the structure of the iconic bridge. In Sect. 6.2, the upper box-beam of the Paderno d'Adda Bridge is then modeled as a 1D multi-span continuous beam on nine firm supports. Additionally, the parameters employed during the numerical tests are presented. Finally, Sect. 6.3 displays the effects caused by varying the ML velocity, the bending stiffness and the structural damping of the mechanical system. Moreover, several pictures of the time history of the beam response, once subjected to a concentrated ML are depicted, through a sequence of frames displaying its displacements, velocities and accelerations.

6.1 Brief Description of the Paderno d'Adda Bridge (1889)

The Paderno d'Adda Bridge is a railway metallic viaduct that crosses the Adda river between Paderno d'Adda and Calusco d'Adda, to a height of approximately 85 m from water, allowing to connect the two provinces of Lecco and Bergamo, near Milano, in the Lombardia region, northern Italy. The riveted iron bridge was rapidly built between 1887 and 1889, to comply with the needs of the growing industrial activities within the region. The viaduct was designed by Swiss engineer Jules Röthlisberger (1851–1911), head of the Technical Office of the Società Nazionale delle Officine di Savigliano (SNOS), Cuneo, the company that built the bridge. As reported in Ferrari and Rizzi [47], the bridge was designed through a graphical-analytical method of structural analysis, by applying the so-called "Theory of the Ellipse of Elasticity". The latter analyzes the flexural elastic response of a structure, by relying on an intrinsic discretization of a continuous beam into a series of elements, each with a proper "elastic weight", directly proportional to its length and inversely proportional to its bending stiffness. So, the problem of determining the flexural elastic response of a beam is reduced to a problem of pure geometry of masses, of a more convenient solution and direct interpretation in terms of the design of the structure.

The bridge, depicted in Fig. 31, displays a 266 m long upper flyover made by a continuous box girder with nine equally distributed supports, at a 33.25 m relative distance. Four of the supports are sustained by a marvellous doubly built-in parabolic arch, a beautiful characteristic feature of the bridge; two of them bear directly on the same arch masonry abutments; a seventh, on the Calusco bank, rests on a smaller masonry foundation placed between the arch shoulder and the higher bridge supports; the last two, in masonry work as well, are the two direct beam bearings at its two ends, on top of the two river banks. The four piers resting on the arch are symmetrically placed, in between

Fig. 33 Bridge beam cross-section and mechanical properties of the corresponding continuous beam employed in the multi-span 1D FEM model



(a) *Box-beam section of the Paderno d'Adda Bridge (courtesy of Archivio Storico Nazionale di Torino).*

Beam properties			
Young's modulus	E	167	GPa
Cross-section area	A	8573×10^{-4}	m^2
Area moment of inertia	I	9529×10^{-4}	m^4
Mass per unit length	μ	6602	kg/m

(b) *Mechanical properties of the box-beam of the Paderno d'Adda Bridge, modeled as a 1D continuous beam in the ML FEM simulation.*

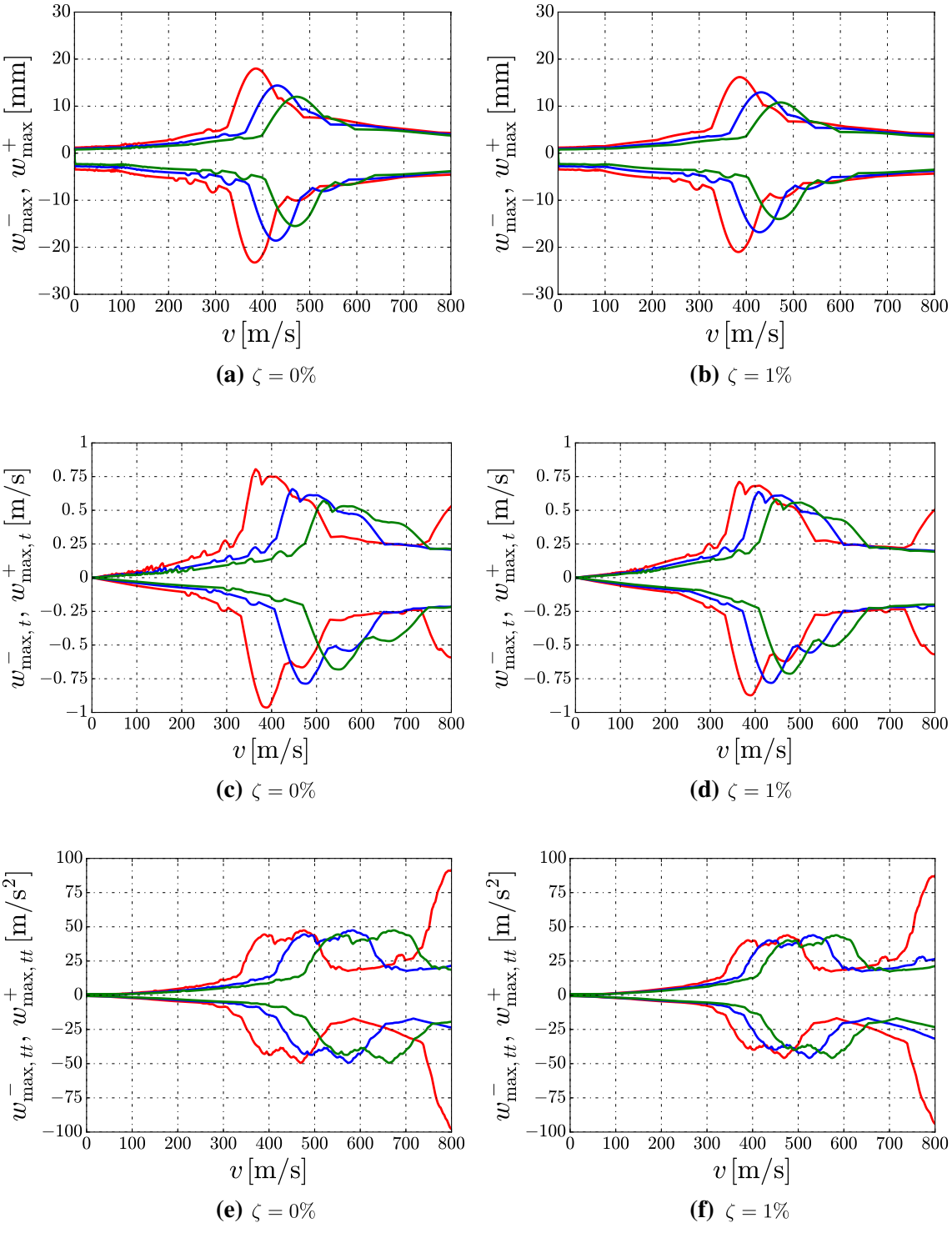
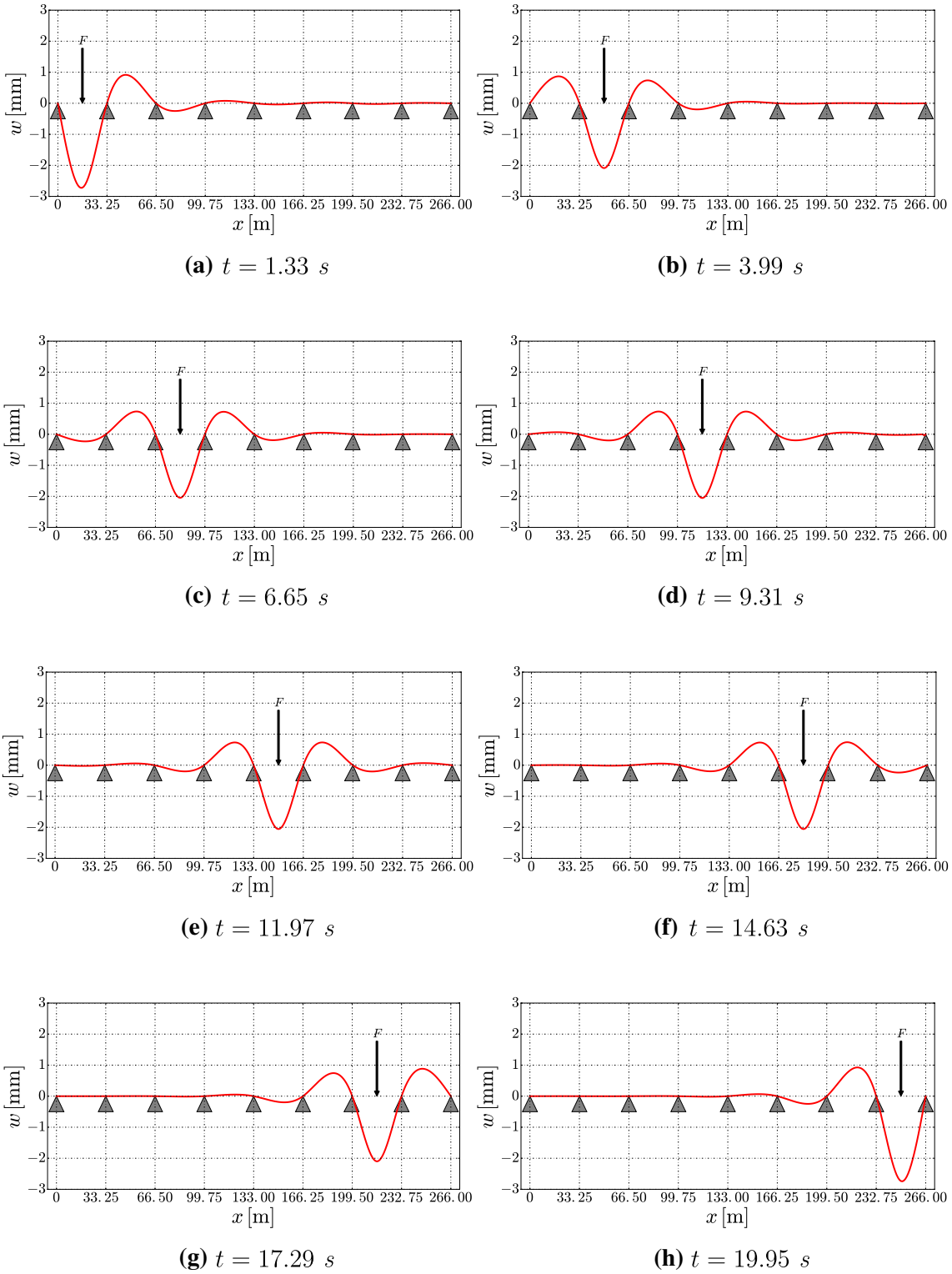


