Facultad
de
Ciencias

Simulación de líneas de fondeo con elasto-plasticidad mediante elementos finitos

(Simulation of elasto-plasticity in mooring lines with finite elements)

Trabajo de Fin de Grado
para acceder al

GRADO EN MATEMÁTICAS

Author: Mar González Alonso
Director: Jose Javier Segura Sala
Codirector: Álvaro Rodríguez Luis
16 de junio de 2025

Este proyecto pone fin a cinco grandes años de aprendizaje y descubrimientos, donde he entablado grandes amistades con gente maravillosa como David, Nadir y Sara. Es más, este trabajo no habría tenido lugar si en Trondheim mis compañeros Ole Gunnar y Alexander no me hubieran hecho ver la belleza de los métodos numéricos y sus aplicaciones.

Quiero dar las gracias a aquellos que me acogieron en el IHCantabria, y en especial a Álvaro, por su paciencia y guía durante tantos meses.

También a Ale, por estar en las buenas y en las malas, por darme los mejores consejos cuando más los necesito.

Abstract

This Final Degree Project focuses on the mathematical modeling and simulation of mooring lines subjected to dynamic conditions in marine environments. The main objective is to develop and implement a computational model capable of simulating the behavior of these lines while accounting for their elasto-plastic behavior, which introduces significant mathematical and computational challenges.

The study begins with Newton's second law equations expressed as nonlinear partial differential equations (PDEs), which are discretized using the Finite Element Method (FEM) and time integration methods such as the second order Backward Differentiation Formula (BDF2) with adaptive time-stepping. The work covers both the mathematical foundations (Sobolev spaces, weak formulations) and the development of the physical model, including its implementation within the inhouse model developed at IHCantabria, highlighting the complexity of its code and the integration of the new tension law. The main contribution lies in the incorporation of a new tension term that more accurately models the elasto-plastic behavior. The project includes verification with experimental and theoretical data from works found in the literature and validation using catenary-type solutions.

Keywords: mooring lines, mathematical modeling, nonlinear partial differential equations, Finite Element Method (FEM), second order Backward Differentiation Formula (BDF2), weak formulation, elasto-plasticity.

Resumen

Este Trabajo de Fin de Grado se centra en la modelización y simulación matemática de líneas de fondeo sometidas a condiciones dinámicas en entornos marinos. El objetivo principal es desarrollar e implementar un modelo computacional que permita simular el comportamiento de estas líneas considerando su comportamiento elasto-plástico, lo que introduce importantes retos matemáticos y computacionales.
Se parte de las ecuaciones de la segunda ley de Newton en forma de ecuaciones en derivadas parciales (EDPs) no lineales, que se discretizan utilizando el método de elementos finitos (FEM) y métodos de integración temporal como BDF2 con paso adaptativo. El estudio abarca desde los fundamentos matemáticos (espacios de Sobolev, formulaciones débiles) hasta el desarrollo del modelo físico y su implementación dentro del modelo desarrollado en el IHCantabria, destacando la complejidad del código y la integración de una nueva ley de tensión. La contribución principal radica en la incorporación de un nuevo término de tensión que modela el comportamiento elasto-plástico de forma más precisa. El trabajo incluye una verificación con datos experimentales y teóricos de trabajos encontrados en la literatura y una validación con soluciones tipo catenaria.
Palabras clave: líneas de fondeo, modelización matemática, Ecuaciones en Derivadas Parciales no lineales, Elementos finitos (FEM), BDF2, formulación débil, elasto-plasticidad

Contents

1 Introduction

1.1 Motivation

Partial differential equations (PDEs) play a central role in modeling phenomena in physics and engineering, yet their analytical solutions are limited to idealized cases. When dealing with real-world systems—particularly those involving nonlinearities, complex boundary conditions, or coupled physical processes—numerical methods become indispensable. The study and development of such methods, including the finite element method (FEM) and advanced time-integration schemes, lie at the heart of modern applied mathematics.

This work focuses on the mathematical modeling and simulation of mooring lines subjected to dynamic marine environments. While the physical context is engineering-oriented, the mathematical challenges are fundamental: modeling wave propagation through nonlinear PDEs, deriving weak formulations suitable for FEM, and handling the temporal evolution of highly nonlinear systems through stiff ordinary differential equations (ODEs).

Moreover, the inclusion of elasto-plastic behavior introduces further complexity, requiring careful treatment of material laws within the numerical scheme. The accurate and stable integration of such models not only advances engineering applications but also contributes to the broader field of computational mathematics by demonstrating how theory—functional analysis, variational methods, numerical stability—translates into algorithms.

This work aims to explore these challenges from a mathematical perspective, highlighting the structure, analysis, and implementation of numerical schemes that bridge theoretical models and computational practice. The implementation of the new model was developed in OASIS, an inhose model developed at IHCantabria for the simulation of floating bodies coupled with mooring lines among other physical systems. Prior work has been carried out before the current project focusing on other parts of the model [1] [2].

1.2 Objectives

The primary objective of this work is to develop, implement, and validate a mathematical and computational model capable of simulating mooring lines under nonlinear conditions, particularly elasto-plastic behavior. To achieve this, the following specific goals are pursued:

- Review the necessary mathematical tools used in the model.
- Explain and analyze the model developed at IHCantabria.
- Couple a elasto-platicity model.
- Test the proper functioning of the implementation.

1.3 Document organization

This document is structured as follows:
Section 2: Theoretical framework. Provides the mathematical background necessary

for understanding the numerical methods employed. It covers functional analysis, partial differential equations, finite element methods, and temporal integration schemes.
Section 3: Description of the Model. Details the physical and mathematical modeling of the mooring lines. It introduces the governing equations, boundary and initial conditions, spatial and temporal discretization, and the weak formulation used in the finite element analysis.
Section 4: Contributions to the model. Focuses on the development and implementation of new model features, such as the elasto-plastic tension law, and the technical challenges involved in integrating them into the codebase.
Section 5: Verification and validation. Proves the reliability of the model through comparison with empirical data, as well as the simulation of real-world conditions for mooring lines.
Section 6: Conclusion and future work. Summarizes the key findings of the project and outlines possible directions for further research and development.

2 Theoretical framework

In order to simulate the behavior of mooring lines governed by partial differential equations (PDEs), it is essential to establish a solid mathematical foundation. This section introduces the key mathematical tools required to formulate, analyze, and numerically solve the governing equations, while Section 3 will bridge these abstract concepts to practical implementation through concrete examples. The structure follows a logical progression from abstract functional spaces to concrete numerical techniques, all of which are critical to developing a stable and accurate computational model.
Though this section is theory-heavy, each concept is purposefully selected to support the numerical treatment of mooring dynamics, particularly the handling of nonlinearities. The transition to applied examples in Section 3 will clarify how these tools ensure stability and accuracy in realistic scenarios.

2.1 Functional Spaces

Spaces play a crucial role in defining the functional framework for PDEs. They provide the mathematical structure to study the properties of solutions, particularly in weak formulations.
Lebesgue spaces are first introduced, which generalize the notion of integrability and provide a framework to handle functions that may not be continuous, yet are still suitable for analysis. This generality is essential because many solutions of PDEs, especially nonlinear ones, may lack classical differentiability. To work with derivatives in a generalized sense, the space $\mathcal{L}^1_{loc}(\Omega)$ is defined, and the concept of tests functions is introduced , which are smooth functions with compact support. These are used to define weak derivatives, allowing one to interpret and work with functions whose classical derivatives may not exist. Then, Sobolev Spaces $W^{k,p}(\Omega)$ are introduced [3, Chapter 5], which extend $\mathcal{L}^p$ spaces by including weak derivatives up to order k. These spaces are fundamental in the variational (weak) formulation of PDEs, which is necessary for the finite element method (FEM) discussed later.

Definition 2.1 (Lebesgue spaces)**.** The space of functions that are Lebesgue integrable on Ω to the power of $p \in [1, \infty)$ is denoted by

$$\mathcal{L}^p(\Omega) = \{f : \int_\Omega |f(x)|^p dx < +\infty\}$$

which is equipped with the norm

$$\|f\|_{\mathcal{L}^p(\Omega)} = \left(\int_\Omega |f(x)|^p dx\right)^{1/p} < +\infty$$

Definition 2.2 (The space of locally integrable functions)**.** Let Ω be an open set in the Euclidean space $\mathbb{R}^n$ and $f : \Omega \to \mathbb{C}$ be a Lebesgue measurable function. If f on Ω is such that

$$\int_\Omega |f(x)| dx < +\infty$$

i.e. its Lebesgue integral is finite on all compact subsets K of Ω, then f is called locally integrable. The set of all such functions is denoted by $\mathcal{L}^1_{loc}(\Omega)$:

$$\mathcal{L}^1_{loc}(\Omega) = \{f : \Omega \to \mathbb{C} \text{ measurable } : f|_K \in \mathcal{L}^1(K)\ \forall K \subset \Omega, K \text{ compact}\}$$

Definition 2.3 (Test function)**.** Let $C^\infty_c(\Omega)$ denote the space of infinitely differentiable functions $\phi : \Omega \to \mathbb{R}$, with compact support in Ω (i.e. every ϕ is non zero in a compact set $K \subset \Omega$ and $\phi(x) = 0$ for every $x \in K \setminus \Omega$). The function ϕ belonging to $C^\infty_c(\Omega)$ is a test function.

Definition 2.4 (Weak n^{th} partial derivative)**.** Suppose that $f, g \in \mathcal{L}^1_{loc}(\Omega)$, and $\alpha = (\alpha_1, ..., \alpha_n)$ is a multiindex of order $|\alpha| = \alpha_1 + ... + \alpha_n$. We say that g is the α^{th} weak partial derivative of f, written

$$D^\alpha f = g$$

provided

$$\int_\Omega f D^\alpha \phi dx = (-1)^{|\alpha|} \int_\Omega g\phi dx$$

for all test functions $\phi \in C^\infty_c(\Omega)$, where $D^\alpha = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} ... \partial x_n^{\alpha_n}}$ is the partial derivative of order α.

In other words, if we are given f and there happens to exist a function g which satisfies the previous equality for all ϕ, we say that $D^\alpha f = g$ in the weak sense. If there does not exist such a function g, then f does not possesss a weak α^{th} partial derivative.

Lemma 2.1 (Uniqueness of weak derivatives)**.** *A weak α^{th} partial derivative of f, if it exists, is uniquely defined up to a set of measure zero.*
Proof. *Assume that $g, \tilde{g} \in \mathcal{L}^1_{loc}(\Omega)$ satisfy*

$$\int_\Omega f D^\alpha \phi dx = (-1)^{|\alpha|} \int_\Omega g\phi dx = (-1)^{|\alpha|} \int_\Omega \tilde{g}\phi dx$$

for all $\phi \in C^\infty_c(\Omega)$. Then

$$\int_\Omega (g - \tilde{g})\phi dx = 0$$

for all $\phi \in C^\infty_c(\Omega)$; whence $g - \tilde{g} = 0$ a.e.

We now define certain function spaces whose members have weak derivatives of various orders lying in various $\mathcal{L}^p$ spaces.

Definition 2.5 (Sobolev spaces)**.** Let $k \in \mathbb{N} \cup \{0\}$, α a multiindex and $p \in [1, \infty)$. The Sobolev space $W^{k,p}(\Omega)$ consists of functions $f \in \mathcal{L}^p(\Omega)$ whose weak derivatives up to order k also belong to $\mathcal{L}^p(\Omega)$:

$$W^{k,p}(\Omega) := \{f \in \mathcal{L}^p(\Omega) : \partial^\alpha f \in \mathcal{L}^p(\Omega) \ \forall \alpha \text{ with } |\alpha| \leq k\}$$

This space is equipped with the norm

$$\|f\|_{W^{k,p}(\Omega)} := \left(\sum_{|\alpha| \leq k} \|D^\alpha f\|^p_{\mathcal{L}^p(\Omega)} \right)^{1/p}$$

where $D^\alpha = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} ... \partial x_n^{\alpha_n}}$ is the partial derivative of order α.

Hilbert spaces are now defined, particularly $H^k(\Omega) = W^{k,2}(\Omega)$, which provide an inner product structure allowing the use of projection methods and orthogonality arguments tools in numerical approximation.

Definition 2.6 (Hilbert space (informal))**.** A Hilbert space is a vector space equipped with an inner product operation, which allows lengths and angles to be defined.

