POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

High-Performance Computing for Fractional Rheology

Laurea Magistrale in High Performance Computing Engineering - Ingegneria del Calcolo ad Elevate Prestazioni

Author: Lorenzo Fonnesu, Andrea Grassi

Advisor: Dr. Alessandra Bonfanti

Co-advisors: Prof. Alexandre Kabla, Dr. Jonathan Louis Kaplan

Academic year: 2025-2026

1. Introduction

Soft materials such as polymers, hydrogels and biological tissues respond to mechanical loading in a time-dependent way, typically dissipating stress gradually according to a power-law trend; these materials, which exhibit characteristics that are a combination of those of elastic solids and viscous fluids, are known as *viscoelastic*. This behaviour is relevant across very different applications: predicting how a hydrogel deforms under physiological load matters for tissue-scaffold and drug-delivery design, while the long-term creep of asphalt pavements informs infrastructure maintenance planning. Fractional calculus, which extends differentiation and integration to non-integer order, captures this multi-scale dissipative behaviour with a small number of parameters and underlies fractional viscoelastic models [1].

Despite their descriptive power, fractional models are rarely adopted in engineering practice because their evaluation requires high computational costs: the constitutive response depends on the entire loading history through a convolution integral, which scales quadratically in time and linearly in memory, and the associated relaxation kernel is expressed through the Mittag-Leffler function, whose evaluation is itself slow and numerically delicate. Both issues compound when the model must be evaluated thousands of times during parameter fitting or at every integration point of a finite-element simulation, which is precisely where the bottleneck becomes prohibitive.

This thesis addresses that bottleneck by reformulating fractional viscoelastic models in a differential, time-domain framework that discretizes the governing equation directly, replacing the convolution-based evaluation with an approach suited to high-performance computing. Four objectives follow from this aim: (i) develop and benchmark numerical schemes for fractional derivatives; (ii) apply the resulting differential approach to parameter identification; (iii) apply it to the prediction of stress and strain under arbitrary loading; and (iv) release the methods as open-source tools to be integrated into already existing dedicated libraries such as RHEOS [2], one of the main open-source frameworks designed for viscoelastic material simulations.

1.1. Linear Viscoelasticity

In this work, we focus exclusively on linear viscoelasticity, which applies to all materials whose

stress $\sigma(t)$ and strain $\varepsilon(t)$ functions are linearly related [1]. More specifically, the *Boltzmann Superposition Principle* relates an arbitrary strain (or stress) history to the resulting stress (or strain) through a convolution with the relaxation modulus $G(t)$ (or creep compliance $J(t)$):

$$\sigma(t) = \int_0^t G(t-\tau)\,\frac{d\varepsilon(\tau)}{d\tau}\,d\tau, \tag{1}$$

$$\varepsilon(t) = \int_0^t J(t-\tau)\,\frac{d\sigma(\tau)}{d\tau}\,d\tau. \tag{2}$$

While these integrals provide a rigorous mathematical description of memory effects, their direct numerical evaluation acts as the primary source of the computational bottlenecks addressed in this work, as will be discussed later.

1.2. Classical Viscoelastic Models

Real-life materials can be represented using conceptual networks analogous to electrical circuits, where two basic components model the fundamental elastic and viscous properties of the matter. These components can be arranged in series or parallel configurations to dictate the overall behavior of the system.

In this framework, the two integer-order building blocks are:

- **The Hookean Spring:** Represents purely elastic energy storage ($\sigma(t) = k\varepsilon(t)$).
- **The Newtonian Dashpot:** Represents purely viscous energy dissipation ($\sigma(t) = \eta\frac{d\varepsilon(t)}{dt}$).

By combining these basic components, we construct the simplest multi-element configurations: parallel networks, called Kelvin-Voigt model, and series networks, called Maxwell model. Each configuration yields a unique governing constitutive equation that relates the macroscopic stress $\sigma(t)$ and strain $\varepsilon(t)$ through the internal laws of the circuit.

For example, the Maxwell model governing differential equation is

$$\dot{\varepsilon}(t) = \frac{\dot{\sigma}(t)}{k} + \frac{\sigma(t)}{\eta} \tag{3}$$

Despite their simplicity, these classical integer-order models are often inadequate for real-world soft materials.

1.3. Fractional Viscoelastic Models

The underlying mathematics of ordinary differential equations restricts their material response to pure exponential functions (e.g., $e^{-t/\tau}$). In contrast, complex polymers and biological tissues inherently exhibit broad-spectrum relaxation that decays according to a power-law trend ($t^{-\alpha}$).

