The Journal of Open Source Software

Cocofest: an Open-Source Python Package for Functional Electrical Stimulation Optimization in Optimal Control

Kevin Co [1], **Pierre Puchaud** [1,2], **Florent Moissenet** [1,3,4], **and Mickaël Begon** [1]

1 Laboratoire de Simulation et Modélisation du Mouvement, Université de Montréal, Montréal, Québec, Canada **2** Auctus, Inria, Centre de l'Université de Bordeaux, Talence, France **3** Biomechanics Laboratory, Geneva University Hospitals and University of Geneva, Geneva, Switzerland **4** Kinesiology Laboratory, Geneva University Hospitals and University of Geneva, Geneva, Switzerland

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 06 September 2025
Published: unpublished

Summary

Functional electrical stimulation (FES) is a rehabilitation method intended to promote motor recovery notably after neurological impairment. Applying coordinated electrical pulses to muscles elicits functional movements like walking, reaching, and grasping. FES rehabilitation mostly relies on empirical settings, as responses to stimulation vary across populations and muscles. Empirical settings often cause overstimulation and premature fatigue (Ibitoye et al., 2016), shortening rehabilitation sessions and diminishing therapeutic benefit. Consequently, advanced control approaches like optimal control-driven FES are gaining interest in personalizing and improving FES rehabilitation efficiency, meanwhile delaying muscle fatigue. To address this need, we designed `Cocofest` (Custom Optimal COntrol for Functional Electrical STimulation), an open-source Python package for optimal control-driven FES. `Cocofest` provides a framework to generate personalized pulse trains (Fig. 1) based on nonlinear dynamics models for FES (Table. 1), for several musculoskeletal models and motor tasks. The package provides more than 10 examples, covering optimization of FES-related pulse train parameters (including frequency, pulse width, pulse intensity), FES model parameters identification from in vivo measurements, and long duration predictive simulations.

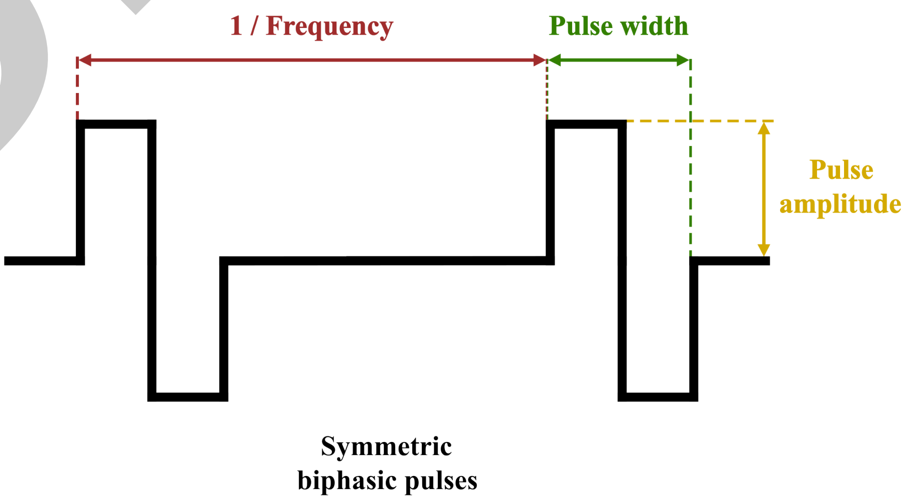

Figure 1: Pulse train parameters that can be optimized in Cocofest

Statement of Need

Since the pioneer study on optimal control-driven FES (Hunt et al., 1997), no code has been shared in the field, limiting objective comparison and replicability across studies. The lack of open-source practice led to an absence of consensus on how to choose nonlinear dynamics for FES, and which cost functions to use for dedicated clinical needs, hindering standardization and cumulative progress (Co et al., 2025). To address these challenges and support collective scientific progress, `Cocofest` fulfills the following four needs:

Firstly, the relationship between the pulse train parameters (e.g., frequency, pulse width and intensity; Fig. 1) and the resulting muscle force, joint torque, and muscle fatigue (termed as state variables) can be modeled with different nonlinear dynamics (Ding et al., 2003; Veltink et al., 1992) (Table 1). Gathering them within a unified package would facilitate comparison for more informed modelling choices.