Fig. 34 Representation of beam maximum displacements, velocities and accelerations as a function of moving load velocity for the multi-span 1D beam model varying the value of EI (— 80% EI, — $EI = 1.62 \times 10^{10}$ kg m^2 and — 120% EI), for different values of (light) damping ratio. (Color figure online)

keystone, haunches and shoulders of the arch. The big arch is composed of two couples of secondary inclined arches. Each couple is formed by two arches posed at a respective distance of 1 m and symmetrically laying to a mean plane inclined to the vertical. Practically, the resulting cross section of the main parabolic arch supporting the horizontal



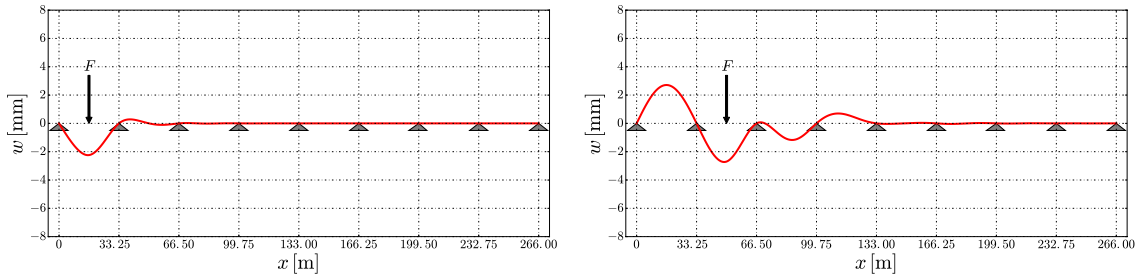
(a) $t = 1.33\ s$ **(b)** $t = 3.99\ s$

(c) $t = 6.65\ s$ **(d)** $t = 9.31\ s$

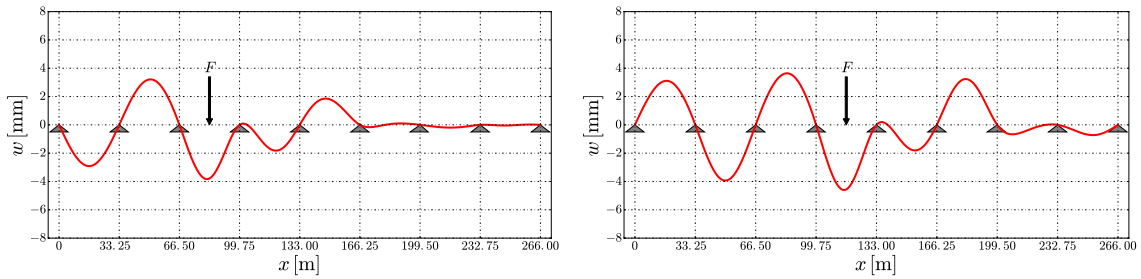
(e) $t = 11.97\ s$ **(f)** $t = 14.63\ s$

(g) $t = 17.29\ s$ **(h)** $t = 19.95\ s$

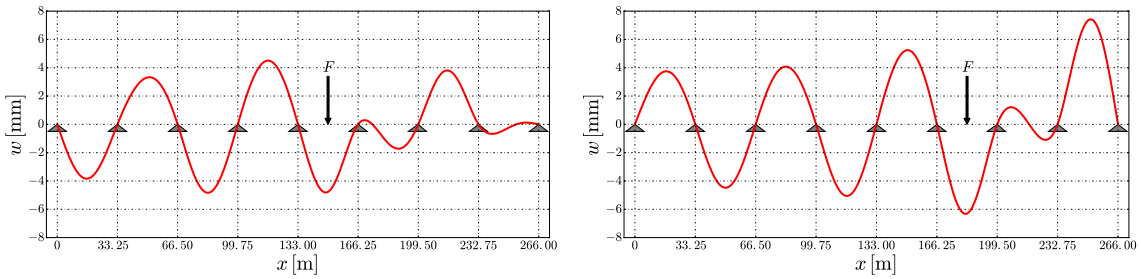
Fig. 35 Time shots of the multi-span 1D beam model subjected to a constant-magnitude load moving at constant velocity $v = 12.5$ m/s = 45 km/h. Frames represent the beam deflection when the load is at the middle of each span. Damping in not taken into account ($\zeta = 0$)



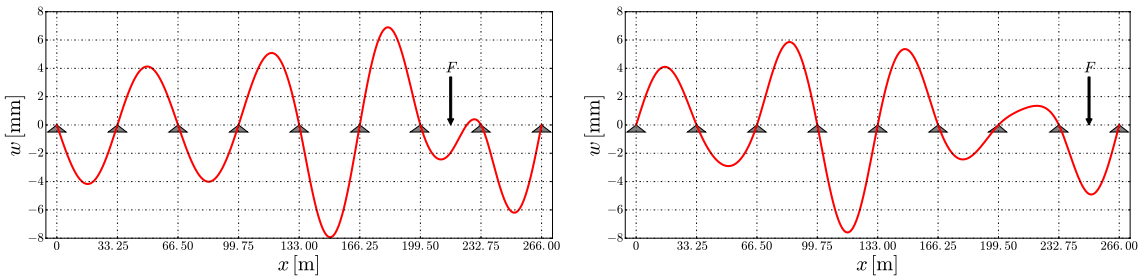
(a) $t = 0.0380\ s$ **(b)** $t = 0.1157\ s$



(c) $t = 0.1929\ s$ **(d)** $t = 0.2700\ s$



(e) $t = 0.3472\ s$ **(f)** $t = 0.4243\ s$



(g) $t = 0.5015\ s$ **(h)** $t = 0.5786\ s$

Fig. 36 Time shots of the multi-span 1D beam model subjected to a constant-magnitude load moving at a constant velocity equal to critical value $v = 431$m/s $= 1552$ km/h. Frames represent the beam deflection when the load is at the middle of each span. Damping in not taken into account ($\zeta = 0$)