Definition 2.7 (Hilbert space)**.** A Hilbert space is a real or complex vector space H equipped with an inner product

$$\langle \cdot, \cdot \rangle : H \times H \to \mathbb{R} \text{ or } \mathbb{C}$$

that satisfies the following properties for all $x, y, z \in H$ and all scalars α:

1. **Linearity:**
$$\langle \alpha x + y, z \rangle = \alpha \langle x, z \rangle + \langle y, z \rangle$$
2. **Conjugate symmetry (for complex Hilbert spaces)**:
$$\langle x, y \rangle = \overline{\langle y, x \rangle}$$
In the real case, this simplifies to $\langle x, y \rangle = \langle y, x \rangle$.
3. **Positive definiteness:**
$$\langle x, x \rangle \geq 0, \quad \text{with equality if and only if } x = 0.$$

The inner product induces a norm on H given by

$$\|x\| = \sqrt{\langle x, x \rangle}, \quad \forall x \in H.$$

The space H is called a Hilbert space if it is **complete** with respect to this norm, meaning that every Cauchy sequence (x_n) in H satisfies:

$$\forall \epsilon > 0, \quad \exists N \in \mathbb{N} \text{ such that } m, n \geq N \Rightarrow \|x_n - x_m\| < \epsilon$$

Moreover, there exists an element $x \in H$ such that

$$\lim_{n \to \infty} \|x_n - x\| = 0$$

Remark 2.1. The completeness property ensures that limits of convergent sequences remain in H, making Hilbert spaces a fundamental setting for functional analysis, spectral theory, and the study of partial differential equations.

Remark 2.2. If $p = 2$, we usually write

$$H^k(\Omega) = W^{k,2}(\Omega) \quad (k = 0, 1, ...)$$

The letter H is used, since $H^k(\Omega)$ is a Hilbert space.

Note that $H^0(\Omega) = \mathcal{L}^2(\Omega)$.

Definition 2.8 (Space H_0^1)**.** The Sobolev space $H_0^1(\Omega)$ is defined as the closure of $C_c^\infty(\Omega)$ (the space of infinitely differentiable functions with compact support in Ω) with respect to the H^1-norm:

$$H_0^1(\Omega) = \overline{C_c^\infty(\Omega)}^{\|\cdot\|_{H^1}}$$

The H^1-norm is given by

$$\|\phi\|_{H^1} = \left(\int_\Omega |\phi|^2 \, dx + \int_\Omega |\nabla \phi|^2 \, dx \right)^{1/2}$$

Theorem 2.1.

$$H_0^1(\Omega) = \{ f : \Omega \subset \mathbb{R}^n \to \mathbb{R}, f \in H^1(\Omega), f = 0 \text{ on } \partial\Omega \}$$

Proposition 2.1. *The space $H_0^1(\Omega)$ consists of functions in $H^1(\Omega)$ that, in a weak sense, vanish on the boundary of Ω. More precisely, any function in $H_0^1(\Omega)$ is the limit (in the H^1-norm) of a sequence of compactly supported smooth functions in Ω.*

Remark 2.3. Although functions in $H_0^1(\Omega)$ may not vanish on the boundary in the classical sense, they are approximable by smooth functions that do. This space is fundamental in the weak formulation of partial differential equations, particularly for problems with homogeneous Dirichlet boundary conditions.

Definition 2.9 (1D Convolution)**.** Convolution is an operator in which

- for a fixed kernel $g : \mathbb{R} \to \mathbb{R}$, and
- for any input function $f : \mathbb{R} \to \mathbb{R}$
- the operation gives an output function, denoted as $f * g : \mathbb{R} \to \mathbb{R}$, according to the formula

$$(f * g)(t) = \int_{\mathbb{R}} g(t - \tau) f(\tau) d\tau$$

Proposition 2.2 (Convolution and Regularity)**.** *Let $f \in \mathcal{L}_{loc}^1(\mathbb{R})$ and $g \in C_c^\infty(\mathbb{R})$. The convolution $f * g$ satisfies:*

1. *$f * g \in C^\infty(\mathbb{R})$,*
2. *$D^\alpha(f * g) = f * (D^\alpha g)$ for any multiindex α,*

where $D^\alpha = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} ... \partial x_n^{\alpha_n}}$ is the partial derivative of order α.

Proof can be found in [4].

This proposition justifies the use of convolution to approximate non-smooth functions and appears in the hysteresis model, which is described in Section 4.

2.2 PDEs analysis

Partial differential equations (PDEs) that arise in real-world applications can rarely be solved in closed form (i.e., their solution cannot be expressed using a finite combination of well-known elementary functions). In most cases, these equations are too complex to admit explicit solutions, especially when they involve nonlinearity, irregular domains, or complex boundary conditions. However, numerical methods such as the finite element method (FEM), finite difference methods (FDM), or spectral methods, can succesfully be applied to almost all well-posed PDEs. Additionally, weak formulations and variational approaches are frequently employed to extend the notion of a solution, allowing for the study of PDEs in broader functional spaces where classical solutions may not exist.

2.2.1 Strategies for studying PDEs

Definition 2.10. We say that a given problem for a partial differential equation is well-posed (in the sense of Hadamard) if:

1. the problem has a solution;
2. the solution is unique;
3. the solution depends continuously on the data given in the problem.

The last condition is particularly important for problems arising from physical applications: it would be preferable that the (unique) solution changes a little when the conditions specifying the problem change a little. For many problems, on the other hand, uniqueness is not to be expected.

For partial differential equations that cannot be solved in the classical sense, it is a reasonable strategy to consider as separate the existence and the smoothness (or regularity) problems. The idea is to define for a given PDE a reasonable wide notion of a weak solution, with the expectation that, since we are not asking too much by way of smoothness of this weak solution, it may be easier to stablish its existence, uniqueness, and continuous dependence on the given data.

2.2.2 Linear wave equation

The wave equation is a classical example of a second-order hyperbolic PDE. In its linear form, it models wave propagation in an elastic and homogeneous medium.

Definition 2.11 (Linear wave equation)**.** Let $\Omega \subset \mathbb{R}^n$ be a bounded domain with boundary Γ, and $I = [0, T)$ be the time interval. The linear wave equation is given by:

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} - c^2 \Delta u = f & \text{in } \Omega \times I \\ u = 0 & \text{on } \Gamma \times I \\ u(x,0) = u_0(x) & \text{for } x \in \Omega \\ \frac{\partial u}{\partial t}(x,0) = u_1(x) & \text{for } x \in \Omega \end{cases} \tag{2.1}$$

where $u(x,t)$ is the unknown wave amplitude, $c > 0$ is the wave speed, the function $f : \Omega \times I \to \mathbb{R}$ is given, and $\Delta = \nabla^2$ is the Laplacian operator, where $\nabla = \left(\frac{\partial}{\partial x_1}, ..., \frac{\partial}{\partial x_n}\right)$.

In practice, especially in engineering applications, it is often impossible to find classical (smooth) solutions to wave equations, since Equation 2.1 requires u to be twice differentiable in space and time, which may not be realistic in many applications. Instead, the weak formulation is needed, which seeks solutions in Sobolev spaces such as $H^1(\Omega)$, where only weak derivatives are required to exist. This approach allows for discontinuous data and irregular geometries. The weak formulation is constructed through the following steps:

- Multiplying the PDE by a smooth test function v that vanishes on the boundary Γ, i.e., $v \in H_0^1(\Omega)$, and integrate over Ω.
$$\int_\Omega \left(\frac{\partial^2 u}{\partial t^2} - c^2 \Delta u \right) v \, dx = \int_\Omega f v \, dx$$
- Integrating by parts (formally or in the sense of distributions), which transfers derivatives from the solution to the test function, thereby reducing the differentiability requirements on the solution itself.
Applying the multidimensional version of integration by parts (Green's identity):
$$\int_\Omega (\Delta u) v \, dx = \int_\Omega \nabla \cdot (\nabla u) v \, dx = -\int_\Omega \nabla u \cdot \nabla v \, dx + \underbrace{\int_\Gamma \left(\frac{\partial u}{\partial n} \right) v \, ds}_{=0 \text{ since } v \in H_0^1(\Omega)} \text{ where } \frac{\partial u}{\partial n} = \nabla u \cdot \mathbf{n}$$
The equation becomes:
$$\int_\Omega \frac{\partial^2 u}{\partial t^2} v \, dx + c^2 \int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx$$
- Rewriting the equation as an integral identity that must hold for all test functions in the chosen space, as well as adding initial conditions.

Definition 2.12 (Weak formulation of the wave equation)**.** The wave equation 2.1 can be given the following variational formulation. Find $u(t) \in V = H_0^1(\Omega)$, such that for $t \in I$:
$$\begin{cases} \int_\Omega \frac{\partial^2 u}{\partial t^2} v \; dx + c^2 \int_\Omega \nabla u \cdot \nabla v \; dx = \int_\Omega f(t) v \; dx \quad \forall v \in V \\ u(0) = u_0 \\ \frac{\partial u}{\partial t}(0) = u_1 \end{cases} \tag{2.2}$$

To prepare for numerical simulation via the finite element method (explained in Section 2.3), the spatial domain is discretized while keeping time continuous.

Definition 2.13 (Weak semi-discrete formulation of the wave equation)**.** Suppose the finite element space $V_h = H_0^1(\Omega)$ is given (e.g., piecewise linear functions on a mesh). The semi-discrete weak formulation reads: Find $u_h(t) \in V_h$ such that for $t \in (0, T)$
$$\begin{cases} \int_\Omega \frac{\partial^2 u_h}{\partial t^2}(t) v \; dx + c^2 \int_\Omega \nabla u_h(t) \cdot \nabla v \; dx = \int_\Omega f(t) v \; dx \quad \forall v \in V_h \\ u_h(0) = u_{0h} \\ \frac{\partial u_h}{\partial t}(0) = u_{1h} \end{cases} \tag{2.3}$$
where $u_{0h}, u_{1h} \in V_h$ are approximations of the initial data u_0 and u_1.

Proposition 2.3 (Well-posedness of the weak formulation)**.** *For $f \in \mathcal{L}^2(\Omega \times I)$, $u_0 \in H_0^1(\Omega)$ and $u_1 \in \mathcal{L}^2(\Omega)$, there exists a unique solution $u \in C^0(I, H_0^1(\Omega)) \cap C^1(I, \mathcal{L}^2(\Omega))$, and the solution depends continuously on the data.*

This result can be found in [3, Chapter 7, Section 2].

2.2.3 Non-linear wave equation

Many real-world phenomena involve nonlinear wave behavior, requiring extensions of the linear model. A general nonlinear wave equation takes the form:

$$\frac{\partial^2 u}{\partial t^2} - c^2 \Delta u + F(u, \Delta u) = f \tag{2.4}$$

where $F(u, \Delta u)$ encapsulates the nonlinear effects.

Despite the differences, the strategy used in the linear case extends naturally: weak formulation is constructed by testing the nonlinear equation against smooth functions, integrating by parts, and seeking solutions in Sobolev spaces. The resulting formulation serves as the basis for the finite element implementation developed in this work and is introduced formally in Section 2.4. The well-posedness of nonlinear wave equations in weak form is generally more delicate than in the linear case. Depending on the structure of the nonlinearity (particularly its continuity, growth conditions, and monotonicity) existence and uniqueness can often still be established via Galerkin approximations, energy estimates, and compactness arguments [3, Chapter 12]. The Galerkin method presented in [5, Chapter 2] provides the abstract framework used to establish existence and uniqueness of the semi-discrete problem in the nonlinear case, even though the full treatment of time-dependent hyperbolic problems is beyond their scope.

2.3 Separation of Variables: The Homogeneous Problem

Separation of variables is an important technique to master when studying solution methods of PDEs. It is explained in this section not only as an analytical tool, but also as a foundational step in the Galerkin finite element method (FEM), explained later in Section 2.4. However, separation of variables does have its drawbacks, since it requires both the PDE and the boundary conditions (BCs) be homogeneous. Second, in general the spatial variable must have finite boundaries. If the spatial variable has semi-infinite or infinite boundaries, separation usually does not work. While separation of variables cannot be applied directly to nonlinear or inhomogeneous problems, the structure it has is used in more advanced techniques such as the Galerkin finite element method.

The main idea of separation of variables is to assume that the solution can be written as the product of functions, each depending on a single variable. For a PDE involving two variables, say x and t, we seek a solution of the form:

$$u(x, t) = X(x)T(t)$$

where $X(x)$ is a function that depends only on x and $T(t)$ is a function that depends only on t.
By substituting this assumption into the PDE and using algebraic manipulations, the equation is split into separate ordinary differential equations (ODEs), each depending on a single variable.

