The Journal of Open Source Software

BioMAC-Sim-Toolbox: A MATLAB toolbox for biomechanical motion analysis and creation through simulation

Anne D. Koelewijn [1,2¶], **Marlies Nitschke** [1], **Eva Dorschky** [1], **Markus Gambietz** [1], **Alexander Weiss** [1], **Bjoern M. Eskofier** [1], and **Antonie J. van den Bogert** [2]

1 Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany **2** Cleveland State University, Cleveland, Ohio, United States of America ¶ Corresponding author

DOI: 10.21105/joss.07945

Software

- Review
- Repository
- Archive

Editor: Prashant Jha

Reviewers:

- @rmndrs89
- @Aravind-Sundararajan

Submitted: 09 December 2024
Published: 16 November 2025

Summary

We present a MATLAB toolbox for human movement simulation and analysis. The toolbox's focus is on creating simulations of human walking and running by solving trajectory optimization problems. In these problems, the muscle excitations and initial state are found that minimize an objective related to energy expenditure. To solve the trajectory optimization problems, we have implemented a direct collocation framework. Different objectives have been implemented that represent energy or effort minimization. We have also included different objectives for data tracking, including tracking of joint angles, ground reaction forces, accelerometer data, gyroscope data, and marker positions. We have implemented two different two-dimensional (2D) musculoskeletal models and one three-dimensional (3D) musculoskeletal model. To show how the toolbox can be used, we include a tutorial, a 2D and a 3D introductory example, as well as different applications that are based on previous publications. The toolbox is versatile due to its object-oriented design, such that different dynamics models can be easily combined with different problem classes. In the future, we aim to implement different other problem classes as well, such as inverse kinematics and dynamics to process experimental data, static and dynamic optimizations to find muscle simulations from joint moments, while also forward shooting can be used to investigate neural control of gait.

Statement of Need

Movements of humans and other animals are extremely versatile, while our efficiency is unmatched by human-made machines, even though humans seemingly have inaccurate and inefficient controllers. Therefore, a better understanding of human movement can have a large impact for persons with movement disabilities, sports performance optimization, and design of human-made devices. Movement simulations are a key tool for creating this understanding. On the one hand, we can use so-called predictive simulations (Ackermann & Van den Bogert, 2010; Falisse et al., 2019; Koelewijn et al., 2018) to investigate unseen movements by replicating the optimization in the central nervous system (Zarrugh et al., 1974) in a computer optimization. Such simulations can help us decipher fundamentals of movement in humans and other animals. On the other hand, we can use so-called reconstructive simulations to estimate variables that cannot or were not measured directly by minimizing a data tracking error (Dorschky, Nitschke, et al., 2019; Nitschke et al., 2023, 2024). These reconstructive simulations do not necessarily require laboratory equipment, instead allowing biomechanical variables to be estimated accurately from wearable or contactless sensors, such as inertial sensors. Widespread measurement of movements and their biomechanical analysis are vital to be able to reach a deep understanding of human movement.

BioMAC-Sim-Toolbox

Here, we present the `BioMAC-Sim-Toolbox`, a MATLAB toolbox that can be used to simulate and analyse human movements. This toolbox is released under an Apache-2.0 license and can be accessed on GitHub: https://github.com/mad-lab-fau/BioMAC-Sim-Toolbox. The main functionality of the toolbox is that it solves trajectory optimization problems, or optimal control problems, for human musculoskeletal dynamics models. These dynamics models combine multibody dynamics to model the movements of the skeleton with muscle dynamics models to model the dynamics of muscular contraction and activation. In the trajectory optimization problem, the muscle excitations and initial state, as well as possibly other gait variables such as speed or duration, are found that minimize an objective. This objective generally includes terms to minimize muscular effort and to minimize a tracking error with respect to measured movement data (Dorschky, Nitschke, et al., 2019; Dorschky, Krüger, et al., 2019; Koelewijn & Selinger, 2022; Koelewijn & Van den Bogert, 2016; Nitschke et al., 2023, 2024), but several additional objectives have been implemented as well, such as minimization of metabolic energy expenditure (Koelewijn et al., 2018). The optimization problem can be described mathematically as follows:

$$\underset{\mathbf{x}(0),\mathbf{u}(t),T}{\text{minimize}} \quad J(\mathbf{x},\mathbf{u}) = \sum_{j=1}^{N_W} W_j \int_{t=0}^{T} c_j(x(t),u(t))\mathrm{d}t + c_f(x(T),u(T))$$