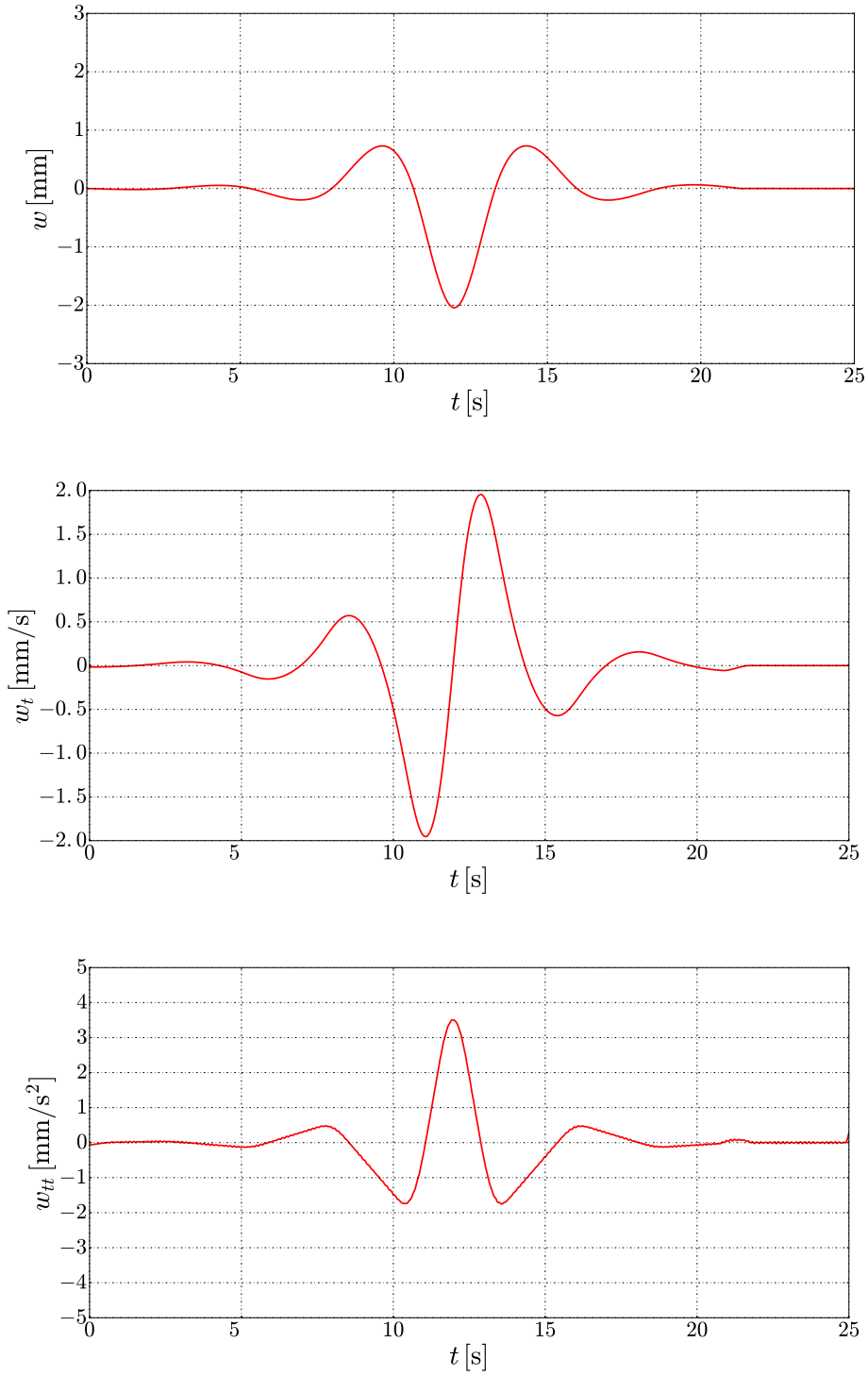


Fig. 37 Representation of the time histories of displacement, velocity and acceleration recorded at the middle point of the fifth span of the 1D beam's bridge, from left end where moving load departs. The velocity of the moving load is $v = 12.5$ m/s $= 45$ km/h. Damping is not taken into account ($\zeta = 0$). Plots have been obtained by applying a numerical filter in order to smoothen the spurious oscillations in the highly dynamic solution and reveal its primary nature

beam is trapezoidal, with a variable, increasing cross section from the crown to the shoulders. The vertical bridge piers that sustain the upper continuous beam are made by eight T-section columns, linked to each other by a bracing system with horizontal bars and St. Andrew's crosses and, on top, by transverse beams that directly serve as supports for the bearing devices of the upper beam. The main vertical longitudinal trussed beams of the upper continuous girder are 6.25 m high and placed at a respective transverse distance of 5.00 m, leaving a free passage for the trains of 4.60 m. They are composed of two main T-ribs connected by a metallic truss. The upper-level road is 5.00 m wide and includes also two additional cantilever sidewalks, each 1 m long, with iron parapets 1.50 m high. Further information on

Table 10 Maximum upward and downward values of displacements, velocities and accelerations varying the moving load velocity. Damping in not taken into account ($\zeta = 0$)

v (km/h)	$w^{+}_{\max}$ (mm)	$w^{-}_{\max}$ (mm)	$w^{+}_{\max,t}$ (mm/s)	$w^{-}_{\max,t}$ (mm/s)	$w^{+}_{\max,tt}$ (mm/s^2)	$w^{-}_{\max,tt}$ (mm/s^2)
10	0.9523	− 2.764	0.7712	− 1.375	384.4	− 600.7
20	0.9555	− 2.767	1.543	− 2.749	392.1	− 633.3
30	0.9579	− 2.768	2.314	− 4.123	399.8	− 666.1
40	0.9550	− 2.774	3.116	− 5.496	407.6	− 698.8
50	0.9653	− 2.782	3.858	− 6.868	415.3	− 731.5
60	0.9691	− 2.800	4.632	− 8.239	423.0	− 764.2
70	0.9794	− 2.809	5.492	− 9.608	430.7	− 796.9
80	0.9805	− 2.796	6.196	− 10.97	438.5	− 829.6
90	0.9856	− 2.774	7.321	− 12.34	454.0	− 862.3
100	1.0042	− 2.800	8.611	− 13.71	461.7	− 895.0
110	0.9933	− 2.820	9.880	− 15.07	469.5	− 927.7
120	0.9997	− 2.786	10.96	− 16.42	528.2	− 960.4

the bridge may be accessed in Ferrari et al. [46], and therein quoted references.

6.2 1D FEM Modelization of the Bridge Beam

The FEM implementation presented in the previous sections has been employed to analyze the dynamic response of a simplified 1D model of the upper continuous beam of the Paderno d'Adda Bridge. Furthermore, the ideal critical velocity of the beam of the bridge subjected to a concentrated transverse ML has been investigated.

The bridge has been modeled as a unique uniform continuous beam resting on nine equally spaced firm supports, as depicted in Fig. 32. The total length of the beam is 266 m.

Although the real 3D box-form beam's structure is reduced to a 1D model, the equivalent mechanical characteristics of the 1D beam have been defined, in order to provide a realistic approximation of the global beam behavior. Thus, due to the complexity of the upper flyover, global bending stiffness EI of the continuous beam may be derived in different ways, for instance according to the following two different approaches:

- *FEM approach*: it consists in defining equivalent bending stiffness EI and mass per unit length μ of a 1D beam, by extracting them from a complete 3D FEM model of the upper box-beam of the bridge [46]. A possible strategy may be that of considering a unique simply supported span of length l_s. Two moments of a unitary magnitude may be applied at the extremes of the equivalent beam, inducing a relative rotation $\Delta\phi$ of the two cross-sections at the ends of the beam. From well-known elastic relation

$$\Delta\varphi = \frac{M\, l_s}{EI} \tag{84}$$

reading the relative cross-section rotation through a 3D FEM simulation, the value of equivalent EI may be extrapolated.

- *Geometric approach*: supposing Young's modulus to be constant and known over the beam, it relies on the definition of geometrical property I of the beam's cross-section, i.e. the *area moment of inertia*.

  This is considered, to find out the area moment of inertia of an equivalent beam representing the flyover of the Paderno d'Adda Bridge. Owing to a CAD representation of the flyover cross-section, the value of I may be evaluated. For simplicity, the cross-section of the bridge is concentrated into two main areas, to that purpose, representing the running elements within the upper and lower decks of the bridge, neglecting the contributions of the structural elements connecting the two decks. Thus, the area moment of inertia may be calculated by neglecting the contributions of vertical struts and St. Andrew's crosses, and by considering only the most recurring beam longitudinal elements and relevant cross-sections, involved in the assembly of the upper flyover.

The mechanical properties of the beam corresponding to the bridge cross-section in Fig. 33a are gathered in the table sketched in Fig. 33b. The value of equivalent bending stiffness EI has been compared with that provided by the Final Technical Report edited by the SNOS [142], and a reasonable agreement has been recovered among the two results. At design stage, the SNOS has set the minimum value of the bending stiffness equal to $EI_{SNOS} = 0.9846 \times 10^{10}$ kg m^2. Thus, considering a possible effect of a considered appropriate safety factor, the values obtained as from above, from the FEM approach ($EI_{FEM} = 1.617 \times 10^{10}$ kg m^2) and from the geometric approach ($EI_{CAD} = 1.625 \times 10^{10}$ kg m^2) may be considered as reasonably correct (being also much reciprocally homologous, at about $EI = 1.62 \times 10^{10}$ kg m^2).

The multi-span 1D FEM model has been employed for the examination of the dynamic response of the structure subjected to a ML. A locomotive of a weight of 83 t, traveling at a constant velocity is considered for the numerical tests. It corresponds to a transverse (vertical) ML of a magnitude value $F = 8.14 \times 10^5$ N, acting downward. That value corresponds to one of the six special locomotives adopted to test the bridge in the first original try-outs in 1889 (see SNOS [142] and Nascè et al. [118]).

Similarly to the numerical analyses presented in the previous section, the mesh size has been conceived as by imposing an element size of nearly 1 m. However, since each span is 33.25 m long, the finite element size has been adjusted to about $h = 0.9779$ m, in order to obtain a uniform mesh, with an even number of elements (34) along each intermediate span. Thus, the mesh of the continuous 1D beam turns out to be composed of $34 \times 8 = 272$ beam finite elements.

Regarding the aspects of numerical integration, the α parameter expressing the numerical dissipation rate in the time integration is still taken equal to -0.1, according to Eq. (59). The time step involved in the numerical procedure corresponds to $\Delta t = h/(5\,v)$, where v is the ML velocity. The numerical integration is performed until the ML, starting from one support at one extreme of the whole beam, reaches the last support at the other end of the continuous beam.

Since here the effect of a distributed underlying elastic support along the beam is not present, the damping matrix cannot be evaluated according to Eq. (79) of Sect. 5. Thus, the effects induced by structural damping have been considered by employing Eq. (47), still for mass-proportional Rayleigh damping, this time imposing

$$a_0 = 2\zeta \sqrt{\left(\frac{\pi}{l_s}\right)^4 \frac{EI}{\mu}} \tag{85}$$

meaning that damping shall approximately affect the first mode of vibration of the multispan beam only. A more detailed description of structural damping could also be employed, although the adopted choice is believed to be sufficient, especially in relation to the considered simplified 1D mechanical model of the bridge beam.