2.4 Finite element methods (FEM)

The Finite Element Method (FEM) is a powerful numerical technique for solving partial differential equations (PDEs) and integral equations. It is particularly well-suited for problems involving complex geometries, material properties, and boundary conditions. The method transforms continuous PDEs into discrete systems of algebraic equations by dividing the domain into smaller, simpler subdomains called finite elements.

2.4.1 Historical context and overview

The FEM emerged as a formal computational tool in the mid-20th century, with pioneering contributions from Turner, Clough, Martin, and Topp (1956) and Argyris (1957). Its name reflects the discretization of a continuous domain into finite elements, where approximate solutions are constructed. Unlike classical analytical methods, which seek exact solutions to idealized problems, FEM provides approximate solutions to real-world problems by making piecewise polynomial interpolation over these elements. The method gained widespread adoption in engineering and applied mathematics due to its flexibility and robustness, as documented in foundational texts like Zienkiewicz (1971) and Oden (1991). [6]

2.4.2 Mathematical formulation

The core idea of FEM lies in approximating the solution of a PDE using a finite-dimensional subspace. This involves three key steps:

1. Domain discretization: The spatial domain Ω is partitioned into a mesh of finite elements (e.g., triangles in 2D, tetrahedra in 3D).

2. Weak formulation: The PDE is reformulated in an integral (weak) form, reducing differentiability requirements and enabling the use of Sobolev spaces (see Section 2.1).

3. Galerkin approximation: The solution is projected onto a subspace spanned by basis functions $\{\phi_i\}$, typically piecewise polynomials, leading to a system of algebraic equations.

A finite element is a mathematical tool used to break down complex problems into smaller, solvable pieces. The domain (the region where the problem is defined, like a metal plate or fluid volume) is divided into small, simple shapes (triangles, quadrilaterals, tetrahedrons, etc.). These shapes are called elements, and their corners (or edges) are called nodes. Inside each element, the unknown function (e.g., temperature, displacement) is approximated using simple polynomial functions (often linear or quadratic). Each finite element gives a small set of equations. To solve the full problem, all those equations are combined (assembled) into one big system of equations. The global system is solved numerically (using linear algebra methods) to find the approximate solution at all nodes. Once solved, the solution can be reconstructed everywhere by combining the local approximations.

Definition 2.14 (Ciarlet´s finite element)**.** Let

i) $K \subseteq \mathbb{R}^n$ be a bounded closed set with nonempty interior and piece-wise smooth boundary (the element domain),

ii) $\mathcal{P}$ be a finite dimensional space of functions on K (the space of shape functions: a set of polynomials of a certain degree defined on K that interpolate the solution),

iii) $\mathcal{N} = \{n_1, n_2, ..., n_k\}$ be a basis of the dual space of $\mathcal{P}$, $\mathcal{P}'$. The dual space $\mathcal{P}'$ consists of linear functionals (e.g., point evaluations $n_i(f) = f(x_i)$) that map shape functions $f \in P$ to real numbers.

Then, $(K, \mathcal{P}, \mathcal{N})$ is called a finite element.

Definition 2.15 (Nodal basis)**.** Given a finite element $(K, \mathcal{P}, \mathcal{N})$, where $\mathcal{N} = \{n_1, n_2, ..., n_k\}$ is a set of nodal variables (degrees of freedom), the nodal basis $\{\phi_1, \phi_2, ..., \phi_k\} \subset \mathcal{P}$ is the unique set of shape functions satisfying:

$$n_i(\phi_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

Definition 2.16 (Galerkin method)**.** The Galerkin method is a finite element method (FEM) used for solving partial differential equations (PDEs). It is based on approximating the solution in a finite-dimensional subspace while ensuring that the residual of the equation is orthogonal to the subspace. Below, the method is explained in the context of PDEs, particularly wave equations.

Multiplying the linear wave equation by a test function $v \in V = H_0^1(\Omega)$ and integrating over Ω the weak form is obtained, where derivatives are transferred to v via integration by parts:

$$\int_\Omega \frac{\partial^2 u}{\partial t^2} v \, dx + c^2 \int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx$$

The space V is discretized by choosing a finite-dimensional subspace $V_h \subset V$ with basis $\{\phi_1, \phi_2, ..., \phi_N\}$. Then the approximate solution u_h can be expressed as:

$$u_h(x, t) = \sum_{i=1}^{N} \alpha_i(t) \phi_i(x)$$

where $\alpha_i(t)$ are time-dependent coefficients to be determined.
Then, the basis functions are used as test functions: $v = \phi_j$ for $j = 1, ..., N$.
Substituting into the weak form u with u_h and v with ϕ_j:

$$\int_\Omega \left(\sum_{i=1}^{N} \ddot{\alpha}_i(t) \phi_i \right) \phi_j \, dx + c^2 \int_\Omega \left(\sum_{i=1}^{N} \alpha_i(t) \nabla \phi_i \right) \cdot \nabla \phi_j \, dx = \int_\Omega f \phi_j \, dx$$

which can be rewritten:

$$\sum_{i=1}^{N} \ddot{\alpha}_i(t) \int_\Omega \phi_i \phi_j \, dx + c^2 \sum_{i=1}^{N} \alpha_i(t) \int_\Omega \nabla \phi_i \cdot \nabla \phi_j \, dx = \int_\Omega f \phi_j \, dx \quad \text{for } j = 1, \ldots, N$$

Let´s define:

- Mass matrix $M \in \mathbb{R}^{N\times N}$:
$$M_{ij} = \int_\Omega \phi_i \phi_j \, dx$$
- Stiffness matrix $K \in \mathbb{R}^{N\times N}$:
$$K_{ij} = \int_\Omega \nabla\phi_i \cdot \nabla\phi_j \, dx$$
- Force vector $F(t) \in \mathbb{R}^N$:
$$F_j(t) = \int_\Omega f(x,t)\phi_j \, dx$$

The equation becomes:

$$\sum_{i=1}^{N} M_{ij}\ddot{\alpha}_i(t) + c^2 \sum_{i=1}^{N} K_{ij}\alpha_i(t) = F_j(t), \quad j = 1, \ldots, N.$$

In matrix form:

$$M\ddot{\alpha}(t) + c^2 K\alpha(t) = F(t) \tag{2.5}$$

where $\alpha(t) = [\alpha_1(t), \ldots, \alpha_N(t)]^T$.

Algorithm for implementation:

1. Discretize the domain: Divide Ω into smaller elements (e.g., triangles or quadrilaterals in 2D).
2. Define Basis Functions: Choose piecewise polynomial basis functions $\phi_i(x)$ (e.g., linear or quadratic).
3. Assemble Matrices: Compute the entries of M and K by integrating over elements. Then form the global system by assembling contributions from all elements.
4. Solve the System: Use numerical methods (e.g., implicit schemes like Newmark-Beta or BDF2) to integrate the ODEs system in time.

Theorem 2.2 (Convergence of FEM)**.** *Given $u \in H_0^1(\Omega)$ as the solution to the weak form of a PDE, and u_h its FEM approximation in a finite-dimension subspace $V_h \subset H_0^1(\Omega)$, the approximation error satisfies:*

$$\|u - u_h\|_{H^1(\Omega)} \leq Ch^k \|u\|_{H^{k+1}(\Omega)} \tag{2.6}$$

where h is the maximum element size and k is the polynomial degree of V_h.

Proof of the theorem can be found in [7].

2.5 Temporal integration methods

Temporal integration methods are essential for solving systems of ordinary differential equations (ODEs). This subsection introduces the basic concepts, challenges, and key techniques for temporal resolution, with a focus on stability, accuracy, and efficiency.

2.5.1 Boundary conditions problem

In the study of partial differential equations (PDEs), boundary conditions (BCs) play a fundamental role in ensuring the existence, uniqueness, and stability of solutions. In the context of the finite element method (FEM), the correct implementation of boundary conditions is essential for obtaining accurate numerical approximations. This subsection explores the types of boundary conditions, their mathematical formulation, and common challenges in their numerical implementation.

- **Types of boundary conditions.**
  Boundary conditions describe how the solution behaves at the domain boundaries and are classified into three main categories:

  1. **Dirichlet Boundary Conditions**: These conditions impose a fixed value of the function at the boundary:

     $$u(x) = g(x), \ x \in \partial\Omega$$

     They are often used in problems where the solution is known at the boundary, such as temperature or displacement constraints in elasticity.

  2. **Neumann Boundary Conditions**: These conditions specify the normal derivative of the function at the boundary:

     $$\frac{\partial u}{\partial n} = h(x), \ x \in \partial\Omega$$

     They arise in problems involving fluxes, such as heat conduction or stress conditions in elasticity.

  3. **Robin (Mixed) Boundary Conditions**: These conditions involve a combination of Dirichlet and Neumann terms:

     $$\alpha u + \beta \frac{\partial u}{\partial n} = f(x), \ x \in \partial\Omega$$

     They are used in heat transfer with convection, wave propagation problems, and fluid-structure interactions.

In time-dependent problems, initial conditions play a critical role in ensuring the solution's accuracy and stability. The general form of a temporal problem is:

$$\frac{du}{dt} = f(t, u), \ u(t_0) = u_0$$

where $u(t_0) = u_0$ specifies the initial state of the system. For second-order problems like wave equations, the initial conditions may include both displacement and velocity:

$$u(0) = u_0, \ \frac{du}{dt}(0) = v_0$$

2.5.2 Common Challenges

1. Accuracy of Initialization: Errors in u_0 or v_0 propagate through the solution.
2. Compatibility with Spatial Discretization: Initial conditions must align with the discretized system to prevent instability.
3. High-Frequency Artifacts: Poorly chosen initial conditions (e.g., non-smooth) can create rapid oscillations in the numerical solution that do not correspond to the true physical behavior.

2.5.3 Methods to Address Challenges

1. Use consistent initialization derived from the continuous equations (in this work static equilibrium conditions are imposed, see Section 3.2.5).
2. Apply regularization techniques to smooth out incompatible initial conditions.
3. Use temporal integrators that damp or control high-frequency errors from poor initialization (e.g., BDF2).

2.5.4 Stiff ODEs system

There is no precise definition of what a stiff equation is, but it is known that certain numerical methods became unstable when solving the equation, and they require extremely small step sizes to achieve stability. The reason why this subsection is of high importance is that the central model of this project results in an ODEs system, explained in detail in Section 3, which is stiff.
The integration method chosen to solve ODEs systems must be as fast as possible and present great stability at the same time. Explicit methods (the solution at a future time step is calculated directly from the known values at the current time step) are not usually recommended for problems which can be stiff. Stiffness is typically characterized by the eigenvalues of the system's Jacobian matrix (stiffness matrix K in the ODE system 2.5). While analytical eigenvalue computation is intractable for large nonlinear systems (like mooring lines), stiffness arises when a system's eigenvalues λ span widely separated time scales [8]. In practice, stiffness is deduced from the system's behavior (e.g., explicit solvers fail unless step sizes are impractically small), not explicit eigenvalue calculations.

Definition 2.17 (Stability function)**.** The function $R(z)$ is called the stability function of the method. It can be interpreted as the numerical solution after one step for the Dahlquist test equation:

$$y' = \lambda y, \quad y_0 = 1, \quad z = h\lambda, \tag{2.7}$$

This is a simple linear test equation used to analyze stability, where λ is a constant (possibly complex), h is the step size of the numerical method, and $z = h\lambda$ is dimensionless. The exact solution of the equation is:

$$y(t) = e^{\lambda t}$$

A numerical method approximates this evolution, and its behavior is captured by the function $R(z)$, which expresses the numerical result after one time step.

For a given numerical method, applying one step to the test equation $y' = \lambda y$ results in an update of the form $y_{n+1} = R(z)y_n$. Thus, after one step, the numerical solution is $y_1 = R(z)y_0$. If we started with $y_0 = 1$, then $y_1 = R(z)$. So, $R(z)$ tells us how the numerical method transforms the initial condition after one step, relative to the exact exponential solution.
The set

$$S = \{z \in \mathbb{C}; \quad |R(z)| \leq 1\}$$

is called the stability domain of the method.

Definition 2.18 (A-Stability (Dahlquist 1963))**.** A method, whose stability domain satisfies

$$S \supset \mathbb{C}^- = \{z; \ \mathrm{Re}(z) \leq 0\}$$

is called A-stable.