Secondly, no study has compared different optimal control problem (OCP) formulations applied to FES, due to OCP implementation challenges (Co et al., 2025). Easily customizable OCP formulation, involving objective functions, models, and transcriptions is required to provide an adequate research framework. Having the possibility to switch between several OCP transcriptions, such as direct collocation or direct multiple shooting, is essential when dealing with stiff differential equations (Puchaud et al., 2023), often embed in FES models. Muscle fatigue is the primary challenge in FES. Enabling the development and comparison of different OCP formulations could help address research questions, yield novel stimulation patterns and enhance fatigue reduction. Moreover, using receding-horizon estimation for longer simulations reduces the computational complexity associated with time-varying dynamics (e.g., fatigue) (Ding et al., 2003).

Thirdly, predictive simulations of FES-driven or FES-assisted motions (e.g., walking, cycling, reaching, and grasping) require the coupling of FES models with the equations of motion as well as adequate muscle force-length-velocity relationships. Predictive simulations are usually actuated through Hill-type muscle models (Wakeling et al., 2023). A package capable of replacing muscle actuation by FES models in multibody musculoskeletal models will allow us to simulate realistic FES-driven tasks.

Fourthly, personalized rehabilitation strategy is required to facilitate the motor recovery. Therefore, identifying the patient-specific muscle response to FES is a crucial step. Unfortunately, current complex identification methods are a barrier to clinical translation (Le et al., 2010). Providing a robust and customizable framework for the development of more patient-friendly protocols would help to overcome this barrier.

Overall, despite its potential, optimal control–driven FES remains unadopted in clinical practice due to its low technology readiness level (Co et al., 2025). `Cocofest` is a comprehensive package designed to bridge the gaps and foster clinical adoption. It integrates nonlinear muscle dynamics dedicated to FES, manages muscle fatigue, interfaces FES with musculoskeletal models, supports customizable cost functions and parameter identification routines. With the goal of bringing this technology to patient care, we believe this package will contribute to the open-science effort. `Cocofest` is expected to accelerate the increase of technology readiness level by strengthening knowledge foundation.

Functionality and Features

`Cocofest` already integrates six FES muscle dynamics from the literature (Table 1).

Table 1: FES models in Cocofest

Name	Purpose	States	Controls
Veltink et al. (1992)	Joint angle control	Activation	Pulse intensity
Riener et al. (1996)	Predict fatigue	Fatigue	None*
Ding et al. (2003)	Isometric force control with fatigue	Calcium, force, force scaling factor, cross-bridges sensitivity, time to force decline	Frequency
Ding et al. (2007)	Isometric force control	Calcium, force	Frequency, pulse width
Marion et al. (2009)	Force control with fatigue for a motion	Calcium, force	Frequency
Marion et al. (2013)	Force control with fatigue for a motion	Calcium, force, force scaling factor, cross-bridges sensitivity, time to force decline	Frequency, pulse width
Hmed et al. (2018)	Isometric force control	Calcium, force	Frequency, pulse intensity

* *Only the muscle fatigue prediction was implemented. The model is used in combination with Veltink et al. (1992).*

`Cocofest` relies on Biorbd, a musculoskeletal physics engine (Michaud & Begon, 2021), and Bioptim, an open-source optimization framework for biomechanical problems (Michaud et al., 2022). Specifically, Bioptim enables easy OCP customization including cost functions, bounds, constraints, transcription methods (e.g., direct collocation), integration methods, and solving methods (e.g., full- and receding-horizon OCPs).