To capture this non-exponential behavior using classical elements, one would need to arrange an infinite or impractical number of springs and dashpots into generalized networks (such as the Generalized Maxwell model). However, this approach is fundamentally unfeasible for engineering practice, as it introduces an overwhelming number of parameters, leading to severely ill-conditioned optimization loops during material characterization.

Fractional calculus resolves this trade-off by introducing the *springpot* as a generalized, non-integer circuit component. The springpot interpolates directly between purely elastic and purely viscous behaviors, its governing equation is

$$\sigma(t) = c_\beta \frac{d^\beta \varepsilon(t)}{dt^\beta}, \quad \text{with } 0 \le \beta \le 1, \tag{4}$$

where c_β represents the material consistency coefficient and β is the fractional order. When $\beta = 0$, the element acts as a pure Hookean spring; when $\beta = 1$, it becomes a pure Newtonian dashpot. For any intermediate value ($0 < \beta < 1$), a single springpot intrinsically captures continuous relaxation spectra and power-law memory using only two parameters.

By replacing the classical springs and dashpots with the springpot, we construct fractional-order differential models, such as the Fractional Kelvin-Voigt and Fractional Maxwell frameworks. While these systems maintain a compact parameter set, the combination of multiple springpots structurally couples the fractional operators on both sides of the constitutive equation. Specifically, the Fractional Maxwell model is governed by the following relation:

$$\sigma(t) + \frac{c_\alpha}{c_\beta}\frac{d^{\alpha-\beta}\sigma(t)}{dt^{\alpha-\beta}} = c_\alpha \frac{d^\alpha \varepsilon(t)}{dt^\alpha}. \tag{5}$$

While these systems maintain a compact parameter set, solving their governing equations via the traditional analytical convolution approach requires the mathematical introduction of the

transcendental *Mittag-Leffler function* ($E_{a,b}(z)$), defined as

$$E_{a,b}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(ak+b)}. \tag{6}$$

The Mittag-Leffler function functions as a generalized exponential, smoothly transitioning the material response from an initial fast power-law decay to a slower asymptotic behavior. However, because this function is defined as an infinite series, its numerical evaluation introduces severe instabilities and overhead, directly establishing the primary computational challenges addressed in the following section.

1.4. Computational Challenges

The first main issue in fractional calculus is *non-locality*: unlike integer-order derivatives, which depend only on a small neighbourhood of points, the fractional derivatives in Eq. (1) and (2) require the entire history of the signal, giving linear memory complexity and quadratic time complexity in the number of time steps. The second bottleneck is the *Mittag-Leffler function*, which is used in many cases to compute $G(t)$ and $J(t)$ and whose evaluation, even with optimized quadrature and asymptotic expansions, remains numerically unstable and slow. The differential approach proposed in this thesis removes the Mittag-Leffler bottleneck by never evaluating the function during execution, while the non-local property is mitigated, rather than removed, through the acceleration strategies described in Section 2.

2. Numerical Methods for Fractional Derivatives

In this work, rather than computing the convolution integral shown in Eq. (1) and (2), which would require a further calculation of the Mittag-Leffler function, we focus instead on directly handling with the generalized constitutive equation of fractional viscoelastic models, defined as:

$$\sum_i a_i \frac{d^{\alpha_i}\sigma(t)}{dt^{\alpha_i}} = \sum_j b_j \frac{d^{\beta_j}\varepsilon(t)}{dt^{\beta_j}} \tag{7}$$

where a_i and b_j are material coefficients and α_i, $\beta_j \in [0,1]$ represent the fractional derivative orders.

The differential approach we used, based on the Finite Difference Method, discretizes fractional derivatives directly in the time domain, allowing us to set the foundation for the main focus of this work, that is parameter estimation from experimental data and, following, the prediction of mechanical response under arbitrary loading conditions. All of the following methods were consolidated into `NumFracDiff` [3], an open-source Julia library dedicated to the high performance evaluation of fractional derivatives.

2.1. Baseline Implementation

Three finite difference schemes were implemented and compared: Grünwald–Letnikov (GL), Caputo, and Riemann–Liouville (RL). For the GL scheme, the derivative at step n is approximated as

$${}^{GL}_{t_0}D^{\alpha}_t f_n \approx \frac{1}{\Delta t^{\alpha}} \sum_{j=0}^{n} \omega_j^{(GL)} f_{n-j}, \tag{8}$$

where the weights $\omega_j^{(GL)}$ are generalized binomial coefficients computed through an $O(1)$ recurrence rather than the Gamma function [4]. All three methods were validated against analytical solutions, demonstrating that GL is more accurate than the other two. As to performance evaluation, GL and RL proved to be approximately $10\times$ faster than Caputo. All implementations were then parallelized across threads by exploiting the independence of the derivative evaluations at distinct time indices, which entirely avoids race conditions, and *SIMD* processing. Results are shown in Figure 1.
Due to its superior accuracy and favorable algebraic structure for convolution, subsequent optimization strategies were focused exclusively on the GL scheme.