Some methods are stable on the entire left half-plane $\mathbb{C}^-$. This is precisely the set of eigenvalues, where the exact solution of (2.7) is stable too [9]. A desirable property for a numerical method is that it preserves this stability property. A-stability ensures that the method remains stable for all step sizes when applied to problems with eigenvalues in the left half-plane (common in stiff systems).
Multistep methods use solutions from several prior time steps to compute the current step, offering higher-order accuracy with fewer function evaluations compared to single-step methods. Their implicit nature makes them particularly suited for stiff systems, where stability constraints dominate step size selection.

Theorem 2.3 (The Dahlquist second barrier)**.** *The highest order of an A-stable multistep method is 2. [10]*

Definition 2.19 (BDF2 adaptive step)**.** The Backward Differentiation Formula (BDF) is a family of implicit multistep methods widely used for stiff ODEs. The second-order BDF (BDF2) is particularly attractive due to its balance between accuracy and stability. For the problem $\frac{du}{dt} = f(t, u)$, BDF2 approximates the solution at time t_{n+2} using:

$$\frac{3}{2}u_{n+2} - 2u_{n+1} + \frac{1}{2}u_n = hf(t_{n+2}, u_{n+2})$$

where h is the time step size.

Remark 2.4. It's important to note that this requires solving a nonlinear system for u_{n+2} at each step.

Nevertheless, the advancing formula of the adaptive step size BDF2 is given by the next expression [11]:

$$y_{n+2} - \frac{(1+w_{n+1})^2}{1+2w_{n+1}}y_{n+1} + \frac{w_{n+1}^2}{1+2w_{n+1}}y_n = h_{n+2}\frac{1+w_{n+1}}{1+2w_{n+1}}f_{n+2}$$

being $w_{n+1} = \frac{h_{n+2}}{h_{n+1}}$, $h_{n+2} = t_{n+2} - t_{n+1}$ and $h_{n+1} = t_{n+1} - t_n$

The local truncation error (LTE) becomes:

$$LTE = y(t_{n+2}) - y_{n+2} \approx \frac{h_{n+2}^2(h_{n+1} + h_{n+2})}{6} y^{(3)}(t_{n+2}) \tag{2.8}$$

where $y^{(3)}$ is an approximation of the third derivative of y:

$$y^{(3)}(t_{n+2}) \approx \frac{1}{h_{n+2}^2}\left[\frac{y_{n+2} - y_{n+1}}{h_{n+2}} - (1 + \frac{h_{n+2}}{h_{n+1}})\frac{y_{n+1} - y_n}{h_{n+1}} + \frac{h_{n+2}}{h_{n+1}h_n}(y_n - y_{n-1})\right] \tag{2.9}$$

2.5.5 Summary

The BDF2 algorithm solves the system of ODEs derived from the finite element discretization by:

- Applying a second-order accurate implicit multistep method.
- Solving a nonlinear equation at each time step using Newton-Raphson iterations for example.
- Adapting the time step based on an estimate of the local truncation error.
- Ensuring stability for stiff problems due to its A-stability up to second order.

Advantages of Adaptive BDF2:

1. Stability: Well-suited for stiff problems.
2. Efficiency: Saves computation time by using larger time steps in smooth regions.
3. Accuracy: Maintains second-order convergence.

3 Mooring line model

The dynamic behavior of mooring lines in marine environments is governed by complex interactions between mechanical forces, material properties, and hydrodynamic effects. Accurately simulating these systems requires a well defined and numerical stable approach capable of capturing nonlinearities, geometric deformations, and time-dependent responses. This section presents a model for mooring line dynamics, beginning with the derivation of the governing equations from Newtonian mechanics and finishing in their numerical implementation via the Finite Element Method (FEM).
With the theoretical framework of Section 2 established—including Sobolev spaces, weak formulations, and semidiscretization— the focus now lies on their practical application to mooring line dynamics. This section details the steps to transform the governing PDEs into a solvable numerical system. A nonlinear partial differential equation (PDE) describes the motion of the mooring line under tension, external forces, and boundary constraints. To address the challenges of this PDE, such as its high nonlinearity and coupling between spatial and temporal variables, the problem is reformulated in a weak form, enabling the use of FEM for spatial discretization. The resulting system of ordinary differential equations (ODEs) is then solved using advanced time-integration techniques, including the adaptive BDF2 method, which ensures stability and accuracy for stiff systems.

3.1 PDE formulation

Considering that no external angular momentum is applied, Newton´s equation expressed per unit of line length is used to model mooring and towing lines [12], [13], [14], [15].

$$\rho_0 \frac{\partial^2 r(t,s)}{\partial t^2} = \frac{\partial}{\partial s}\left(T(t,s)\frac{\partial \mathbf{r}(t,s)}{\partial s}\right) + f(t,s)\cdot\left|\frac{\partial r}{\partial s}\right| \tag{3.1}$$

where ρ_0 is the cable mass per unit length, $s \in [0, L]$ the arc-length parameter and $r : [0,\infty) \times [0,L] \to \mathbb{R}^3$ the position in the inertial frame. $T : [0,\infty) \times [0,L] \to \mathbb{R}$ is the tension vector and $f : [0,\infty) \times [0,L] \to \mathbb{R}^3$ the external forces per unit of length vector. The considered external forces are:

$$f = f_{hg} + f_{dt} + f_{dn} + f_{mt} + f_{mn} + f_{gnk} + f_{gnd} \tag{3.2}$$

where

- $f_{hg} = \frac{\rho_0 - \rho_w \cdot A}{\left|\frac{\partial r}{\partial s}\right|}\mathbf{g}$ represents the combined gravitational and buoyancy forces acting on the mooring line, based on Archimedes' principle. Since the mooring line is always submerged, both forces are continuously acting. The gravitational force tends to pull the line downward, while the buoyancy force acts in the opposite direction.

- $f_{dt} = -\frac{1}{2}\cdot C_{DT}\cdot d\cdot \rho_w \cdot |v_t|\cdot v_t$ and $f_{dn} = -\frac{1}{2}\cdot C_{DN}\cdot d\cdot \rho_w \cdot |v_t|\cdot v_n$ are the tangential and normal drag forces, respectively, due to fluid resistance as the mooring line moves through seawater. This drag force arises from the interaction between the moving mooring line and the surrounding water, which resists the motion of the line.

- $f_{mt} = -C_{MT}\cdot \frac{\pi d^2}{4}\cdot \rho_w \cdot \mathbf{a}_t$ and $f_{mn} = -C_{MN}\cdot \frac{\pi d^2}{4}\cdot \rho_w \cdot \mathbf{a}_n$ are the tangential and normal added mass forces, respectively, that arise from the displacement of the surrounding fluid as the mooring line moves through it, creating an additional inertia force that must be accounted for. This force is due to the fact that when the mooring line accelerates, it not only moves its own mass but also displaces and accelerates the fluid around it. This results in added resistance to motion, known as the added mass effect.

- $f_{gnk} = z\cdot |f_{hg}|\cdot e^{-G_K d(r_z - z_f)/|f_{hg}|}$ is the normal ground reaction force, which represents the contact force exerted by the seabed on the mooring line when it touches the ground. This force prevents the mooring line from penetrating the seabed by pushing back upward.

- $f_{gnd} = f_{hg}\cdot G_D \cdot d \cdot \min(v_z, 0)^2$ is the normal ground damping force, which accounts for the energy dissipation (damping) that occurs as the mooring line interacts with the seabed. It is a velocity-dependent force that resists the vertical motion of the mooring line as it moves downward toward the seabed.

and the parameters are
The tangent vector velocity is computed as $\mathbf{v_t} = (\mathbf{v}\cdot\mathbf{t})\cdot\mathbf{t}$, where $\mathbf{t}$ is the unitary tangent vector. Normal velocity is $\mathbf{v}_n = \mathbf{v} - \mathbf{v}_t$. The same applies for acceleration.

ρ_w	water density	$\mathbf{v}_t$	line tangential velocity
$\mathbf{g}$	gravity acceleration vector	$\mathbf{v}_n$	line normal velocity
A	line section	$\mathbf{a}_t$	line tangential acceleration
d	line diameter	$\mathbf{a}_n$	line normal acceleration
e	axial strain	G_K	ground normal stiffness
C_{DT}	line tangential drag coefficient	G_D	ground´s critical damping
C_{DN}	line normal drag coefficient	$\mathbf{z}$	vertical unitary vector
C_{MT}	line tangential added mass coefficient	r_z	vertical coordinate of a line node
C_{MN}	line normal added mass coefficient	z_f	vertical coordinate of ocean floor
		v_z	vertical velocity of a line node

It is important to note that the strain $\varepsilon(t,s)$ is defined as:

$$\left|\frac{\partial \mathbf{r}(t,s)}{\partial s}\right| = 1 + \varepsilon(t,s) \tag{3.3}$$

Furthermore, Eq. 3.1 is restricted to the following conditions.

- The spatial boundary conditions on the lines will be the positions of both ends: the fixed anchor at point $\mathbf{r}(t,0) = \mathbf{r}_0$, and the time-dependent position of the fairlead $\mathbf{r}(t,L) = \mathbf{r}_\mathbf{F}(t)$, describing the wave movement that is typically a sinusoidal equation.
- As second derivatives appear in the PDE system in 3.1, both the initial mooring line position $\mathbf{r}(0,s)$ and initial velocity $\frac{\partial \mathbf{r}(0,s)}{\partial t}$ must be provided to be able to solve it.

The initial mooring line position will be described by a function $g(s)$, and the initial mooring line velocity will be considered zero (static approach).

$$\frac{\partial \mathbf{r}(0,s)}{\partial t} = 0; \quad \mathbf{r}(0,s) = g(s) \tag{3.4}$$

Existence and uniqueness of solutions are proved in [12] for when the mooring line is purely elastic (i.e. linear tension term). When the material exhibits hysteresis (i.e. non linear tension term, which is introduced in Section 4), proving the existence and uniqueness is more delicate, and shown in [16].

3.2 FEM: From PDE to ODEs system

3.2.1 Weak formulation

The PDE line system 3.1 is multiplied with the test function and taking the integral with s along the line length:

$$\rho_0 \frac{\partial^2 \mathbf{r}(t,s)}{\partial t^2} w(s) = \frac{\partial}{\partial s}\left(T(t,s)\frac{\partial \mathbf{r}(t,s)}{\partial s}\right) w(s) + \mathbf{f}(t,s) \cdot \left|\frac{\partial r}{\partial s}\right| w(s) \Longrightarrow$$

$$\int_0^L \rho_0 \frac{\partial^2 \mathbf{r}(t,s)}{\partial t^2} w(s) ds = \int_0^L \frac{\partial}{\partial s}\left(T(t,s)\frac{\partial \mathbf{r}(t,s)}{\partial s}\right) w(s) ds + \int_0^L \mathbf{f}(t,s) \cdot \left|\frac{\partial r}{\partial s}\right| w(s) ds$$

Finally, using integration by parts and knowing that as $w(s) \in V \implies w(0) = w(L) = 0$, the weak formulation of the problem is obtained

$$\int_0^L \rho_0 \frac{\partial^2 \mathbf{r}(t,s)}{\partial t^2} w(s) ds = -\int_0^L T(t,s) \frac{\partial \mathbf{r}(t,s)}{\partial s} \frac{dw(s)}{ds} ds + \int_0^L \mathbf{f}(t,s) \cdot \left| \frac{\partial r}{\partial s} \right| w(s) ds \quad (3.5)$$

equivalent to finding the minimum error (minimization problem) in L^2 (Burden and Faires, 2013)

3.2.2 Galerkin Finite Element Method

Lets consider the division of the mooring line into N equidistance nodes $\{s_i\}_{i=0}^{N-1}$. The finite sub-base of V will have dimension N, $\{\phi_i\}_{i=0}^{N-1}$, which must satisfy $\phi_i(0) = \phi_i(L)$. The mooring line position can be written as a combination of the base functions:

$$\mathbf{r}(t,s) = \sum_{i=0}^{N-1} r_i(t) \phi_i(s)$$

where $r_i(t) = \mathbf{r}(t, s_i)$ are the mooring line nodes positions. It is important to note that the coefficients $\{r_i\}_{i=0}^{N-1}$ are vectors in $\mathbb{R}^3$.
Also, for the derivatives:

$$\frac{\partial^m \mathbf{r}(t,s)}{\partial t^m} = \sum_{i=0}^{N-1} \frac{\partial^m r_i(t)}{\partial t^m} \phi_i(s)$$

and

$$\frac{\partial^{m+1} \mathbf{r}(t,s)}{\partial t^m \partial s} = \sum_{i=0}^{N-1} \frac{\partial^m r_i(t)}{\partial t^m} \frac{\partial \phi_i(s)}{\partial s}$$