In conventional Hill-type muscle model, muscle force (F_m) is the product of a the muscle activation, F_{max} the maximal isometric muscle force, f_l the force-length, f_v the force-velocity and f_{pas} the passive force-length relationship: $F_m(t) = a(t)\, F_{\max}\, f_l(\tilde{l}_m)\, f_v(\tilde{v}_m) + f_{pas}(\tilde{l}_m)$. `Cocofest` replaces $a(t) \times F_{max}$ by the force obtained using FES models (e.g., Ding, Marion, Hmed). This approach allows motions driven-FES simulations, meanwhile benefiting from musculoskeletal model properties (e.g., muscle insertion, weight, inertial).

An identification feature is available to personalize FES models based on experimental data. Model's parameters (e.g., rested force scaling factor, cross-bridges sensitivity, and time to force decline in (Ding et al., 2003) model) are personalized by minimizing the difference between the simulated and the experimental forces.

Additionally, a feature for solving initial value problems was implemented to enable model comparison. The FES nonlinear dynamics is integrated forward in time to simulate the model's behavior from given initial state and controls (i.e., series of pulse trains). `Cocofest` also incorporates the recent numerical truncation method to speed up convergence (Coelho-Magalhães et al., 2025). This method limits the number of past stimulations considered in the dynamics to reduce the dependency on time-varying states.

An optimization example: Pulse width optimization to match a force profile using the Ding et al. (2007) model

This example shows how to optimize a FES pulse width using `Cocofest`, coupled with bioptim (Michaud et al., 2022) version 3.3.0.

```python
import numpy as np
from bioptim import (ControlType, ObjectiveFcn, ObjectiveList, OdeSolver,
                     OptimalControlProgram, Node, SolutionMerge)
from cocofest import ModelMaker, OcpFes, FesModel


def prepare_ocp(model: FesModel,
                final_time: float,
                pw_max: float,
                force_tracking: list) -> OptimalControlProgram:
    """
    Prepare the Optimal Control Program by setting dynamics, bounds and cost functions.

    Parameters
    ----------
    model : DingModelPulseWidthFrequency
        The chosen FES model to use as muscle dynamics.
    final_time : float
        The ending time for the simulation.
    pw_max : float
        The maximum pulse width, used for stimulation bounds.
    force_tracking : list
        The force to track.

    Returns
    -------
    ocp : OptimalControlProgram
        The Optimal Control Program to solve.
    """
    # --- Set dynamics --- #
    # Create the number of shooting points for the OCP
    n_shooting = model.get_n_shooting(final_time=final_time)
    time_series, stim_idx_at_node_list = model.get_numerical_data_time_series(
        n_shooting, final_time
    )  # Retrieve time and indexes at which occurs the stimulation for the FES dynamic
    dynamics = OcpFes.declare_dynamics(
        model,
        time_series,
        ode_solver=OdeSolver.RK4(n_integration_steps=10),
        # Possibility to use a different solver
        # ode_solver=OdeSolver.COLLOCATION(polynomial_degree=3, method="radau"),
    )

    # --- Set initial guesses and bounds for states and controls --- #
    x_bounds = OcpFes.set_x_bounds(model)
    x_init = OcpFes.set_x_init(model)
    u_bounds = OcpFes.set_u_bounds(model, max_bound=pw_max)
    u_init = OcpFes.set_u_init(model)
```

```python
    # --- Set objective functions --- #
    objective_functions = ObjectiveList()
    # Reshape list to track to match Bioptim's target size
    force_to_track = force_tracking[np.newaxis, :]
    objective_functions.add(
        ObjectiveFcn.Mayer.TRACK_STATE,
        key="F",
        target=force_to_track,
        node=Node.ALL,
        quadratic=True,
    )

    return OptimalControlProgram(
        bio_model=[model],
        dynamics=dynamics,
        n_shooting=n_shooting,
        phase_time=final_time,
        objective_functions=objective_functions,
        x_init=x_init,
        x_bounds=x_bounds,
        u_bounds=u_bounds,
        u_init=u_init,
        control_type=ControlType.CONSTANT,
        n_threads=20,
    )

def main():
    final_time = 1
    stim = 33
    model = ModelMaker.create_model("ding2007",
                                    stim_time=list(np.linspace(0, 1, stim,
                                                               end-point=False)))

    # --- Building force to track ---#
    time = np.linspace(0, 1, 34)
    # Example of force to track between 10 and 150 N
    force = 10 + (150 - 10) * np.abs(np.sin(time * 5))
    force[0] = 0.0  # Ensuring the force starts at 0 N

    ocp = prepare_ocp(model=model,
                      final_time=final_time,
                      pw_max=0.0006,
                      force_tracking=force)
    sol = ocp.solve()

    # --- Show the optimization results --- #
    sol.graphs()


if __name__ == "__main__":
    main()
```