6.3 Numerical Results

In this section, the effects of the ML velocity, of the beam bending stiffness and of the structural damping, on the maximum downward and upward displacements, velocities and accelerations of the continuous beam are presented. Given a value of the beam bending stiffness (with under- or over-estimation to such a reference value) and of the structural damping ratio, for each simulation performed at a certain value of the ML velocity, the maximum upward (positive) and downward (negative) displacements, velocities and accelerations of the beam are scored. Then, such outcomes of the FEM simulations are plotted as a function of the ML velocity. From such curves, the critical velocities of the beam-foundation system, for a finite beam, may be detected, as the velocity of the ML at which a maximum response is attained.

Moreover, considering only the first span, as just a single-span simply supported beam, a first reference estimation of the critical velocity may be obtained by employing the analytical equation provided by Dimitrovová and Rodrigues [38]:

$$v_{cr} = \frac{l_s}{\pi} \sqrt{\left(\frac{\pi}{l_s}\right)^4 \frac{EI}{\mu}} \simeq 463 \text{ m/s} \simeq 1667 \text{ km/h} \tag{86}$$

The relationships between beam maximum upward and downward displacements, velocities, accelerations and ML velocities, are shown in the sequence of plots in Fig. 34, respectively, for three values of bending stiffness (reference value plus/minus 20% of it, see table in Fig. 33) and for two values of structural damping, namely $\zeta = 0\%$ (undamped) and $\zeta = 1\%$ (lightly damped; light damping shall anyway be expected for such a wrought iron slender structure, as first assessed by experimental campaigns on the bridge, see Gentile and Saisi [65]).

The effect of increasing/decreasing the equivalent bending stiffness of the multi-span beam, as expected, is that of increasing/decreasing the value of the critical velocity. An opposite effect is displayed by the response amplitudes, which reduce by increasing the bending stiffness.

The effect of viscous damping results more complex. Regarding the maximum displacements (Fig. 34a, b), the effect is that of reducing the maximum response only, while keeping unchanged the value of the critical velocities, as also reported by Castro Jorge et al. [19] and Froio et al. [51]. It may also be noticed that the reduction of the response induced by viscous damping is effective only in the neighborhood of the critical velocity. This is due to the specific choice of the viscous damping matrix in Eq. (1). A richer structure of the damping matrix may lead to a structural damping acting on a broader range of the ML velocity. As an interesting feature of the present analysis, it may be remarked that by even adding a small amount of structural damping, as considered here, differently from the displacement response, the positions of the peaks of velocities and accelerations also vary.

Figures 35, 36 aim at representing the beam deflection when the ML is located at the middle point of each span, at no structural damping ($\zeta = 0$). The plots have been made by collecting the required data from the output file of the simulation, which gathers displacements, velocities and accelerations, for each time step and for each node of the FEM model. Detecting the correct time instant, the coordinates of

each node of the discretized and deformed beam have been stored in a different file, in order to plot the time shots.

This procedure has been carried out for two different values of locomotive velocity: $v = 12.5$ m/s $= 45$ km/h and $v = 431$ m/s $= 1552$ km/h. The first value corresponds to a realistic much subcritical velocity, which was employed in 1889 in the try-out tests on the bridge, although the present speed of the train crossing the flyover shall now be limited to about 10÷15 km/h. The second value coincides with the critical velocity detected thanks to the numerical tests depicted in Fig. 34a (see also Eq. (86)).

Observing the subcritical simulation, it could be noticed how the deflection of the beam appears to move together with the ML. The shape of the deformed beam shifts without changes, passing from a span to the next one. Furthermore, the maximum (upward) and minimum (downward) displacements remain fixed at constant values. The two main upward peaks are symmetrically situated within the previous and the next span regarding the ML position. Instead, the maximum downward displacement location coincides with the ML position.

Dealing with the critical velocity, the regularity of the beam deformation recorded in the previous case disappears, due to the arising resonance effects. The maximum upward and downward displacements are not always located at the same position, with respect to the ML location. The maximum upward displacement stands behind the ML in the shots from (*b*) to (*e*), while a local maximum grows ahead. Thus, in frame (*f*), that local maximum becomes the global one for the remaining duration of the ML passage, due to a constructive interference between the rightward wave and the leftward (reflected) wave. Regarding the maximum downward displacement, only shots (*a*), (*b*), (*c*), (*d*) and (*f*) display a behaviour similar to that in the subcritical case. In fact, frames (*e*), (*g*) and (*h*) depict the maximum downward value in a span preceding the ML position.

Another way of exploiting the output file is displayed in Fig. 37. The obtained plots have been drawn by gathering the displacement, the velocity and the acceleration values at a single node of the multi-span 1D FEM model of the beam of the bridge. In particular, it has been decided to monitor the kinematic response of the node located at the middle of the fifth span, which corresponds to the keystone of the underlying arch supporting the bridge. The results show the dynamic response of the beam in time, as if a sensor would be placed at that precise point.

Finally, Table 10 presents the maximum upward and downward displacements, velocities and accelerations, varying the velocity of the ML in a range of realistic values (that may theoretically explore the possibility of an increase of the present train speed). These data, coupled with the previous plots, may be employed, in order to analyze useful data at design-stage condition for the historical bridge, to guarantee security standards and passenger comfort as imposed by international regulations, or to schedule Structural Health Monitoring or maintenance operations.

Obviously, the ranges of running speed apt to contain the amount of acceleration response (last column of Table 10) should not go beyond what was originally scheduled at design stage (try-outs in 1889 carried out at a maximum crossing speed of about 45 km/h). This testifies the hint of not conceiving possible extra uses of the historical infrastructure that may go beyond the limited range of velocity around $10 \div 15$ km/h currently scheduled on the bridge, and this even disregarding possible ageing and damage effects that the monumental viaduct may have experienced in the 130 years of duty, as a combined railway and road bridge.

7 Conclusions

The present work has aimed at presenting a review framework in terms of computational Moving Load problem analysis and of parallelism implementation strategy in that context, by then developing a novel object-oriented FEM parallel implementation in C++, with the purposes of performing efficient numerical analyses for the dynamic response of beams under a high-velocity ML. This shall be of interested to people that may approach the field of Moving Load problem analysis, with the idea of possibly setting efficient numerical computations based on distributed computing, allowing to use HPC resources that may be accessed, e.g. on computer networks or consortia.

By employing the developed computer code, as briefly described in Sect. 4, it has been possible to draw down a complete C++ parallel implementation based on an earlier sequential FEM coding developed in Froio et al. [51], Froio [50] within a MatLab environment. The present implementation shall allow for a further exploitation and use in the context of HPC resources, toward achieving complete parametric studies, and in view of theoretical investigations on the physics of the underlying dynamic phenomena and of practical implications on the engineering response of structures subject to important ML regimes (for instance, for railway infrastructures and tracks).

The considered finite 1D beam ML problem has been regarded within two main configurations. Firstly, a simply supported beam lying on a distributed (visco)elastic foundation has been examined, in order to investigate the beam-foundation response, as resembling that of railway tracks. Secondly, a multi-span bridge beam has been modeled as a continuous beam on nine equally spaced firm supports. Both structural configurations have been subjected to a transverse concentrated ML traveling with a constant velocity along the beam. The magnitude of the ML has been considered

as either constant (in both cases) or as harmonic-varying in time (in the first case).

The literature examination first presented in Sect. 2 has revealed the most significant dynamic characteristics of the above-mentioned structural systems subjected to MLs. Thus, by outlining the distinctive traits of this topic, many different approaches have been presented, attempting to provide an overall picture concerning the investigation of ML problems, specifically for ML beam problems. Both analytical and numerical approaches have been overviewed, in order to gather the essential terminology and all the basic modeling principles and physical features of such mechanical systems.

In Sect. 3, the specific application of the FEM on ML structural problems has been reviewed, to obtain the structural matrices and vectors of a finite beam element, possibly introducing the contribution of modeling an underlying (visco)elastic foundation. In particular, two different types of elastic foundation have been considered: a linear and a nonlinear cubic one, the latter likely leading to a more accurate description of the real mechanical behaviour for railway lines. As a result, it has been possible to investigate the dynamic response of a railway track and its dependence upon the characteristic mechanical parameters of the system, and to validate the reliability of the implemented FEM program, by comparing the present results with reference results from the recent literature. In particular, an extensive campaign of numerical simulations has been carried out, to investigate the effects of variations of the velocity of the ML and of its magnitude frequency of oscillation. The FEM distributed computing formulation has been successfully implemented within a Unix-like environment. First of all, the code exploits a linear algebra library, Eigen, providing the needed data structures for assembling the governing matrices and vectors, and for the numerical solvers. In addition, the program interfaces with a GMSH tool, for the mesh generation and post-processing activities. At a later stage, the object-oriented C++ code has been overhauled, adopting a parallel implementation paradigm.

Section 4 has set the basics needed to develop a parallel algorithm implementation, by presenting three different kinds of a "divide and conquer" approach, toward taking advantage of distributed computing systems. After a summary review on the peculiarities of the existing computing architectures, the developed algorithm based on the "operator splitting" strategy has been described. Furthermore, the implemented code has indeed revealed to be able to exploit the computational resources provided by HPC distributed systems, such as GALILEO and MARCONI, two of the main clusters owned and available from Cineca, in Italy.