Furthermore,

$$\frac{\partial \mathbf{r}}{\partial s} = \sum_{k=0}^{N-1} r_k(t) \frac{\partial \phi_k(s)}{\partial s}$$

and so

$$\left| \frac{\partial r}{\partial s} \right| = \sqrt{\left(\sum_{k=0}^{N-1} r_k(t) \frac{\partial \phi_k(s)}{\partial s} \right)^2}$$

for which the following approximation can be done:

$$\left| \frac{\partial r}{\partial s} \right| \approx \sum_{k=0}^{N-1} J_k(t) \phi_k(s)$$

With a good choice of the base and knowing the positions and velocities of certain points in the line, the derivatives needed to compute internal forces $\mathbf{F}(t,s) = T(t,s) \frac{\partial \mathbf{r}(t,s)}{\partial s}$ and external forces $\mathbf{f}$ can be easily obtained, as it is shown later. Considering this, internal and

external forces can be expressed in terms of the base, as it was done for $\mathbf{r}$:

$$\mathbf{F}(t,s) = \sum_{i=0}^{N-1} T(t,s) r_i(t) \frac{d\phi_i(s)}{ds} \qquad \mathbf{f}(t,s) = \sum_{i=0}^{N-1} f_i(t)\phi_i(s)$$

Choosing the test function to be a function of the bases of V, $w(s) = \phi_j(s)$, equation 3.5 results in:

$$\rho_0 \int_0^L \left(\sum_{i=0}^{N-1} \frac{\partial^2 r_i(t)}{\partial t^2} \phi_i(s) \right) \phi_j(s) ds =$$
$$- \int_0^L \left(\sum_{i=0}^{N-1} T(t,s) r_i(t) \frac{d\phi_i(s)}{ds} \right) \frac{d\phi_j(s)}{ds} ds + \int_0^L \left(\sum_{i=0}^{N-1} f_i(t) \cdot \phi_i(s) \right) \left(\sum_{k=0}^{N-1} J_k(t)\phi_k(s) \right) \phi_j(s) ds$$
$$\text{for } j = 0, 1, ..., N-1$$

which applying the integrals´ linearity:

$$\rho_0 \sum_{i=0}^{N-1} \frac{\partial^2 r_i(t)}{\partial t^2} \int_0^L \phi_i(s)\phi_j(s) ds =$$
$$- \sum_{i=0}^{N-1} r_i(t) \int_0^L T(t,s) \frac{d\phi_i(s)}{ds} \frac{d\phi_j(s)}{ds} ds + \sum_{i=0}^{N-1} f_i(t) \sum_{k=0}^{N-1} J_k(t) \int_0^L \phi_i(s)\phi_j(s)\phi_k(s) ds$$
$$\text{for } j = 0, 1, ..., N-1$$

where using matrix notation, a second order system of ODEs is obtained:

$$\rho_0 \mathbf{M} \begin{pmatrix} \frac{\partial^2 r_0(t)}{\partial t^2} \\ \frac{\partial^2 r_1(t)}{\partial t^2} \\ \vdots \\ \frac{\partial^2 r_{N-1}(t)}{\partial t^2} \end{pmatrix} = -\mathbf{K}(t) \begin{pmatrix} r_0(t) \\ r_1(t) \\ \vdots \\ r_{N-1}(t) \end{pmatrix} + \mathbf{F^{ext}}(t) \tag{3.6}$$

$$\text{where: } \mathbf{M_{ij}} = \cdot \int_0^L \phi_i(s)\phi_j(s) ds \quad \mathbf{K}_{ij}(t) = \int_0^L T(t,s) \frac{d\phi_i(s)}{ds} \frac{d\phi_j(s)}{ds} ds \tag{3.7}$$

usually named the mass matrix and stiffness matrix, respectively.
In order to compute $\mathbf{M_{ij}}$, $\mathbf{K}_{ij}$, $\mathbf{F_i^{ext}}$, first it is necessary to describe the basis functions, which has a significant importance in computational cost when solving Eq. 3.6.

3.2.3 Spatial discretization and basis functions.

Let the spatial domain be denoted by the closed interval $[0, L] \subset \mathbb{R}$, representing the length of the mooring line. The discretization of this domain and the choice of basis functions are fundamental in ensuring both numerical accuracy and computational efficiency.

Definition 3.1. The interval $[0, L]$ is discretized into N equidistant points $\{s_i\}_{i=0}^{N-1}$, called *nodes*, satisfying the following properties:

1. $s_0 = 0, \quad s_{N-1} = L$

2. $s_i - s_{i-1} = l = \dfrac{L}{N-1}, \quad$ for all $i = 1, \ldots, N-1$

Remark 3.1. A regular discretization guarantees uniform spacing between nodes, which simplifies the construction and analysis of basis functions.

Definition 3.2. Let $\{\phi_i\}_{i=0}^{N-1}$ be a set of piecewise linear basis functions associated with the nodes $\{s_i\}$. Each function $\phi_i : [0, L] \to \mathbb{R}$ is defined as

$$\phi_i(s) = \begin{cases} 0, & s < s_{i-1} \\ \dfrac{s - s_{i-1}}{l}, & s_{i-1} \leq s < s_i \\ \dfrac{-s + s_{i+1}}{l}, & s_i \leq s < s_{i+1} \\ 0, & s \geq s_{i+1} \end{cases} \tag{3.8}$$

with $l = s_i - s_{i-1}$.

Remark 3.2. The choice of piecewise linear basis functions avoids the Runge phenomenon associated with higher-order polynomials and ensures good interpolation properties. Specifically, polynomials of order higher than 5 are avoided, following best practices highlighted in the literature [12].

Proposition 3.1 (Kronecker Delta Property)**.** *The basis functions 3.8 satisfy the interpolation condition*

$$\phi_i(s_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{3.9}$$

Proof. By the definition of $\phi_i(s)$ in (3.8), we have:

- $\phi_i(s_j) = 0$ if $s_j \notin [s_{i-1}, s_{i+1}]$
- $\phi_i(s_i) = \dfrac{-s_i + s_{i+1}}{l} = \dfrac{l}{l} = 1$

Thus, $\phi_i(s_j) = \delta_{ij}$. □

Proposition 3.2 (Support Overlap Property)**.** *The basis functions 3.8 have the following orthogonality property with respect to their support:*

$$\int_0^L \phi_i(s)\phi_j(s)\, ds = 0 \quad \forall j \notin \{i-1, i, i+1\} \tag{3.10}$$

Definition 3.3 (Weak Derivatives of Basis Functions)**.** The weak derivative of ϕ_i is piecewise constant and given by

$$\frac{d\phi_i(s)}{ds} = \begin{cases} 0, & s < s_{i-1} \\ \dfrac{1}{l}, & s_{i-1} \leq s < s_i \\ -\dfrac{1}{l}, & s_i \leq s < s_{i+1} \\ 0, & s \geq s_{i+1} \end{cases} \tag{3.11}$$

Proposition 3.3 (Derivative Support Overlap Property)**.** *The derivatives of the basis functions satisfy*

$$\frac{d\phi_i(s)}{ds}\frac{d\phi_j(s)}{ds} = 0 \quad \forall j \notin \{i-1, i, i+1\} \tag{3.12}$$

Remark 3.3. The fact that the product of derivatives vanishes outside the index set $\{i-1, i, i+1\}$ leads to a *tridiagonal stiffness matrix* in the finite element formulation. This sparsity structure significantly reduces the computational complexity of matrix assembly and subsequent linear system solutions.

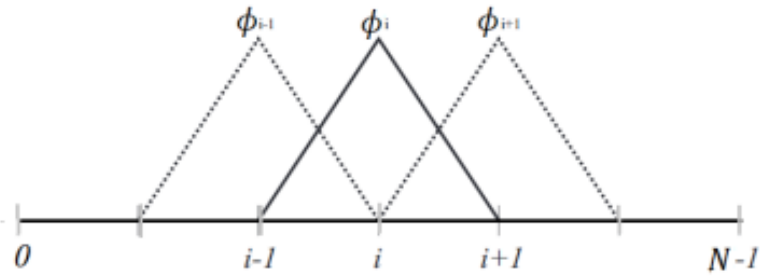


Figure 1: Linear basis functions $\phi_i(s)$, as introduced in Aamo and Fossen [12].

Remark 3.4. The basis functions $\phi_i \in H^1(0, L)$, ensuring they are suitable for Galerkin finite element formulations that require H^1 regularity.

Remark 3.5. The choice of basis functions for constructing the solution space is not unique. Alternative approaches, such as using higher-order functions like B-splines or higher-order polynomials, also exist (Burden and Faires, [17]). One key advantage of these higher-order functions is that they provide smooth and continuous derivatives without requiring their definition in the sense of distributions, offering improved accuracy for certain types of problems.
However, these benefits come with trade-offs. Incorporating higher-order basis functions adds complexity to the formulation, particularly by increasing the number of terms in the mass and stiffness matrices. This increase leads to a higher computational burden when solving the system of ordinary differential equations (ODEs) in 3.6. The computational cost grows significantly unless measures are taken to reduce the number of mooring line nodes or adopt more efficient numerical strategies.
Moreover, while higher-order methods can offer improved precision, especially in capturing complex physical behaviors such as bending and twisting in structures, they may not always be the most practical choice in large-scale simulations. In scenarios where real-time performance or limited computational resources are priorities, the increased complexity could outweigh the accuracy gains, prompting a trade-off between computational efficiency and solution fidelity. Hence, the selection of basis functions often hinges on balancing these competing factors: the need for accuracy, the nature of the problem, and the available computational resources.

Remark 3.6. Note that by choosing this piecewise linear basis, for $s \in [s_{i-1}, s_i]$

$$\left|\frac{\partial r}{\partial s}\right| = \left|r_{i-1}(t)\frac{-1}{h} + r_i(t)\frac{1}{h}\right| = \frac{|r_i(t) - r_{i-1}(t)|}{h}$$

3.2.4 From second order to first order ODEs system

By applying the Galerkin FEM approximation, the second order ODEs system was obtained in 3.6. The following procedure is done to convert the N equations of the second order ODEs system into $2N$.
From Eq. 3.6:

$$\frac{\partial^2 r_i(t)}{\partial t^2} = -\frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{K}(t)r_i(t) + \frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{F}^{\mathbf{ext}}(t) \quad \forall i \in \{0, ..., N-1\}$$

Lets define $\forall i \in \{0, ..., N-1\}$

$$u_i(t) = r_i(t)$$
$$v_i(t) = \frac{dr_i(t)}{dt}$$

then, $\forall i \in \{0, ..., N-1\}$

$$\frac{du_i(t)}{dt} = v_i(t)$$
$$\frac{dv_i(t)}{dt} = \frac{d^2 r_i(t)}{dt^2} = -\frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{K}(t)u_i(t) + \frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{F}^{\mathbf{ext}}(t)$$

These equations can be reordered into the following first order ODEs system:

$$\begin{pmatrix} \frac{d\mathbf{u}(t)}{dt} \\ \frac{d\mathbf{v}(t)}{dt} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{K}(t) & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}(t) \\ \mathbf{v}(t) \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{\rho_0}\mathbf{M}^{-1}\mathbf{F}^{\mathbf{ext}}(u(t)) \end{pmatrix} \tag{3.13}$$

3.2.5 Initial conditions

Let $s \in [0, L] \subset \mathbb{R}$ denote the spatial parameter along the mooring line, with $L > 0$ being the total length of the line. The initial configuration of the mooring line 3.4 is given by a mapping

$$\mathbf{r}(0, s) = g(s) : [0, L] \to \mathbb{R}^3,$$

where $g(s)$ describes the initial position in space of each material point along the mooring line. The initial velocity of the mooring line is assumed to be zero:

$$\frac{\partial \mathbf{r}}{\partial t}(0, s) = \mathbf{0}.$$

Remark 3.7. The function $g(s)$ corresponds to the static equilibrium configuration of the mooring line. Thus, the determination of $g(s)$ reduces to solving a static equilibrium problem.

The static equilibrium configuration $g(s)$ is computed by solving the nonlinear system of equations associated with the balance of internal and external forces at the discrete nodal points.