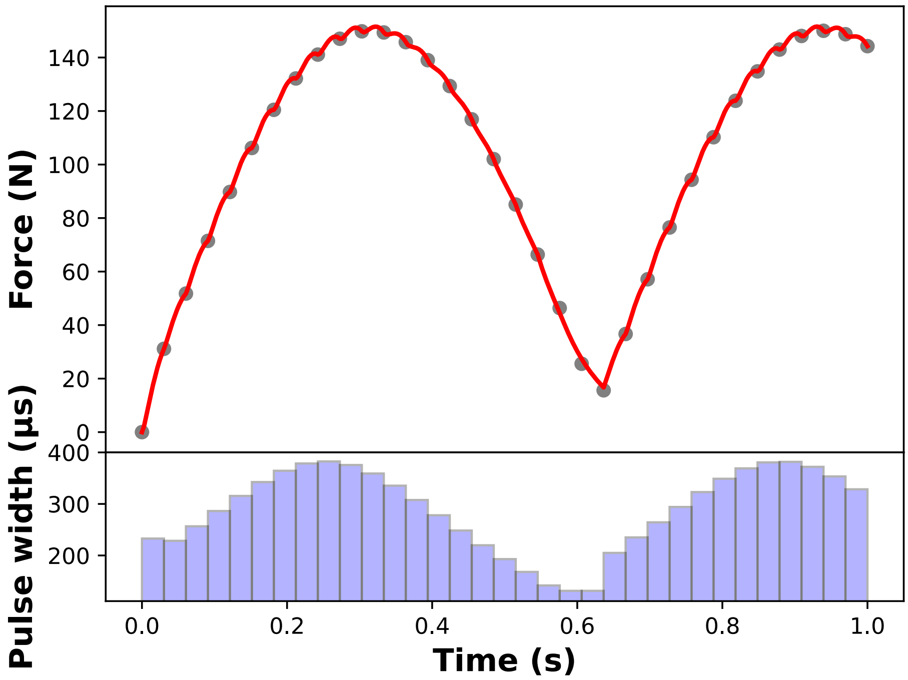

Figure 2: Tracked force (grey dots) and optimized force generated (red) by pulse width optimization (blue).

Acknowledgements

The package development was supported by the Fonds de recherche du Québec – Nature et technologies (FRQNT, Grant 341023) and by the FRQ strategic group in Ingénierie de technologies interactives en réadaptation (INTER #160 OptiStim).

References

Co, K., Begon, M., Bailly, F., & Moissenet, F. (2025). Optimal control driven functional electrical stimulation: A scoping review. *arXiv Preprint arXiv:2508.02899*. https://doi.org/10.48550/arXiv.2508.02899

Coelho-Magalhães, T., Azevedo-Coste, C., & Bailly, F. (2025). Numerical-optimal-control-compliant muscle model for electrically evoked contractions. *IEEE Transactions on Medical Robotics and Bionics*. https://doi.org/10.1109/TMRB.2025.3590453

Ding, J., Chou, L.-W., Kesar, T. M., Lee, S. C., Johnston, T. E., Wexler, A. S., & Binder-Macleod, S. A. (2007). Mathematical model that predicts the force–intensity and force–frequency relationships after spinal cord injuries. *Muscle & Nerve: Official Journal of the American Association of Electrodiagnostic Medicine*, *36*(2), 214–222. https://doi.org/10.1002/mus.20806