The parallel algorithm consists of five key stages: the creation of the mesh partitions, the allocation of the proper amount of memory to store the data sets, the assembly of the global distributed matrices and vectors, the solution of the governing algebraic system of linear or nonlinear equations and the post-processing phase. Coupled with the already mentioned packages, Eigen and GMSH, the developed parallel FEM program exploits the characteristics of three external packages. The pre-processing phase has been enhanced with METIS, a set of programs for partitioning FEM meshes. Open MPI has also been employed, to allow for the communication between the processes made available by distributed systems. Finally, PETSc has provided a series of tools to generate distributed matrices and vectors and to solve large sparse systems.

Thus, as presented in Sect. 5, the object-oriented C++ parallel code has demonstrated far better performances than for a previous sequential MatLab implementation, in terms of execution time, and achieved computational efficiency. Moreover, the code has proven to be well suited to run on distributed computing systems, by manifesting a satisfactory behaviour with regard to scalability. This should encourage further developments, in order to completely exploit the code potentialities by further accomplishing, through extensions and generalizations, multi-dimensional 2D or 3D FEM analyses involving several degrees of freedom.

Section 5 has presented complete parametric validation analyses on a beam-foundation system under ML. The foundation has been represented as a homogeneous Winkler elastic support, one of the most common existing mechanical models, while the behavior of the beam has been assumed according to classical Euler–Bernoulli theory for slender beams of a uniform cross-section. The dynamic transient response of a (long) finite, simply supported beam, lying on a linear and a nonlinear cubic elastic foundation, subjected to a concentrated ML traveling at a high constant velocity, with either a constant or harmonic-varying magnitude in time, has been considered. Then, its dynamic response has been investigated, by employing the above-mentioned homemade FEM implementation. The latter includes some routines dedicated to the numerical time integration, namely those relative to the HHT-α algorithm. The implemented FEM code has managed to numerically detect those characteristic velocities defined to be critical ("critical velocities"), leading to the largest displacements of the beam-foundation system. The reliability of the analyses and the quality of the recorded dynamic responses has been proven, by performing a consistent validation comparison, showing an adequate degree of agreement with results earlier reported in the very recent dedicated literature [19, 51, 54, 56].

The simulations performed within this stage of the work have offered useful insights on some physical characteristics of ML problems. First of all, the finite beam lying on an elastic foundation and subjected to a constant-magnitude ML has been taken into account. The effects of the linear or the nonlinear parameters defining the foundation constitutive law have been examined. For all the analyzed foundation

models, a sensible amplification of the beam response amplitude has been systematically detected, when the ML moves at the critical velocity. It has been identified that the value of the foundation stiffness affects the critical velocity. Furthermore, the damping provided by the beam-foundation system only induces a reduction of the recorded maximum and minimum displacements, basically without displacing the peak of the critical velocity.

Once the critical velocities have been proven to be fairly accurate in estimation for a constant-amplitude ML, a harmonic law has been considered, to describe a time-varying magnitude of the ML. Exciting the beam with a harmonic ML, with different mean values and frequencies, two further critical velocities have been detected from the dynamic response of the beam-foundation system. Thus, three peaks have been recorded for a harmonic ML with a non-zero mean magnitude, depending on both the frequency and the mean magnitude of the ML. The values of critical velocities tend to separate, as the loading frequency increases. Such findings shall display relevant implications in the modeling of the dynamic analysis of railway tracks, since the load harmonic amplitude variation may lead to a dangerous decrease of the lower critical velocity.

In Sect. 5, the preceding sequential MatLab coding and the present parallel object-oriented C++ implementation have been widely compared. It has been highlighted how the current FEM code has drastically reduced the execution time of each simulation. Thus, although the current implementation has still required a great amount of time, the vast numerical campaign has been completed in just some hours, instead as of several days. Furthermore, the reduction of the total time needed to carry out a simulation has not influenced the acquired accuracy of the results.

Section 6 has pertained to a simplified though convenient modelization of the dynamic response of the upper continuous beam of the historical railway riveted iron arch bridge at Paderno d'Adda (1889), as a 1D multi-span structure under a ML. The FEM model consists of a continuous beam lying on nine firm supports. Once the equivalent bending stiffness of the continuous beam has been estimated, a complete campaign of numerical tests has been carried out. Similarly to the previous analyses regarding railway tracks, simulations have been performed at a certain value of ML velocity, by recording the maximum upward and downward displacements, velocities and accelerations of the beam, given a certain value of the beam bending stiffness (sceneries of degraded or increased stiffness have been considered, with respect to the expected bending rigidity) and of the structural damping ratio (for zero or light inherent damping). Moreover, the time history of the system subjected to a moving locomotive has been illustrated, through a sequence of time frames, describing the evolution of the beam deflection due to the passage of the ML. Finally, the recorded displacements, velocities and accelerations at the middle point of the fifth span of the beam, corresponding to the keystone of the underlying parabolic arch of the bridge have been displayed, in order to show how the implemented parallel code successfully managed to generate information that could be employed toward assessing the current dynamic behaviour of the structure, in view of possible Structural Health Monitoring strategies, devoted to conservation and possible intervention.

The combination of the new C++ parallel implementation with the mechanical and engineering implications derived from the developed ML numerical tests, with all the competent literature framing, shall consistently show that the present approach reveals to be rather feasible, toward inspecting the physical response of such challenging dynamic systems. In contrast with a previous sequential MatLab code, the present C++ parallel implementation manages to exploit the computational resources made available by clusters and other kinds of distributed systems, specifically in the context of HPC machines. Although some implemented C++ routines could still be refined and further dedicated to given specific needs, the present parallel algorithm shall represent a fundamental computational tool, allowing to deepen the understanding of ML problems, extending then the analysis to even more realistic situations, such as those involving complex multi-dimensional structures.

Finally, the mentioned parallel implementation shall also be considered as an important starting point for further enlargements toward achieving a more comprehensive computer code toward FEM analysis, independently from the specific field of application (here ML dynamics). In fact, the parallel algorithm turns out to be extremely customizable, giving users the possibility to implement additional features and computational routines, by employing the already developed main implant.

7.1 Future Developments

This study shall contribute in referencing a fundamental starting point to enhance the understanding of the dynamic behaviour of the considered mechanical problems, which could be roughly classified as ML dynamical problems, such as concerning high-speed railway tracks or multi-span bridges under traffic loads. Future studies on the current subjects could be considered, in order to provide further refined and detailed modelizations, to achieve an even more realistic dynamic response prediction of the mentioned systems, specifically in practical terms.

Regarding the analyses of railway tracks, real applications usually require the extension to an infinite beam length. Thus, it is necessary to avoid the effects of the beam supports, by preventing the perturbations induced by the spurious reflections of the traveling waves at the ends of a finite

modelled beam. This aim may be achieved by introducing enhanced boundary conditions, which should be able to absorb the incoming waves at the extremes of the beam (see e.g. Froio et al. [57, 58]). Furthermore, a better characterization of the track behaviour may be obtained, for instance by involving a more detailed representation of the underlying foundation, as a continuum, not implemented yet in the current version of the C++ parallel code, for not saying about the modelization of the moving object and the local interaction with the structure and the foundation.

Furthermore, the simplified 1D multi-span FEM model of the upper beam of the Paderno d'Adda Bridge could be improved by the extension to a true 3D beam model, whose computational effort of analysis may be held up by the parallel implementation provided within this work. Same for the whole 3D discretization of the complete structure of the bridge [46]. This may allow to undertake possible model-updating processes, may be based on experimental evidences linked to present-state conditions of the infrastructure. It would require the development of new finite elements, an innovative way to manage the ML path and the implementation of a database to collect and query the mechanical properties of all types of beam cross-section. Further investigations on the effects induced by the time and space discretizations and the time integration parameters should be carried out, to analyze possible spurious oscillations of the solution that may come out from the dynamic computation.

Both situations may also require more truthful vehicle models, in order to replace the single moving concentrated force. Firstly, trains may be represented as a sequence of concentrated MLs, but then, the vehicle-structure interaction could also be described by involving moving spring-mass-damper systems. Moreover, the vibration produced by the irregularities of the railway tracks and the wheel flats could be taken into account, to succeed in realistically describing the examined mechanical systems.

Nonetheless, the present work shall serve as a fundamental basis for future wider studies aimed at collecting useful data, such as critical velocities and vibration amplitudes, which could be employed in the investigated challenging structural dynamics vibration context and at a design or evaluation stage of structures and infrastructures subjected to MLs, at increasing speeds of the considered moving objects.