Problem 3.1 (Static Equilibrium)**.** *Let $\mathbf{r} \in \mathbb{R}^{3N}$ be the concatenated vector of nodal positions of the mooring line, defined as*

$$\mathbf{r} = \left(r_{0,x}, r_{0,y}, r_{0,z}, \ldots, r_{N-1,x}, r_{N-1,y}, r_{N-1,z}\right)^\top,$$

where N is the number of nodes, and the subscripts x, y, z refer to the spatial components. Find $\mathbf{r}^* \in \mathbb{R}^{3N}$ such that

$$\mathbf{F}(\mathbf{r}^*) = \mathbf{0},$$

where $\mathbf{F} : \mathbb{R}^{3N} \to \mathbb{R}^{3N}$ is the nonlinear vector-valued function representing the net force acting on each node.

Newton's Method for Static Equilibrium
The solution $\mathbf{r}^*$ is computed using Newton's iterative method, defined as follows:

Algorithm 3.3.1 (Newton's Method for Static Equilibrium)

- **Initialization:** Choose an initial guess $\mathbf{r}^{(0)} \in \mathbb{R}^{3N}$, typically corresponding to a catenary shape between the anchor point and the fairlead. Set $k = 0$.
- **Iteration:** For $k = 0, 1, 2, \ldots$, compute the update direction $\mathbf{d}^{(k)} \in \mathbb{R}^{3N}$ by solving the linear system:
$$\mathbf{JF}(\mathbf{r}^{(k)})\mathbf{d}^{(k)} = -\mathbf{F}(\mathbf{r}^{(k)}),$$
where $\mathbf{JF}(\mathbf{r}^{(k)}) \in \mathbb{R}^{3N\times 3N}$ is the Jacobian matrix of $\mathbf{F}$ at $\mathbf{r}^{(k)}$.
- **Update:**
$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} + \mathbf{d}^{(k)}.$$
- **Convergence criteria:** The iteration is stopped if either of the following conditions is satisfied:
$$\|\mathbf{r}^{(k+1)} - \mathbf{r}^{(k)}\| < \varepsilon_{\text{abs}} = 10^{-6} \quad \text{(absolute tolerance)},$$
$$\frac{\|\mathbf{r}^{(k+1)} - \mathbf{r}^{(k)}\|}{\|\mathbf{r}^{(k)}\|} < \varepsilon_{\text{rel}} = 10^{-3} \quad \text{(relative tolerance)}.$$
- **Termination:** The algorithm terminates if convergence is achieved or if $k \geq k_{\max} = 30$, in which case the method is considered to have failed to converge.

Definition 3.4 (Jacobian Approximation by Finite Differences)**.** The Jacobian $\mathbf{JF}(\mathbf{r}^{(k)}) \in \mathbb{R}^{3N\times 3N}$ is approximated using finite differences as:

$$\mathbf{JF}(\mathbf{r}^{(k)}) \approx \left[\frac{\mathbf{F}(\mathbf{r}^{(k)} + h\mathbf{e}_1) - \mathbf{F}(\mathbf{r}^{(k)})}{h} \;\cdots\; \frac{\mathbf{F}(\mathbf{r}^{(k)} + h\mathbf{e}_{3N}) - \mathbf{F}(\mathbf{r}^{(k)})}{h}\right],$$

where $\{\mathbf{e}_j\}_{j=1}^{3N}$ is the canonical basis of $\mathbb{R}^{3N}$, and $h = 10^{-12}$ is the finite difference increment.

Remark 3.8 (Dimension of the System). The nonlinear system consists of $3N$ equations corresponding to the balance of forces in three spatial dimensions at each of the N nodes. The vector $\mathbf{r} \in \mathbb{R}^{3N}$ describes the unknown positions of these nodes.

Proposition 3.4 (Convergence of Newton's Method)**.** *If $\mathbf{F} : \mathbb{R}^{3N} \to \mathbb{R}^{3N}$ is continuously differentiable in a neighborhood of $\mathbf{r}^*$, and $\mathbf{JF}(\mathbf{r}^*)$ is nonsingular, then Newton's method converges quadratically to $\mathbf{r}^*$, provided the initial guess $\mathbf{r}^{(0)}$ is sufficiently close to $\mathbf{r}^*$.*

3.2.6 Solving the first order ODEs system: BDF2

The system of ordinary differential equations (ODEs) considered in Eq. 3.13 has the general form:

$$\frac{dy}{dt} = f(y, t), \quad y(0) = y_0, \tag{3.14}$$

where $y \in \mathbb{R}^n$ is the state vector, and $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ represents the dynamics of the system. The Backward Differentiation Formula of Second Order (BDF2) is a second-order

accurate implicit linear multistep method. It approximates the derivative at t_{k+1} using the values y_{k-1}, y_k, and y_{k+1}.
For variable time steps $h_{k-1} = t_k - t_{k-1}$ and $h_k = t_{k+1} - t_k$, the general form of the BDF2 scheme, as implemented in OASIS, is given by:

$$h_k^2 y_{k-1} - (h_{k-1} + h_k)^2 y_k + h_{k-1}(h_{k-1} + 2h_k) y_{k+1} - h_k h_{k-1}(h_{k-1} + h_k) f(y_{k+1}, t_{k+1}) = 0$$

This equation is nonlinear in y_{k+1} because the function f depends on y_{k+1}. To solve it, a Newton-Raphson iterative method is used.

3.2.7 Newton-Raphson Method for BDF2

The nonlinear residual is defined as:

$$F(y_{k+1}) = h_k^2 y_{k-1} - (h_{k-1}+h_k)^2 y_k + h_{k-1}(h_{k-1}+2h_k) y_{k+1} - h_k h_{k-1}(h_{k-1}+h_k) f(y_{k+1}, t_{k+1})$$

Problem 3.2. *Find the unknown y_{k+1} at the next time step such that:*

$$F(y_{k+1}) = 0 \tag{3.15}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear residual function, dependent on y_{k+1}.

The Jacobian matrix of F with respect to y_{k+1} is:

$$JF(y_{k+1}) = h_{k-1}(h_{k-1} + 2h_k) I - h_k h_{k-1}(h_{k-1} + h_k) J_f(y_{k+1}, t_{k+1}) \tag{3.16}$$

where J_f is the Jacobian of f with respect to y computed with finite differences:

$$J_f(y,t) = \frac{1}{\delta} \left[f(y + \delta e_1, t) - f(y,t), \quad f(y + \delta e_2, t) - f(y,t), \quad \dots, \quad f(y + \delta e_n, t) - f(y,t) \right] \tag{3.17}$$

where:

- δ is the finite difference increment, typically chosen as $\delta = 10^{-8}$.
- e_i is the i-th canonical unit vector in $\mathbb{R}^n$.
- $f(y,t)$ is the right-hand side of the ODE system.

The Newton-Raphson method linearizes $F(y)$ around the current guess $y_{k+1}^{(i)}$:

$$F(y_{k+1}^{(i)} + dy^{(i)}) \approx F(y_{k+1}^{(i)}) + JF(y_{k+1}^{(i)}) dy^{(i)} \tag{3.18}$$

and then solves for $dy^{(i)}$ such that the linearised residual vanishes:

$$JF(y_{k+1}^{(i)}) \cdot dy^{(i)} + F(y_{k+1}^{(i)}) = 0 \tag{3.19}$$

Algorithm: Newton-Raphson Method: The Newton-Raphson method is used to find y_{k+1} that solves 3.15 iteratively. Starting from the initial guess $y_{k+1}^{(0)} = y_k$, the following steps are performed at each iteration i:

1. **Evaluate the residual:**
$$F(y_{k+1}^{(i)})$$

2. **Compute the Jacobian matrix:**
$$JF(y_{k+1}^{(i)}) \text{ as in 3.16}$$

3. **Problem 3.3.** *Solve the linear system for the correction $dy^{(i)}$:*
$$JF(y_{k+1}^{(i)}) \cdot dy^{(i)} = -F(y_{k+1}^{(i)}) \tag{3.20}$$

4. **Update the approximation:**
$$y_{k+1}^{(i+1)} = y_{k+1}^{(i)} + dy^{(i)} \tag{3.21}$$

5. **Check for convergence:** The iteration stops if:
$$\|dy^{(i)}\| \leq atol + rtol \cdot \|y_{k+1}^{(i+1)}\|, \tag{3.22}$$
and optionally:
$$\|F(y_{k+1}^{(i+1)})\| \leq atol \tag{3.23}$$

Purpose of $dy^{(i)}$: Finding $dy^{(i)}$ is the core of the Newton-Raphson process. It gives both:

- The **direction** in which to move to reduce the residual $F(y_{k+1})$.
- The **magnitude** of the step we should take from the current approximation.

3.2.8 Adaptive Time Stepping

The local truncation error (LTE) is estimated as described in 2.8:

$$LTE \approx h_k^2(h_{k-1} + h_k) \cdot y_{k+1}^{(3)},$$

where $y_{k+1}^{(3)}$ is an approximation of the third derivative of y as described in 2.9.
The new time step h_{k+1} is calculated by:

$$h_{k+1} = \sigma \cdot h_k, \tag{3.24}$$

with the scaling factor σ given by:

$$\sigma = \left(\sqrt{n} \cdot \left|\frac{EWT}{LTE}\right|\right)^{1/3}, \tag{3.25}$$

where $EWT = rtol \cdot \|y\| + atol$ is the error weight tolerance.
The value of h_{k+1} is limited between user-defined minimum and maximum values, h_{min} and h_{max}, and adjusted to align with output times.

3.2.9 Initialization with BDF1

The BDF2 method is a two-step method. To compute the solution at time t_{k+1}, it requires the two previous solutions: y_{k-1} and y_k. Knowing the initial condition $y(0) = y_0$, the first step is computed with the backward Euler method (BDF1):

$$\frac{y_1 - y_0}{h_0} = f(y_1, t_1) \tag{3.26}$$

or equivalently,

$$y_1 - y_0 - h_0 f(y_1, t_1) = 0 \tag{3.27}$$

This equation is solved via Newton-Raphson iteration, using the same procedure described above.

1. Define the residual:
$$F(y_1^{(i)}) = y_1^{(i)} - y_0 - h_0 f(y_1^{(i)}, t_1). \tag{3.28}$$

2. Compute the Jacobian:
$$JF(y_1^{(i)}) = I - h_0 J_f(y_1^{(i)}, t_1), \tag{3.29}$$
where $J_f(y_1^{(i)}, t_1)$ is the Jacobian matrix of f with respect to y.

3. Solve the linear system:
$$J_F(y_1^{(i)}) \cdot dy^{(i)} = -F(y_1^{(i)}). \tag{3.30}$$

4. Update the solution:
$$y_1^{(i+1)} = y_1^{(i)} + dy^{(i)}. \tag{3.31}$$

5. Repeat until convergence:
$$\|dy^{(i)}\| \leq atol + rtol \cdot \|y_1^{(i+1)}\|. \tag{3.32}$$

4 Coupling with a elasto-plastic hysteresis model

4.1 Tension model

The challenge of this work lies in the tension term, expressed as a function of the strain. The materials used for the mooring lines present hysteresis, i.e., they are viscoelastic: the relationship between the tension and the strain is path-dependent, meaning that the loading and unloading cycles do not follow the same trajectory (Fig. 2). This results in energy dissipation, which must be accounted for in the modelling of the system. The tension in the mooring lines depends not only on the instantaneous strain but also on the strain history, adding complexity to the problem.
To model the viscoelastic behavior of materials with memory, Banks et al. [18] proposed a constitutive law based on the Boltzmann superposition principle. This approach accounts for how the stress at a given time depends not only on the current strain but also on its entire history.
When modelling, time is discretized as $0 \leq t_0 < t_1 < \ldots < t_N$, and at each time point, the strain $\varepsilon(t)$ and its rate $\dot{\varepsilon}(t)$ are known or computed. The viscoelastic response is

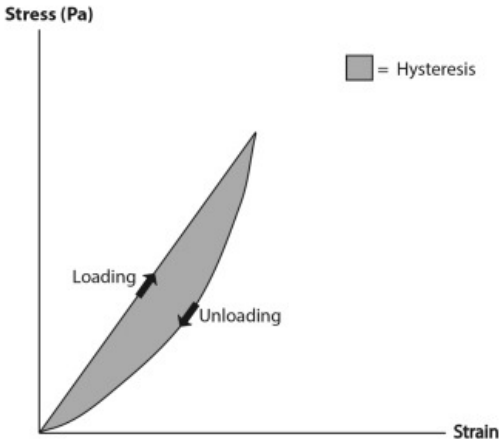


Figure 2: A schematic stress-strain curve of a viscoelastic material, showing hysteresis: dissipation of energy.

described by nonlinear functions $g_e(\varepsilon)$ and $g_v(\varepsilon, \dot{\varepsilon})$, representing the instantaneous elastic and rate-dependent viscoelastic contributions, respectively.
For a single step change in strain applied at time t_0, the stress response over the time interval $t_0 < t < t_1$ is given by

$$\sigma(t) = Y(t - t_0) \cdot g_v(\varepsilon(t_0), \dot{\varepsilon}(t_0)),$$

where $Y(t - t_0)$ is the relaxation function (or memory kernel), characterizing how the material's memory fades over time.
When additional strain changes occur at subsequent times, the Boltzmann Superposition Principle states that the total stress is the sum of the effects of all previous strain increments, weighted by the relaxation kernel. In the continuous case, this principle leads to the following integral form of the constitutive law:

$$\sigma(t) = g_e(\varepsilon(t)) + \int_0^t Y(t - s) \frac{d}{ds} g_v(\varepsilon(s), \dot{\varepsilon}(s)), ds \tag{4.1}$$

In this equation:

- $g_e(\varepsilon(t))$ models the instantaneous elastic response.
- The convolution integral accounts for the history-dependent viscoelastic stress, where the derivative $\frac{d}{ds} g_v(\varepsilon(s), \dot{\varepsilon}(s))$ represents the rate at which the viscoelastic contribution evolves with time.
- $Y(t - s)$ is the memory kernel or relaxation function, describing how past strain events influence the current stress.