2.2. Short Memory Principle

Because Eq. (8) still sums over the full history, two complementary acceleration strategies were added on top of the GL scheme. The *Short Memory Principle* [5], illustrated in Figure 2, truncates the summation to the most recent L steps, the minimum window for which the discarded tail contribution stays below a prescribed tolerance ϵ_0.
To compensate for the loss of accuracy caused by truncation error, the result can be corrected

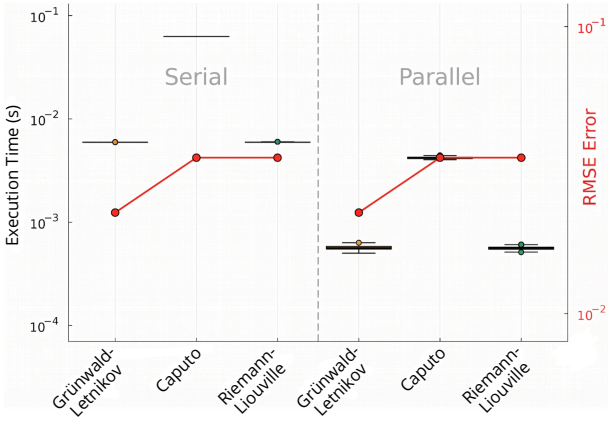

Figure 1: Comparison between baseline implementations (left: serial version; right: parallel version with 32 threads) using a Unit Step input function.

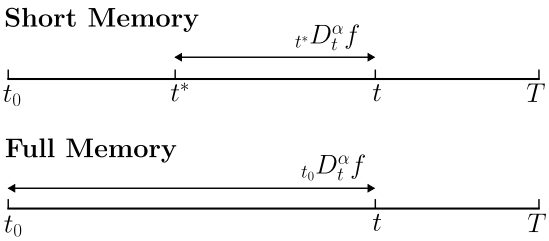

Figure 2: Visual comparison between Short (top) and Full (bottom) Memory approaches.

with a first-order correction term evaluated at the edge of the memory window, without further increasing the computational cost (this last implementation is referred in this work as High Precision Short Memory).

2.3. Fast Fourier Transform

In parallel with the implementation of the Short Memory Principle, the convolution structure of Eq. (8) was reformulated using the Fast Fourier Transform (FFT), exploiting the convolution theorem to evaluate the full-history sum in $O(N \log N)$ instead of $O(N^2)$; since this reformulation preserves the exact causal convolution, its accuracy is fundamentally identical to the non-accelerated baseline, with no trade-off between speed and precision.

2.4. Performance Evaluation

Performance experiments showed that both Short Memory variants and the FFT implementation roughly halved the execution time of the full-memory parallel baseline, whilst - with the exception of the implementation of the short

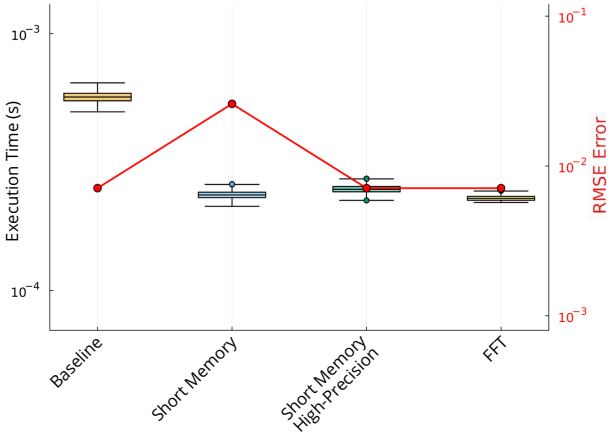

Figure 3: Comparison between each Grünwald-Letnikov implementation (32 threads) using Step input function.

memory principle without error correction – all methods have the same error, as illustrated in Figure 3. Regarding scalability, Figure 4 shows that the parallel implementations follow Amdahl's law, with strong scaling holding closely up to 16 threads before plateauing, and a 32-thread speedup of $\sim 11\times$ over the serial baseline.