Acknowledgements Open access funding provided by Università degli studi di Bergamo within the CRUI-CARE Agreement. The authors acknowledge public research support from "Fondi di Ricerca d'Ateneo ex 60%" and a former ministerial doctoral Grant and funds at the ISA Doctoral School, University of Bergamo. Access to HPC resources at Cineca, Italy, by the University of Bergamo is gratefully acknowledged.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Abu Hilal M, Mohsen M (2000) Vibration of beams with general boundary conditions due to a moving harmonic load. J Sound Vib 232(4):703–717
2. Abu Hilal M, Zibdeh HS (2000) Vibration analysis of beams with general boundary conditions traversed by a moving force. J Sound Vib 229(2):377–388
3. Adam C, Di Lorenzo S, Failla G, Pirrotta A (2017) On the moving load problem in beam structures equipped with tuned mass dampers. Meccanica 52(13):3101–3115
4. Amdahl GM (1967) Validity of the single processor approach to achieving large scale computing capabilities. In: Proceedings of the April 18–20, 1967, Spring joint computer conference, AFIPS '67 (Spring). ACM, New York, NY, USA, pp 483–485. https://doi.org/10.1145/1465482.1465560
5. Amestoy PR, Duff IS, L'Excellent J-Y (2000) Multifrontal parallel distributed symmetric and unsymmetric solvers. Comput Methods Appl Mech Eng 184(2):501–520
6. Amiri SN, Onyango M (2010) Simply supported beam response on elastic foundation carrying loads. J Eng Sci Technol 5(1):52–66
7. Ansari M, Esmailzadeh E, Younesian D (2010) Internal–external resonance of beams on non-linear viscoelastic foundation traversed by moving load. Nonlinear Dyn 61(1):163–182
8. Argyris JH (2013) Energy theorems and structural analysis: a generalised discourse with applications on energy principles of structural analysis including the effects of temperature and non-linear stress–strain relations. Springer, Berlin
9. Arno B (2017) LatexDraw web page. http://latexdraw.sourceforge.net
10. Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, Dalcin L, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, May DA, McInnes LC, Rupp K, Smith BF, Zampini S, Zhang H, Zhang H (2017) PETSc web page. http://www.mcs.anl.gov/petsc
11. Basu D, Kameswara Rao NSV (2013) Analytical solutions for Euler–Bernoulli beam on visco-elastic foundation subjected to moving load. Int J Numer Anal Methods Geomech 37(8):945–960
12. Bathe KJ (2006) Finite element procedures, I edn. Prentice-Hall, Upper Saddle River
13. Beskou ND, Theodorakopoulos DD (2011) Dynamic effects of moving loads on road pavements: a review. Soil Dyn Earthq Eng 31(4):547–567

14. Bezanson J, Edelman A, Karpinski S, Shah VB (2014) Julia: a fresh approach to numerical computing. CoRR, arXiv:1411.1607
15. Bogacz R, Krzyỳski T, Popp K (1989) On the generalization of Mathews problem of the vibrations of a beam on elastic foundation. Z Angew Math Mech (ZAMM) 69(8):243–252
16. Bogaert PV (2008) Experimental verifications of vibration—and fatigue conduct of viaducts and large span bridges, crossed by high-speed trains. Department of Civil Engineering, Ghent University, Ghent
17. Bonnet M, Frangi A, Rey C (2014) The finite element method in solid mechanics. McGraw-Hill Education, New York
18. Castro Jorge P, Pinto da Costa A, Simões FMF (2015) Finite element dynamic analysis of finite beams on a bilinear foundation under a moving load. J Sound Vib 346(23 June 2015):328–344
19. Castro Jorge P, Simões FMF, Pinto da Costa A (2015) Dynamics of beams on non-uniform nonlinear foundations to moving loads. Comput Struct 148(February 2015):26–34
20. CEN (2002) Eurocode 1: actions on structures—part 2: traffic loads on bridges. European Committee for Standardization. Final draft prEN 1991–2
21. Chang SY (2008) Performance of the HHT-α method for the solution of nonlinear systems. Int J Struct Stab Dyn 8(2):321–337
22. Chang TP, Liu YN (1996) Dynamic finite element analysis of a nonlinear beam subjected to a moving load. Int J Solids Struct 33(12):1673–1688
23. Chen YH, Huang YH (2003) Dynamic characteristics of infinite and finite railways to moving loads. J Eng Mech ASCE 129(9):987–995
24. Chen YH, Huang YH, Shih CT (1987) General dynamic-stiffness matrix of a Timoshenko beam for transverse vibrations. Earthq Eng Struct Dyn 15(3):391–402
25. Chen YH, Huang YH, Shih CT (2001) Response of an infinite Timoshenko beam on a viscoelastic foundation to a harmonic moving load. J Sound Vib 241(5):809–824
26. Chonan S (1978) Moving harmonic load on an elastically supported Timoshenko beam. Z Angew Math Mech (ZAMM) 58(1):9–15
27. Clough RW (2004) Early history of the finite element method from the view point of a pioneer. Int J Numer Methods Eng 60(1):283–287
28. Cook RD, Malkus DS, Plesha ME, Witt RJ (2001) Concepts and applications of finite element analysis, 4th edn. Wiley, New York
29. D-214 Committee (1999) Rail bridges for speeds over 200 km/h. European Rail Research Institute (ERRI). RP 9, Final report, Utretch, The Netherlands
30. Davis TA (2006) Direct methods for sparse linear systems. Fundamentals of algorithms. Society for Industrial Mathematics, Philadelphia
31. Delgado R, Calçada R, Goicolea JM, Gabaldón F (2008) Dynamics of high-speed railway bridges: selected and revised papers from the advanced course on dynamics of high-speed railway bridges, 1st edn. Taylor and Francis, Porto
32. Di Lorenzo S, Di Paola M, Pirrotta A (2017) On the moving load problem in Euler–Bernoulli uniform beams with viscoelastic supports and joints. Acta Mech 228(3):805–821
33. Dimitrovová Z (2010) A general procedure for the dynamic analysis of finite and infinite beams on piece-wise homogeneous foundation under moving loads. J Sound Vib 329(13):2635–2653
34. Dimitrovová Z (2016) Critical velocity of a uniformly moving load on a beam supported by a finite depth foundation. J Sound Vib 366(31 March 2016):325–342
35. Dimitrovová Z (2017) New semi-analytical solution for a uniformly moving mass on a beam on a two-parameter visco-elastic foundation. Int J Mech Sci 127(July 2017):142–162
36. Dimitrovová Z (2018) Complete semi-analytical solution for a uniformly moving mass on a beam on a two-parameter visco-elastic foundation with non-homogeneous initial conditions. Int J Mech Sci 144(August 2018):283–311
37. Dimitrovová Z (2019) Semi-analytical solution for a problem of a uniformly moving oscillator on an infinite beam on a two-parameter visco-elastic foundation. J Sound Vib 438(January 2019):257–290
38. Dimitrovová Z, Rodrigues AFS (2012) Critical velocity of a uniformly moving load. Adv Eng Softw 50(1):44–56
39. Dimitrovová Z, Varandas JN (2009) Critical velocity of a load moving on a beam with a sudden change of foundation stiffness: applications to highspeed trains. Comput Struct 87(19):1224–1232
40. Ding H, Chen LQ, Yang SP (2012) Convergence of Galerkin truncation for dynamic response of finite beams on nonlinear foundations under a moving load. J Sound Vib 331(10):2426–2442
41. Dupros F, De Martin F, Foerster FE, Komatitsch D, Roman J (2010) High-performance finite-element simulations of seismic wave propagation in three-dimensional nonlinear inelastic geological media. Parallel Comput 36(5):308–325
42. Fangohr H (2004) A comparison of C, MATLAB, and Python as teaching languages in engineering. Springer, Berlin, pp 1210–1217
43. Felippa CA (2004) Introduction to finite element methods–course lecture notes. Department of Aerospace Engineering Sciences and Center for Aerospace Structures, University of Colorado at Boulder, Boulder
44. Ferrari R, Cocchetti G, Rizzi E (2016) Limit analysis of a historical iron arch bridge. Formulation and computational implementation. Comput Struct 175(15 October 2016):184–196. https://doi.org/10.1016/j.compstruc.2016.05.007
45. Ferrari R, Cocchetti G, Rizzi E (2018) Computational elastoplastic limit analysis of the Paderno d'Adda bridge (Italy, 1889). Arch Civ Mech Eng 18(1):291–310. https://doi.org/10.1016/j.acme.2017.05.002
46. Ferrari R, Cocchetti G, Rizzi E (2019) Reference structural investigation on a 19th-century arch iron bridge loyal to design-stage conditions. Int J Archit Heritage. 05 July 2019:1–31. https://doi.org/10.1080/15583058.2019.1613453
47. Ferrari R, Rizzi E (2008) On the theory of the ellipse of elasticity as a natural discretisation method in the design of Paderno d'Adda Bridge (Italy). In: D'Ayala and Fodde (eds) Structural analysis of historic construction: preserving safety and significance. ISBN: 978-1-4398-2822-9, 2–4 July 2008, Bath, UK, pp 583–591. https://doi.org/10.1201/9781439828229.ch66
48. Ferrovie dello Stato Italiano (1997) Sovraccarichi per il collaudo dei ponti, FF/SS
49. Foyouzat MA, Mofid M, Akin JE (2016) On the dynamic response of beams on elastic foundations with variable modulus. Acta Mech 227(2):549–564
50. Froio D (2018) Structural dynamics modelization of one-dimensional elements on elastic foundations under fast moving load. In: Doctoral thesis in engineering and applied sciences, Advisor E. Rizzi, Co-Advisor F.M.F. Simões. Università degli studi di Bergamo, p 233. https://doi.org/10.6092/TDUnibg.105179
51. Froio D, Moioli R, Rizzi E (2016) Numerical dynamical analysis of beams on nonlinear elastic foundations under harmonic moving load. In: Proceedings of the VII European congress on computational methods in applied sciences and engineering (ECCOMAS2016), vol 3, ISBN: 978-618-82844-0-1, 5–10 June 2016, Crete Island, Greece, pp 4794–4809. https://doi.org/10.7712/100016.2149.7515
52. Froio D, Rizzi E (2015) Analytical solution for the elastic bending of beams lying on a variable Winkler support. Acta Mech 227(4):1157–1179. https://doi.org/10.1007/s00707-015-1508-y
53. Froio D, Rizzi E (2017) Analytical solution for the elastic bending of beams lying on a linearly variable Winkler support.