This formulation captures the nonlinear nature of the material response, since both g_e and g_v can be nonlinear functions of strain and strain rate.
Furthermore, Banks et al. proposed the following viscoelastic function, where the loading and unloading responses are different:

$$g_v(\varepsilon(s), \dot{\varepsilon}(s)) = \begin{cases} g_{vi}(\varepsilon(s)) & \text{if} \quad \dot{\varepsilon}(s) > 0 \\ g_{vd}(\varepsilon(s)) & \text{if} \quad \dot{\varepsilon}(s) < 0 \end{cases} \tag{4.2}$$

These material´s memory significantly depends on a history of finite length. If we define the turning points t_i, $i = 0, ..., M$ as the points for which $\dot{\varepsilon} = 0$, and assume $t_k < t < t_{k+1}$ for $0 \leq k < M$. Since g_v need not be continuous, derivatives must be interpreted in the sense of distributions.
Integration by parts is done, taking

$$u = Y(t-s) \Longrightarrow du = -\dot{Y}(t-s) \text{ and } dv = \frac{d}{ds} g_v(\varepsilon(s))ds \Longrightarrow v = g_v(\varepsilon(s))ds$$

the viscoelastic term in Eq. 4.1 results in

$$\int_0^t Y(t-s)\frac{d}{ds} g_v(\varepsilon(s), \dot{\varepsilon}(s))ds = Y(t-s)g_v(\varepsilon(s), \dot{\varepsilon}(s))\Big|_0^t - \int_0^t -\dot{Y}(t-s)g_v(\varepsilon(s), \dot{\varepsilon}(s))ds =$$

$$Y(0)g_v(\varepsilon(t), \dot{\varepsilon}(t)) - Y(t)g_{vi}(\varepsilon(0)) + \sum_{k=0}^{M} Y(t-t_k)(-1)^k \left[g_{vi}(\varepsilon(t_k)) - g_{vd}(\varepsilon(t_k))\right] +$$

$$\int_0^t \dot{Y}(t-s)g_v(\varepsilon(s), \dot{\varepsilon}(s))ds$$

and so Eq. 4.1 results in

$$\begin{aligned} \frac{T(t)}{A} = \sigma(t) = g_e(\varepsilon(t)) &+ \int_0^t \dot{Y}(t-s)g_v(\varepsilon(s), \dot{\varepsilon}(s))ds \\ &+ Y(0)g_v(\varepsilon(t), \dot{\varepsilon}(t)) - Y(t)g_{vi}(\varepsilon(0)) \\ &+ \sum_{k=0}^{M} Y(t-t_k)(-1)^k [g_{vi}(\varepsilon(t_k)) - g_{vd}(\varepsilon(t_k))] \end{aligned} \tag{4.3}$$

where A is the cross-sectional area.
Furthermore, the following functions were proposed:
• A third degree polynomial for the elastic component,

$$g_e(\varepsilon(t)) = \sum_{i=1}^{3} E_i \varepsilon^i(s) \tag{4.4}$$

• A third degree polynomial for the viscoelastic component,

$$g_{vi}(\varepsilon(s)) = \sum_{i=1}^{3} a_i \varepsilon^i(s) \tag{4.5}$$

$$g_{vd}(\varepsilon(s)) = \sum_{i=1}^{3} b_i \varepsilon^i(s) \tag{4.6}$$

• A decay of exponential type for the memory kernel,

$$Y(t) = e^{-Ct} \tag{4.7}$$

The aim is to couple this hysteresis model for the simulation of mooring lines in floating marine structures.

4.2 Implementation considerations

The development and implementation of the elasto-plastic mooring line simulation model presented several challenges that had to be carefully addressed to ensure numerical stability, accuracy, and efficiency. The primary difficulties encountered during this work include numerical instabilities at zero crossing times, the need for an additional damping term, issues with buffer management and convergence, precision problems in strain rate detection, and computational efficiency concerns. This subsection provides a detailed discussion of each difficulty and the strategies employed to overcome them.

1. **Numerical instability at the zero crossing times.** Since initial conditions are not perfect, small vibrations occur when going from the static initial conditions to the dynamic simulation. Because of this, during the early stages of the simulation, numerical instabilities were observed when computing the zero crossing times t_i, where $\dot{\varepsilon} = 0$. These instabilities caused oscillations in the computed tension, leading to physically unrealistic results.
   To mitigate this issue, a smoothing function was introduced at the beginning of the simulation. By gradually increasing the applied forces rather than applying them instantaneously, the model could stabilize the strain rate evolution and reduce sudden jumps in the numerical solution.

2. **Implementation of a Kelvin-Voigt damping term for stability.**
   To improve numerical stability, a Kelvin-Voigt damping term of the form $\beta\varepsilon$ was added to the constitutive law. This term acts as a dissipative force that smooths out high-frequency oscillations, preventing excessive strain rate fluctuations. The coefficient β needs to be carefully calibrated to provide damping without excessively altering the physical behavior of the system. This adjustment proved particularly useful at the start of the simulation, where abrupt transitions often caused numerical problems.

3. **Buffer management and convergence issues.**
   The simulation required a buffer system to store past and present steps, which is essential for computing the Jacobian matrix in the time integration scheme. However, an unexpected issue arose when the buffer continued calling past values even after numerical convergence had been achieved. This resulted in persistent fluctuations in the computed tension, making it difficult to determine a stopping criterion.
   To address this, the algorithm was modified by quitting instants after or equal to the current function call. It is called with previous steps because when convergence is not achieved, the time step is repeated with a smaller dt. This way, once convergence was achieved, the buffer was prevented from making redundant calls, ensuring a smooth transition to the next time step.

4. **Precision issues in detecting strain rate sign changes.**
   One of the most critical aspects of modeling the hysteresis effect is detecting changes in the strain rate $\dot{\varepsilon}$, as these transitions define the turning points t_i in the material's response. Initial implementations showed inconsistencies when determining these sign changes, particularly when the strain rate approached zero but fluctuated due to numerical round-off errors.

To solve this, a tolerance threshold $tol_{zero} = 1 \times e - 9$ was introduced. Instead of strictly detecting $\dot{\varepsilon} = 0$, the algorithm now considers the small interval around zero $-tol_{zero} \leq \dot{\varepsilon} \leq tol_{zero}$ as a valid transition region. This adjustment significantly improved the robustness of the detection algorithm, ensuring that switching between loading and unloading regimes was handled correctly.

5. **Computational efficiency and cost considerations.**
   Given the complexity of the finite element formulation, the simulation of long-duration mooring line behavior required significant computational resources. The adoption of a Boltzmann-type stress-strain relation involves time-history dependent terms. As a result, at each timestep, the numerical scheme must evaluate integral expressions that depend on past strain history, and solve a nonlinear system of equations due to the elasto-plastic constitutive law. These computations increase the CPU time per timestep, particularly in long simulations where memory access and accumulation of past data (if not optimized) may create bottlenecks. Since BDF2 is an implicit method, each time step also requires a Newton–Raphson solver, involving Jacobian evaluations or approximations, further adding to the cost.
   To improve efficiency, the OASIS model required significant extension to couple the new tension law, such as designing new class attributes to represent the evolving internal state (e.g., strain memory) and force calculations. Proper memory management and efficiency were essential due to the long-term simulation goals (tens of thousands of time steps with fine spatial resolution).

Overcoming these challenges was essential for achieving a stable, accurate, and computationally efficient simulation of mooring lines with elasto-plastic behavior. The combination of numerical damping, improved strain rate detection, buffer management, and computational optimizations ensured that the model could reliably capture the mechanical response of mooring lines under realistic conditions.

5 Verification and validation

5.1 Verification with Banks

In order to verify the implementation of the elasto-plastic tension model in OASIS, the experiments made by Banks et al. [18] for the uniaxial tension of a viscoelastic material with memory effects have been reproduced. These tests consist of simulating the dynamic response of a discrete chain under uniaxial cyclic loading, where the stress-strain relationship exhibits hysteresis due to nonlinear viscoelastic behavior. The same experiment has been previously analyzed and discussed in detail in the author's Final Degree's Thesis in Physics [19], to which the reader is referred for the full derivation of the model parameters and the physical interpretation of the results.

In order to discretize this problem, a simple three-node mooring line was simulated, subjected to a displacement z-axis displacement at one end while the opposite end remains fixed. This setup allows isolating the material's intrinsic response from external hydrodynamic or geometric effects. The internal tension is governed by the nonlinear stress-strain law proposed by Banks 4.3. The material parameters used in these tests were taken from

[18], ensuring consistency. These include the coefficients of the third-degree polynomials defining g_e, g_{vi} , and g_{vd}, and the exponential decay parameter of the kernel $Y(t)$.

Figure 3 shows the simulated stress-strain curves obtained from the implemented model for several loading-unloading cycles. The typical hysteresis loop associated with energy dissipation is clearly observed, and the qualitative features of the response closely match those reported by Banks et al. The correct reproduction of turning points and the asymmetry between loading and unloading phases confirms that the nonlinear memory effects have been correctly incorporated.

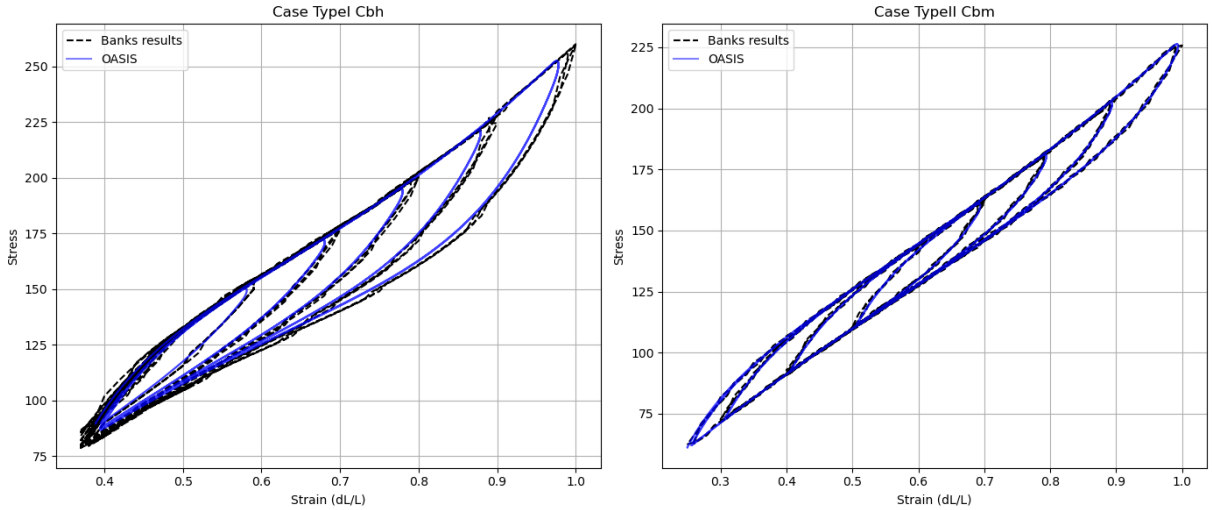


Figure 3: Comparison between the stress-strain curves obtained with the implemented model in OASIS (solid blue lines) and the reference signals from Banks et al. (dashed black lines). The left plot corresponds to Case Type I Cbh, while the right plot corresponds to Case Type II Cbm.