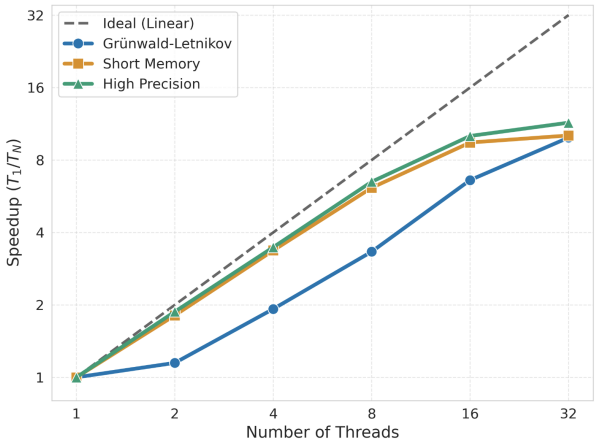

Figure 4: Strong scaling of the three parallel implementations of the Grünwald–Letnikov method: baseline, short memory and high-precision short memory.

3. Parameter Fitting

The first application focuses on identifying optimal material parameters from experimental stress and strain data. This inverse problem is resolved by formulating a quantitative cost function. The discrete time-series are evaluated through the model's constitutive equation us-

ing the high-perfomance fractional derivatives library introduced in the Section 2, and the resulting residual error is minimized using the Mean Squared Error (MSE) metric. This function serves as the objective target for a nonlinear optimization loop managed via the *NLopt* library. Both solvers are implemented as extensions of RHEOS' `modelfit` method, so that existing RHEOS-based workflows require no interface changes to benefit from them.

To validate these performance trends in full optimization scenarios, complete parameter fitting procedures were benchmarked across various models. The results demonstrate that while performance remains comparable for integer-ordered classical models, the proposed parallel differential and FFT-based implementations deliver a massive operational advantage on fractional-ordered derivatives setups. Specifically, for the Fractional Maxwell model, where convolution approach is severely bottlenecked by the repeated evaluation of the Mittag-Leffler function, the newly proposed methods achieved a 20× speedup, reducing total fitting times by 95% while simultaneously converging to an RMSE error one order of magnitude lower than the standard convolution approach, as shown in Figure 5. Finally, a preliminary analysis using the `LIKWID` suite quantifies the energy consumption and carbon emissions associated with the code execution. In these complex scenarios, the proposed differential and FFT-accelerated methods reduce both energy demands and carbon emissions by over 91%.

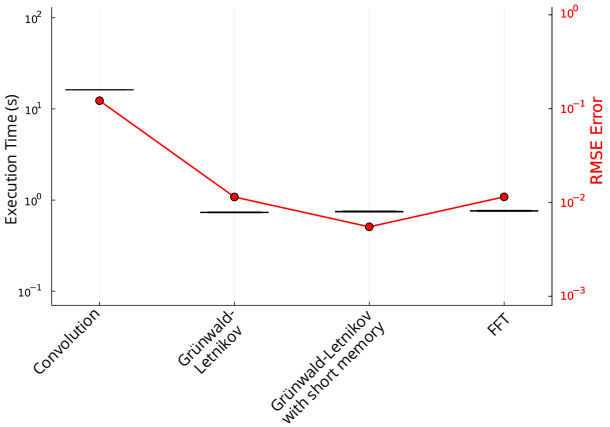

Figure 5: Average time and Error comparisons of the fitting process between Convolution approach and newly introduced approaches.

4. Response Prediction

The second application is the forward prediction of stress or strain under arbitrary loading histories, relevant to finite-element simulations where the constitutive response must be evaluated at every integration point and time step. Two complementary solvers were developed from Eq. (8):

- a *finite-difference time-domain solver* based on the Grünwald-Letnikov fractional derivative, which, owing to its step-by-step nature, is particularly well-suited when the input variable is not known in full but is provided during the process.
- an *FFT-based convolution solver*, capable of calculating the output vector in a single computation given the entire input signal.

Both methods are implemented as extensions of RHEOS' `modelpredict` method; therefore, as was the case previously with fitting, there is no need to modify the library interface.

The two solvers were then validated against exact analytical solutions for various inputs, with the GL and FFT error curves overlapping in every tested scenario, confirming their mathematical equivalence. Regarding performance evaluation, Figures 6 and 7 show that the computational advantage scales with model complexity: for the classical Maxwell model, RHEOS's original convolution-based approach remains competitive due to the simplicity of the model, which does not include fractional terms; for a more complex model as the Fractional Zener, which involves multiple fractional derivatives in both stress and strain, the GL solvers achieve $\sim 5\times$ and $\sim 10\times$ speedups over convolution (respectively the 1-thread and 32-thread versions), while FFT reaches up to 100×. Because the time horizons used for prediction are comparatively short, the absolute energy savings are smaller than for fitting, although the differential and FFT methods still reduce energy use and carbon footprint by up to 31% for the Fractional Zener model.