Ding, J., Wexler, A. S., & Binder-Macleod, S. A. (2003). Mathematical models for fatigue minimization during functional electrical stimulation. *Journal of Electromyography and Kinesiology*, *13*(6), 575–588. https://doi.org/10.1016/S1050-6411(03)00102-0

Hmed, A. B., Bakir, T., Garnier, Y. M., Sakly, A., Lepers, R., & Binczak, S. (2018). An approach to a muscle force model with force-pulse amplitude relationship of human quadriceps muscles. *Computers in Biology and Medicine*, *101*, 218–228. https://doi.org/10.1016/j.compbiomed.2018.08.026

Hunt, K. J., Munih, M., & Donaldson, N. de N. (1997). Feedback control of unsupported standing in paraplegia. I. Optimal control approach. *IEEE Transactions on Rehabilitation Engineering*, *5*(4), 331–340. https://doi.org/10.1109/86.650287

Ibitoye, M. O., Hamzaid, N. A., Hasnan, N., Abdul Wahab, A. K., & Davis, G. M. (2016). Strategies for rapid muscle fatigue reduction during FES exercise in individuals with spinal cord injury: A systematic review. *PloS One*, *11*(2), e0149024. https://doi.org/10.1371/journal.pone.0149024

Le, F., Markovsky, I., Freeman, C. T., & Rogers, E. (2010). Identification of electrically stimulated muscle models of stroke patients. *Control Engineering Practice*, *18*(4), 396–407. https://doi.org/10.1016/j.conengprac.2009.12.007

Marion, M. S., Wexler, A. S., & Hull, M. L. (2013). Predicting non-isometric fatigue induced by electrical stimulation pulse trains as a function of pulse duration. *Journal of Neuroengineering and Rehabilitation*, *10*, 1–16. https://doi.org/10.1186/1743-0003-10-13

Marion, M. S., Wexler, A. S., Hull, M. L., & Binder-Macleod, S. A. (2009). Predicting the effect of muscle length on fatigue during electrical stimulation. *Muscle & Nerve: Official Journal of the American Association of Electrodiagnostic Medicine*, *40*(4), 573–581. https://doi.org/10.1002/mus.21459

Michaud, B., Bailly, F., Charbonneau, E., Ceglia, A., Sanchez, L., & Begon, M. (2022). Bioptim, a python framework for musculoskeletal optimal control in biomechanics. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *53*(1), 321–332. https://doi.org/10.1109/TSMC.2022.3183831

Michaud, B., & Begon, M. (2021). Biorbd: A c++, python and matlab library to analyze and simulate the human body biomechanics. *Journal of Open Source Software*, *6*(57), 2562. https://doi.org/10.21105/joss.02562

Puchaud, P., Bailly, F., & Begon, M. (2023). Direct multiple shooting and direct collocation perform similarly in biomechanical predictive simulations. *Computer Methods in Applied Mechanics and Engineering*, *414*, 116162. https://doi.org/10.1016/j.cma.2023.116162

Riener, R., Quintern, J., & Schmidt, G. (1996). Biomechanical model of the human knee evaluated by neuromuscular stimulation. *Journal of Biomechanics*, *29*(9), 1157–1167. https://doi.org/10.1016/0021-9290(96)00012-7

Veltink, P. H., Chizeck, H. J., Crago, P. E., & El-Bialy, A. (1992). Nonlinear joint angle control for artificially stimulated muscle. *IEEE Transactions on Biomedical Engineering*, *39*(4), 368–380. https://doi.org/10.1109/10.126609

Wakeling, J. M., Febrer-Nafría, M., & De Groote, F. (2023). A review of the efforts to develop muscle and musculoskeletal models for biomechanics in the last 50 years. *Journal of Biomechanics*, *155*, 111657. https://doi.org/10.1016/j.jbiomech.2023.111657