5.2 Validation: catenary

The catenary validation serves as a qualitative evaluation of the model's stability and physical plausibility in a realistic scenario, where quantitative experimental data were unavailable.
The simulation aims to reproduce the behavior of a chain attached to the seabed. The objective is to validate the mechanical accuracy of the implemented finite element model under static equilibrium conditions. To achieve this, a composite mooring line is constructed, consisting of two distinct materials:

- 4 meters of steel chain, representing a heavy and relatively inextensible segment, typically used near the anchor to ensure that part of the line remains on the seabed under low tension.
- 1 meter of polymer rope, which is lighter and more elastic, commonly used for its ability to absorb dynamic loads near the floating structure (e.g., buoy or vessel).

The combined 5-meter line is fixed at one end (representing the seabed anchor) and held at an elevated position at the other end (representing the fairlead or connection to the floating structure). In the simulation, the external force is applied along the horizontal axis (x direction) at the fairlead point. This simulates a scenario where horizontal tension is induced by the floating body due to current, wind, or wave forces. Gravity and buoyancy effects are included in the vertical direction (z-axis), ensuring that the line naturally adopts a catenary shape, the classic equilibrium curve under gravity.
In this validation test, two different approaches have been tested for the polymer rope:

- Elastic model (from 4.4): This linear elastic approximation simplifies the tension-strain relationship and assumes no energy dissipation during loading and unloading.
- Hysteresis model: This approach incorporates nonlinear elasto-plastic behavior with memory effects, accounting for energy dissipation during cyclic loading, as observed in real mooring chains.

The parameters used correspond to those of Cbh, which can be found in [18]. They have been increased by a factor of $e3$ so that the rigidity was realistic, making physical sense.

Figure 4 shows both the tension at the Fairlead and the whole line position. The simulations demonstrate how different material behaviors affect the overall line configuration.
Elastic case (orange):

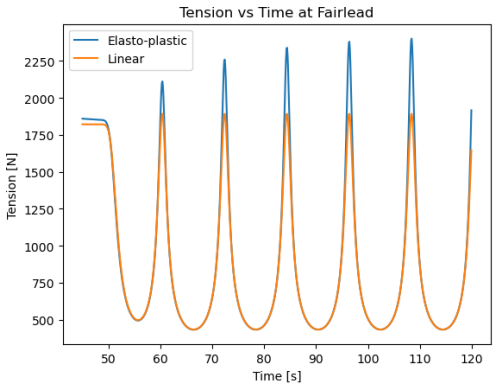
- For the same displacement, the line with hysteresis dissipates more energy because the area inside the curve is larger. The elastic line also presents "hysteresis" because energy is dissipated by hydrodynamic drag, the friction of the water with the chain, even though there is no elastic hysteresis.
- It follows a shape close to the ideal catenary, expected from a line that deforms elastically and conservatively.

Hysteresis case (blue):

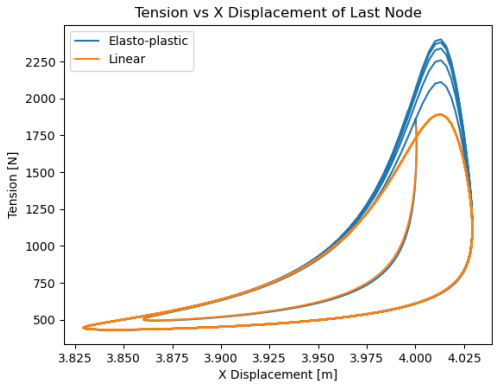
- The mooring line exhibits more pronounced curvature and sag.
- The hysteresis model shows greater tension under the same loading.
- The effect of energy dissipation and material memory can be appreciated since as the cycles pass, the curve stabilizes. This simulates how real mooring systems deform under cyclic loads (due to wave or platform motion).

For the simulation of the linear model, 720 thousand total function calls were made, as well as 257 total Jacobian calls. And for the simulation of the hysteresis model, 2.9 million total function calls were made (four times more), as well as 5930 total Jacobian calls (twenty-three times more). The computational cost increases not only because of the memory usage, but also because a shorter time step is required for the hysteresis model to be stable.

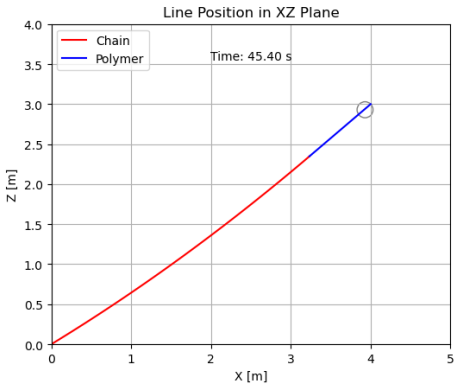
While the results align with theoretical expectations (e.g., catenary geometry, hysteresis loops), the lack of empirical data prevents quantitative validation (e.g., exact tension values or displacement magnitudes). However, the model's stability under dynamic loads and its consistency with mechanical principles support its reliability for practical applications.



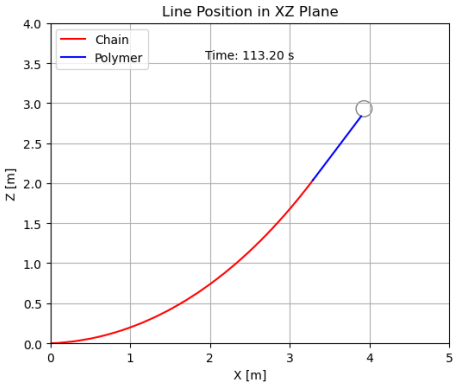
(a)



(b)



(c) Initial line position, static equilibrium, for the hysteresis model.



(d) Line position under applied circular movement at the fairlead at $t = 113.20$s for the hysteresis model

Figure 4: The simulations replicate a catenary setup, where a composite mooring line composed of a heavy, inextensible chain segment and a lighter, more elastic polymer rope is suspended between a fixed anchor and an elevated fairlead point.

6 Conclusion and future work

This project has developed a mathematical and computational model for the simulation of mooring lines subjected to dynamic marine conditions, with a particular focus on elasto-plastic behavior. By formulating the problem through nonlinear partial differential equations (PDEs) and employing the Finite Element Method (FEM), the study bridges theoretical mathematical principles with real-world engineering applications. The inclusion of adaptive BDF2 time integration has allowed for the handling of stiff systems while maintaining numerical stability and efficiency.

The main achievement of this work lies in the successful implementation of a new tension model that incorporates hysteresis-based elasto-plastic behavior. This allowed the simulation to realistically capture material memory and energy dissipation effects, which are fundamental to accurately represent the dynamics of marine mooring lines. The integration of this model into the OASIS model had significant technical challenges, such as numerical instabilities, numerical integration and coupling the non-linear tension term,

but was ultimately achieved with promising results in terms of accuracy. A critical part of the validation process was the comparison between the elastic, linear, model and the elasto-plastic, non-linear, hysteresis model. This comparison demonstrated that while the constant stiffness model produces a more rigid and idealized configuration, the hysteresis model is a more physically accurate that reflects realistic behavior under cyclic loading. This result not only validates the model's mechanical consistency but also confirms that the new implementation significantly enhances the simulation's fidelity to real marine environments.

This project highlights the importance of Sobolev spaces and weak formulations in the numerical treatment of PDEs, especially when dealing with nonlinearity and irregular boundary conditions. The use of Newton-Raphson methods for both static and dynamic systems, along with adaptive step size control, proved essential for the stability and convergence of the simulations.

Future research could focus on:

- Enhancing the numerical stability of the current model in order to reduce computational cost without compromising accuracy.
- Another important direction is the incorporation of creep effects into the model, which account for permanent damages suffered in the material. This phenomena is especially relevant in cables subjected to sustained loads or high temperatures. Adding a creep term to the model would introduce more non-linear formulation. [20]
- Additionally, experimental validation of the proposed catenary models would significantly strengthen the credibility of the computational approach. Controlled laboratory experiments or scaled physical models could be designed to compare displacement, tension, and dynamic responses with those predicted by the simulation.
- Furthermore, future extensions could also include coupling the catenary dynamics with environmental interactions, such as wind loads or water currents, and exploring their effects on fatigue and vibration.

In summary, this project demonstrates how rigorous mathematical theory can be translated into practical computational tools with direct relevance to engineering. The results obtained are not only promising for the simulation of mooring systems but also serve as a foundation for future research in applied mathematics, numerical analysis, and marine engineering.

References

[1] Rodríguez Luis, "Finite element method for underwater cable dynamics," 2018.

[2] P. Desiré Valdor, "Analysis and numerical simulation of mooring lines in complex bathymetries," 2022.

[3] L. Evans, *Partial Differential Equations*. 2 ed., 2010.

[4] H. Brezis, *Functional Analysis, Sobolev Spaces and Partial Differential Equations*, pp. 107–108. New York, NY: Springer New York, 2011.

[5] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, vol. 15 of *Texts in Applied Mathematics*. New York: Springer, 3 ed., 2008.

[6] D. Braess, *Conforming Finite Elements*, p. 27. Cambridge University Press, 2007.

[7] *The Mathematical Theory of Finite Element Methods.*, pp. 136–137. New York, NY: Springer New York, 2008.

[8] W. H. Kendall Atkinson and D. Stewart, *Numerical Solution of Ordinary Differential Equations*, ch. 8, p. 130. 2009.

[9] G. W. Ernst Hairer, *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems, 2nd ed.*, p. 42. 1996.

[10] U. o. C. Arieh Iserles, *A First Course in the Numerical Analysis of Differential Equations, 2nd ed.* 2008.

[11] E. A. Celaya, J. J. A. Aguirrezabala, and P. Chatzipantelidis, "Implementation of an adaptive bdf2 formula and comparison with the matlab ode15s," *Procedia Computer Science*, vol. 29, pp. 1014–1026, 2014. 2014 International Conference on Computational Science.

[12] O. Aamo and T. Fossen, "Finite element modelling of mooring lines," *Mathematics and Computers in Simulation*, vol. 53, no. 4, pp. 415–422, 2000.

[13] J. Azcona, X. Munduate, L. González, and T. A. Nygaard, "Experimental validation of a dynamic mooring lines code with tension and motion measurements of a submerged chain," *Ocean Engineering*, vol. 129, pp. 415–427, 2017.

[14] A. Montano, M. Restelli, and R. Sacco, "Numerical simulation of tethered buoy dynamics using mixed finite elements," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 41, pp. 4117–4129, 2007.

[15] P. Trubat, C. Molins, and X. Gironella, "Wave hydrodynamic forces over mooring lines on floating offshore wind turbines," *Ocean Engineering*, vol. 195, p. 106730, 2020.

[16] G. Li, D. Wang, and B. Zhu, "Well-posedness and decay of solutions for a transmission problem with history and delay," *Electronic Journal of Differential Equations*, vol. 2016, 01 2016.

[17] A. Burden, R. Burden, and J. Faires, *Numerical Analysis, 10th ed.* 2016.

[18] H. T. Banks, G. A. Pintér, L. K. Potter, M. J. Gaitens, and L. C. Yanyo, "Modeling of nonlinear hysteresis in elastomers under uniaxial tension," *Journal of Intelligent Material Systems and Structures*, vol. 10, no. 2, pp. 116–134, 1999.

[19] M. González Alonso, "Modelización de histéresis en materiales viscoelásticos," 11 2024.

[20] L. Hai, S. qing Wang, and W. cheng Liu, "Modeling creep response for hmpe ropes by a viscoelastic damage model based on fractional derivative theory," *Ocean Engineering*, vol. 298, p. 117181, 2024.

A Appendix

- Oasis is a code implemented in C++ with approximately 38,000 lines of code.
- For this work, the code was compiled and executed in a Linux environment on the IHCantabria cluster.
- Its versions are managed using Git with GitHub. For the development of this project, work was done on a specific branch.
- The generation of inputs and the graphical representation of outputs were performed using Python scripts developed by the author.
- The implementation of the elastoplasticity model in Oasis involved the addition of approximately 600 new lines of code by the author, including reading new input data and calculating the elasto-plastic stress term.This implementation was carried out by creating new methods and modifying existing ones in pre-existing classes, due to the object-oriented nature of C++ and OASIS. Additionally, memory was managed using pointers, and new attributes were declared within the same classes.

List of Figures