Int J Mech Sci 128–129(August 2017):680–694. https://doi.org/10.1016/j.ijmecsci.2017.04.021
54. Froio D, Rizzi E, Simões FMF, Pinto da Costa A (2017) Critical velocities of a beam on nonlinear elastic foundation under harmonic moving load. Proc Eng 199:2585–2590. Special issue on the X international conference on structural dynamics (EURODYN2017), 10–13 September, Rome, Italy. https://doi.org/10.1016/j.proeng.2017.09.348
55. Froio D, Rizzi E, Simões FMF, Pinto da Costa A (2017) Universal analytical solution of the steady-state response of an infinite beam on a Pasternak elastic foundation under moving load. Int J Solids Struct 132–133(February 2018):245–263. https://doi.org/10.1016/j.ijsolstr.2017.10.005
56. Froio D, Rizzi E, Simões FMF, Pinto da Costa A (2018) Dynamics of a beam on a bilinear elastic foundation under harmonic moving load. Acta Mech 229(10):4141–4165. https://doi.org/10.1007/s00707-018-2213-4
57. Froio D, Rizzi E, Simões FMF, Pinto da Costa A (2020) DLSFEM–PML formulation for the steady-state response of a taut string on visco-elastic support under moving load. Meccanica. Special issue on computational models for 'complex' materials and structures, beyond the finite elements 55(4 April 2020):765–790. https://doi.org/10.1007/s11012-019-01047-7
58. Froio D, Rizzi E, Simões FMF, Pinto da Costa A (2020) A true PML approach for steady-state vibration analysis of an elastically supported beam under moving load by a DLSFEM formulation. Comput Struct 239(15 October 2020):106295. https://doi.org/10.1016/j.compstruc.2020.106295
59. Frýba L (1972) Vibration of solids and structures under moving loads, III edn. Academia, Academy of Sciences of the Czech Republic, Prague
60. Fărăgău AB, Metrikine AV, van Dalen KN (2019) Transition radiation in a piecewise-linear and infinite one-dimensional structure–a Laplace transform method. Nonlinear Dyn. https://doi.org/10.1007/s11071-019-05083-6
61. Gabaldón Castillo F, Riquelme F, Goicolea-Ruigómez JM, Arribas JJ (2008) Dynamic analysis of structures under high speed train loads: case studies in Spain. In: Dynamics of high-speed railway bridges, pp 143–165
62. Gaël G, Benoît J et al (2010) Eigen v3. http://eigen.tuxfamily.org
63. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness, 1st edn. Series of books in the mathematical sciences. W.H. Freeman, New York
64. GCC (2017) G++. https://www.gnu.org/software/gcc/index.html
65. Gentile C, Saisi A (2011) Ambient vibration testing and condition assessment of the Paderno iron arch bridge (1889). Constr Build Mater 25(9):3709–3720
66. Geradin M, Hogge M, Idelsohn S (1983) Implicit finite element methods. In: Belytschko T, Hughes TJR (eds) Computational methods for transient analysis, chapter 9. North-Holland, Amsterdam, pp 417–470
67. Geuzaine C, Remacle J-F (2009) A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. Int J Numer Methods Eng 79(11):1309–1331
68. Goicolea-Ruigómez JM, Gabaldón Castillo F (2008) Design issues related to dynamic effects for high speed railway bridges in Spain. In: Dynamics of high-speed railway bridges, pp 13–24
69. Grama A, Karypis G, Kumar V, Gupta A (2003) Introduction to parallel computing, 2nd edn. Addison Wesley, Boston
70. Gropp B, Lusk R (2017) MPICH web page. http://www.mpich.org
71. Gropp W, Lusk E, Skjellum A (1999) Using MPI: portable parallel programming with the message passing interface. Scientific and engineering computation, 2nd edn. The MIT Press, Cambridge
72. Gropp W, Lusk E, Thakur R (1999) Using MPI-2: advanced features of the message passing interface. Scientific and engineering computation, 1st edn. The MIT Press, Cambridge
73. Gupta KK, Meek JL (1996) A brief history of the beginning of the finite element method. Int J Numer Methods Eng 39(22):3761–3774
74. Heister T, Kronbichler M, Bangerth W (2010) Massively parallel finite element programming. Springer, Berlin, pp 122–131
75. Henchi K, Fafard M, Dhatt G, Talbot M (1997) Dynamic behaviour of multi-span beams under moving loads. J Sound Vib 199(1):33–50
76. Hendrickson B, Leland R (1995) An improved spectral graph partitioning algorithm for mapping parallel computations. SIAM J Sci Comput 16(2):452–469
77. Hendrickson B, Leland R (1995) A multilevel algorithm for partitioning graphs. In: Proceedings of the 1995 ACM/IEEE conference on supercomputing, supercomputing '95. ACM, New York, NY, USA. https://doi.org/10.1145/224170.224228
78. Hénon P, Ramet P, Roman J PaStiX web page. https://gforge.inria.fr/projects/pastix
79. Hénon P, Ramet P, Roman J (2002) Pastix: a high-performance parallel direct solver for sparse symmetric positive definite systems. Parallel Comput 28(2):301–321
80. Hilber HM, Hughes TJR, Taylor RL (1977) Improved numerical dissipation for time integration algorithms in structural dynamics. Earthq Eng Struct Dyn 3(10):283–292
81. Hill MD (1990) What is scalability? SIGARCH Comput Archit News 18(4):18–21
82. Hoorpah W (2008) Dynamic calculations of high-speed railway bridges in France—some case studies. MIO, Paris
83. Hughes TJR (2000) The finite element method: linear static and dynamic finite element analysis. Dover Civil and Mechanical Engineering, Dover Publications, New York
84. Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9(3):90–95
85. Inglis CE (1934) A mathematical treatise on vibration in railway bridges. The University Press, Cambridge
86. ISOCPP (2017) The C++ standards committee. http://www.open-std.org/jtc1/sc22/wg21
87. Iwankiewicz R, Sniady P (1984) Vibration of a beam under a random stream of moving forces. J Struct Mech 12(1):13–26
88. Janzen R (2017) Transpod ultra-high-speed tube transportation: dynamics of vehicles and infrastructure. Proc Eng 199(Supplement C):8–17
89. Jeffcott HH (1929) On the vibration of beams under the action of moving loads. Philos Mag 8(48):66–97
90. Johansson C, Pacoste C, Karoumi R (2013) Closed-form solution for the mode superposition analysis of the vibration in multi-span beam bridges caused by concentrated moving loads. Comput Struct 119(Supplement C):85–94
91. Ju SH, Lin HT (2003) Resonance characteristics of high-speed trains passing simply supported bridges. J Sound Vib 267(5):1127–1141
92. Karypis G, Kumar V (1999) A fast and highly quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20(1):359–392
93. Keijo R (2013) Lecture notes in graph theory, February 2013
94. Kenney JT Jr (1954) Steady-state vibrations of beams on elastic foundations for moving load. J Appl Mech ASME 21(4):359–364
95. Kien ND, Hai TT (2006) Dynamic response of prestressed Bernoulli beams resting on two-parameter foundation under moving harmonic load. Vietnam J Mech 28(3):176–188
96. Kien ND, Hai TT (2008) Dynamic response of prestressed Timoshenko beams resting on two-parameter foundation to moving harmonic load. Tech Mech 24(3–4):237–258