5. Conclusions

This thesis addresses the fundamental computational bottlenecks inherent in the convolutional framework for fractional viscoelastic models by proposing an alternative differential, time-domain approach. In particular, the primary outcome of this work is the devel-

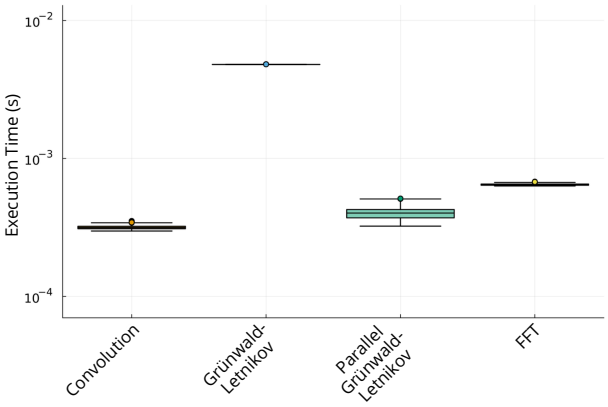

Figure 6: Average execution times for stress prediction using the standard integer-order Maxwell model.

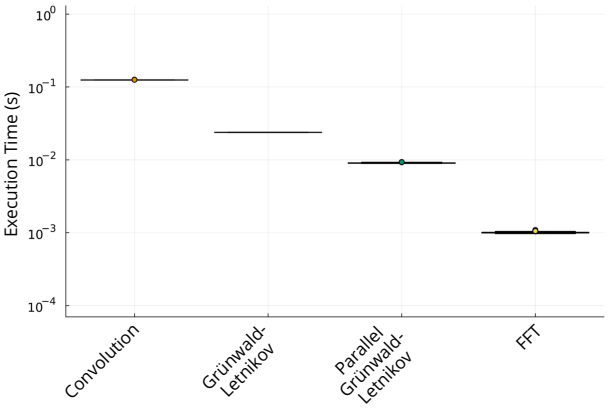

Figure 7: Average execution times for stress and strain prediction using the Fractional Zener model.

opment of `NumFracDiff`, which contains the finite-difference schemes discussed in Section 2. This external package was engineered from the ground up to seamlessly integrate with existing rheological frameworks, such as RHEOS. Using these new numerical tools, the second major contribution involved extending the RHEOS library to natively support both the differential and FFT-based approaches. This architectural extension enables the framework to leverage these advanced numerical methods to solve complex parameter fitting and forward prediction problems in a more efficient way than the original approach, eliminating the need to compute the Mittag-Leffler function and reducing the overall computational cost required.

Several directions remain open for future work. Porting the core routines to GPU architectures would allow the framework to handle the larger datasets and longer time horizons that are currently intractable on CPU clusters alone. A dedicated optimization routine, tailored to the structure of Eq. (7), could remove the current dependency on the general-purpose NLopt library and improve convergence speed and robustness. Another major practical development will involve the integration of these high-performance backends into Finite Element Method (FEM) solvers, which are currently bottlenecked by convolution evaluations at every integration point. Machine-learning techniques could be then integrated to automate constitutive-model selection directly from raw experimental data, prior to fitting. Finally, the theoretical framework could be extended to non-linear viscoelastic models, which are notoriously more complex than their linear counterparts.

References

[1] Alessandra Bonfanti, Jonathan Louis Kaplan, Guillaume Charras, and Alexandre Kabla. Fractional viscoelastic models for power-law materials. *Soft matter*, 16(26): 6002–6020, 2020.

[2] Jonathan Louis Kaplan, Alessandra Bonfanti, and Alexandre Kabla. Rheos. jl-a julia package for rheology data analysis. *arXiv preprint arXiv:2005.02538*, 2020.

[3] Lorenzo Fonnesu, Andrea Grassi, Alexandre Kabla, and Alessandra Bonfanti. Numfracdiff. `https://github.com/JuliaRheology/NumFracDiff`, 2026.

[4] Igor Podlubny. *Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications*, volume 198. elsevier, 1998.

[5] Changpin Li and Fanhai Zeng. *Numerical methods for fractional calculus*. CRC Press, 2015.