97. Knepley MG (2017) Lecture notes in computational science, November 2017
98. Kolousek V (1956) Dynamics of civil enineering structures. SNTL, Berlin
99. Krylov AN (1905) Mathematical collection of papers of the academy of sciences, Petersburg
100. Kumar CPS, Sujatha C, Shankar K (2015) Vibration of simply supported beams under a single moving load: a detailed study of cancellation phenomenon. Int J Mech Sci 99(August 2015):40–47
101. Kumar VP, Gupta A (1994) Analyzing scalability of parallel algorithms and architectures. J Parallel Distrib Comput 22(3):379–391
102. Lowan AN (1935) On transverse oscillations of beams under the action of moving variable loads. Philos Mag 19(127):708–715
103. Lu S, Deng X (1998) Dynamic analysis to infinite beam under a moving line load with uniform velocity. Appl Math Mech 19(4):367–373
104. Mallik AK, Chandra S, Singh AB (2006) Steady-state response of an elastically supported infinite beam to a moving load. J Sound Vib 291(3):1148–1169
105. Martínez-Castro AE, Museros P, Castillo-Linares A (2006) Semi-analytic solution in the time domain for non-uniform multi-span Bernoulli–Euler beams traversed by moving loads. J Sound Vib 294(1):278–297
106. Mathews PM (1958) Vibrations of a beam on elastic foundation. Z Angew Math Mech (ZAMM) 38(3–4):105–115
107. Mathews PM (1959) Vibrations of a beam on elastic foundation II. Z Angew Math Mech (ZAMM) 39(1–2):13–19
108. Mazilu T (2010) Interaction between a moving two-mass oscillator and an infinite homogeneous structure: Green's functions method. Arch Appl Mech 80(8):909–927
109. Mazilu T (2013) Instability of a train of oscillators moving along a beam on a viscoelastic foundation. J Sound Vib 332(19):4597–4619
110. Metrikine A, Dieterman H (1997) Instability of vibrations of a mass moving uniformly along an axially compressed beam on a viscoelastic foundation. J Sound Vib 201(5):567–576
111. Metrikine A, Verichev S (2001) Instability of vibrations of a moving two-mass oscillator on a flexibly supported Timoshenko beam. Arch Appl Mech 71(9):613–624
112. Microsoft (2017) Visual C++. https://docs.microsoft.com/en-us/cpp
113. Miller GL, Teng SH, Vavasis SA (1991) A unified geometric approach to graph separators. In: Proceedings of the 32nd annual symposium on foundations of computer science, SFCS '91, pp 538–547. IEEE Computer Society, Washington, DC, USA. https://doi.org/10.1109/SFCS.1991.185417
114. Moore GE (1965) Cramming more components onto integrated circuits. Electronics 38(8):114–120
115. MPI Forum (2017) Open MPI web page. https://www.open-mpi.org
116. Museros P, Moliner E, Martínez-Rodrigo MD (2013) Free vibrations of simply-supported beam bridges under moving loads: maximum resonance, cancellation and resonant vertical acceleration. J Sound Vib 332(2):326–345
117. Museros P, Romero ML, Poy A, Alarcón E (2002) Advances in the analysis of short span railway bridges for high-speed lines. Comput Struct 80(27):2121–2132
118. Nascè V, Zorgno AM, Bertolini C, Carbone VI, Pistone G, Roccati R (1984) Il ponte di Paderno: storia e struttura—conservazione dell'architettura in ferro. Restauro, Anno XII I(73–74):1–215
119. Noor AK (1988) Parallel processsing in finite flement structural analysis. Eng Comput 3(4):225–241
120. Olsson M (1985) Finite element, modal co-ordinate analysis of structures subjected to moving loads. J Sound Vib 99(1):1–12
121. Olsson M (1991) On the fundamental moving load problem. J Sound Vib 145(2):299–307
122. Ousterhout JK (1998) Scripting: higher-level programming for the 21st century. Computer 31(3):23–30. https://doi.org/10.1109/2.660187
123. Ouyang H (2011) Moving-load dynamic problems: a tutorial (with a brief overview). Mech Syst Signal Process 25(6):2039–2060
124. Papadimitriou CH, Steiglitz K (1998) Combinatorial optimization: algorithms and complexity, unabridged edn. Dover Publications, New York
125. Pavlovic MN, Wylie GB (1983) Vibration of beams on non-homogeneous elastic foundations. Earthq Eng Struct Dyn 11(6):797–808
126. Pioldi F, Salvi J, Rizzi E (2017) Refined FDD modal dynamic identification from earthquake responses with soil-structure interaction. Int J Mech Sci 127(July 2017):47–61. https://doi.org/10.1016/j.ijmecsci.2016.10.032
127. Pothen A, Horst DS, Liou KP (1990) Partitioning sparse matrices with eigenvectors of graphs. SIAM J Matrix Anal Appl 11(3):430–452
128. Prechelt L (2000) An empirical comparison of seven programming languages. Computer 33(10):23–29. https://doi.org/10.1109/2.876288
129. Ramondenc P (2008) Dynamic behaviour of rail bridges: the train excitation. SNCF Engineering Department, Saint Denis
130. Rieker JR, Lin YH, Trethewey MW (1996) Discretization considerations in moving load finite element beam models. Finite Elem Anal Des 21(3):129–144
131. Rodrigues C, Simões FMF, Pinto da Costa A, Froio D, Rizzi E (2018) Finite element dynamic analysis of beams on nonlinear elastic foundations under a moving oscillator. Eur J Mech A/Solids 68(March–April 2018):9–24. https://doi.org/10.1016/j.euromechsol.2017.10.005
132. Salcher P, Adam C (2015) Modeling of dynamic train-bridge interaction in high-speed railways. Acta Mech 226(8):2473–2495
133. Salvi J, Pioldi F, Rizzi E (2018) Optimum tuned mass dampers under seismic soil-structure interaction. Soil Dyn Earthq Eng 114(November 2018):576–597. https://doi.org/10.1016/j.soildyn.2018.07.014
134. Salvi J, Rizzi E (2015) Optimum tuning of tuned mass dampers for frame structures under earthquake excitation. Struct Control Health Monit 22(4):707–725. https://doi.org/10.1002/stc.1710
135. Salvi J, Rizzi E (2016) Closed-form optimum tuning formulas for passive tuned mass dampers under benchmark excitations. Smart Struct Syst 17(2):231–256. https://doi.org/10.12989/sss.2016.17.2.231
136. Salvi J, Rizzi E (2017) Optimum earthquake-tuned TMDs: seismic performance and new design concept of balance of split effective modal masses. Soil Dyn Earthq Eng 101(October 2017):67–80. https://doi.org/10.1016/j.soildyn.2017.05.029
137. Salvi J, Rizzi E, Rustighi E, Ferguson NS (2015) On the optimization of a hybrid tuned mass damper for impulse loading. Smart Mater Struct 24(8):085010. https://doi.org/10.1088/0964-1726/24/8/085010
138. Salvi J, Rizzi E, Rustighi E, Ferguson NS (2018) Optimum tuning of passive tuned mass dampers for the mitigation of pulse-like responses. J Vib Acoust 140(6):061014. https://doi.org/10.1115/1.4040475
139. Sanches R, Simões FMF, Pinto da Costa A (2020) Physical and geometrical nonlinear dynamic analysis of beams on foundations under moving loads. J Eng Mech 146(1):04019114
140. Selim GA (1989) The design and analysis of parallel algorithms, 1st edn. Prentice Hall, Upper Saddle River
141. Snir M, Gropp W (1998) MPI: the complete reference, vol 1, 2nd edn. The MIT Press, Cambridge

142. Società Nazionale delle Officine di Savigliano (1889) Viadotto di Paderno sull'Adda (Ferrovia Ponte S. Pietro-Seregno). Torino: Tip. e Lit. Camilla e Bertolero
143. Steele CR (1967) The finite beam with a moving load. J Appl Mech ASME 34(1):111–118
144. Stokes GG (1849) Discussion of a differential equation relating to the breaking of railway bridges. Transactions of the Cambridge Philosophical Society, Cambridge
145. Thambiratnam D, Zhuge Y (1996) Dynamic analysis of beams on an elastic foundation subjected to moving load. J Sound Vib 198(2):149–169
146. The MathWorks, Inc. (2016) MatLab. Available online at http://www.mathworks.com/products/matlab
147. Timoshenko SP (1908) Forced vibrations of prismatic bars. Izv Kiev Politekh Inst 59:163–203
148. Timoshenko SP (1927) Method of analysis of statical and dynamical stresses in rail. In: Proceedings of the 2nd international congress for applied mechanics, vol 54, pp 1–12
149. Timoshenko SP (1953) History of strength of materials—with a brief account of the history of theory of elasticity and theory of structures. Dover Publications, New York
150. TOP500 Supercomputer web site. https://www.top500.org
151. Toscano Corrêa R, Pinto da Costa A, Simões FMF (2018) Finite element modeling of a rail resting on a Winkler-Coulomb foundation and subjected to a moving concentrated load. Int J Mech Sci 140 (May 2018):432–445. https://doi.org/10.1016/j.ijmecsci.2018.03.022
152. Toscano Corrêa R, Simões FMF, Pinto da Costa A (2017) Moving loads on beams on Winkler foundations with passive frictional damping devices. Eng Struct 152 (1 December 2017):211–225. https://doi.org/10.1016/j.engstruct.2017.09.023
153. Turner MJ, Clough RW, Martin HC, Topp LJ (1956) Stiffness and deflection analysis of complex structures. J Aeronaut Sci 23(9):805–854
154. Ullah Z, Coombs W, Augarde C (2016) Parallel computations in nonlinear solid mechanics using adaptive finite element and meshless methods. Eng Comput 33(4):1161–1191
155. Wang YH, Tham LG, Cheung YK (2005) Beams and plates on elastic foundation: a review. Prog Struct Eng Mater 7(4):174–182
156. Willis R (1849) Report of the commissioners appointed to inquire into the application of iron to railway structures
157. Winkler E (1867) Die Lehre von der Elastizität und Festigkeit. Verlag H. Dominicus, Prague
158. Wu JJ, Whittaker AR, Cartmell MP (2000) The use of finite element techniques for calculating the dynamic response of structures to moving loads. Comput Struct 78(6):789–799. https://doi.org/10.1016/S0045-7949(00)00055-9
159. Xia H, Li HL, Guo WW, De Roeck G (2014) Vibration resonance and cancellation of simply supported bridges under moving train loads. J Eng Mech 140(5):1–11
160. Yoshimura S, Yagawa G, Soneda N (1991) A large scale finite element analysis using domain decomposition method on a parallel computer. Comput Struct 38(5):615–625
161. Yang YB, Yau JD, Hsu LC (1997) Vibration of simple beams due to trains moving at high speeds. Eng Struct 19(11):936–944
162. Yau JD, Yang YB (2006) Vertical accelerations of simple beams due to successive loads traveling at resonant speeds. J Sound Vib 289(1):210–228
163. Yoshida DM, Weaver W (1971) Finite element analysis of beams and plates with moving loads. Publ Int Assoc Bridge Struct Eng 31(1):179–195
164. Zhang J (2016) Class slides of parallel and distributed computation, Spring 2016
165. Zienkiewicz OC, Taylor RL (2000) The finite element method, vol 1, 5th edn. Butterworth-Heinemann, Oxford