$$\text{subject to} \quad \mathbf{f}(\mathbf{x}(t),\dot{\mathbf{x}}(t),\mathbf{u}(t)) = \mathbf{0} \text{ for } 0 \le t \le T \text{ (system dynamics)}$$

$$\mathbf{x_L} \le \mathbf{x}(t) \le \mathbf{x_U} \text{ for } 0 \le t \le T \quad \text{(bounds on states)}$$

$$\mathbf{u_L} \le \mathbf{u}(t) \le \mathbf{u_U} \text{ for } 0 \le t \le T \quad \text{(bounds on controls)}$$

for a musculoskeletal model with state x and input u. Here, the goal is to find the initial state, $\mathbf{x}(0)$, control inputs, $\mathbf{u}(t)$, and duration, T, that minimize the cost function, $J(\mathbf{x},\mathbf{u})$ while following the system dynamics $\mathbf{f}$. The cost function consists of $c_j(\mathbf{x}(t),\mathbf{u}(t))$, which describes the cost functions that are evaluated over time, which are multiplied with associated weight W_j, and of a final cost function $c_f(\mathbf{x}(T),\mathbf{u}(T))$, which is only evaluated at the end of the trajectory, at time T. We normally do not include a final cost function. The subscript L defines the lower bounds, while U defines the upper bounds. When speed or duration is known, we commonly still include them into the optimization variables, while making the lower and upper bound equal to the desired value.

For walking and running, we include a periodicity constraint to the full gait cycle or a single step, in case of a symmetric motion. This constraint can be defined for straight and curved running and walking (Nitschke et al., 2020) through the displacement in the horizontal plane. This periodicity constraint ensures that the states are the same at the end of the trajectory as at the beginning, while applying a horizontal displacement to the states in $\mathbf{x}_{hor}$:

$$\mathbf{x}(T) = \mathbf{x}(0) + vT\mathbf{x}_{hor} \quad \text{(periodicity)}$$

So far, we have created the problem class `Collocation`, which creates a trajectory optimization problem using direct collocation (Betts, 2010). In direct collocation, the numerical integration of the differential equations that represent the dynamics are solved as equality constraints during the optimization. To do so, collocation nodes, or time nodes, need to be predefined, and the integration method should be selected. We have currently implemented a backward Euler and a midpoint Euler integration method. Backward Euler has been more stable and has therefore been used for publications. A direct collocation approach creates a large-scale nonlinear optimization problem, for which the derivatives of the objectives and all constraint can be derived analytically. An optimization problem including a periodicity constraint could be transcribed as follows after applying direct collocation with backward Euler integration:

$$\underset{\mathbf{x}(i),\mathbf{u}(i),v,T}{\text{minimize}} \quad J(\mathbf{x},\mathbf{u}) = \sum_{j=1}^{N_W} W_j \sum_{i=0}^{N} c_j(\mathbf{x}(i),\mathbf{u}(i)) + c_f(\mathbf{x}(N),\mathbf{u}(N))$$

$$\text{subject to} \quad \mathbf{f}(\mathbf{x}(i+1), \frac{\mathbf{x}(i+1)-\mathbf{x}(i)}{\Delta t}, \mathbf{u}(i)) = \mathbf{0} \text{ for } i = 1,\ldots,N \quad \text{(system dynamics)}$$

$$\mathbf{x_L} \leq \mathbf{x}(i) \leq \mathbf{x_U} \text{ for } i = 1,\ldots,N \quad \text{(bounds on states)}$$

$$\mathbf{u_L} \leq \mathbf{u}(i) \leq \mathbf{u_U} \text{ for } i = 1,\ldots,N \quad \text{(bounds on controls)}$$

$$\mathbf{x}(N+1) = \mathbf{x}(0) + vT\mathbf{x}_{hor} \quad \text{(periodicity)}$$

for N collocation points i. Compared to the description of the optimization problem in continuous time, there are only subtile changes. The integral in the objective has changed into a sum, and we have used the backward Euler transcription in the dynamics. Furthermore, the bounds and the periodicity constraint are now applied to discrete variables.

Problems can be solved using the solvers that are implemented in the class `solver`. So far, we have implemented an interface to IPOPT (Wächter & Biegler, 2006) to solve the resulting optimization problem. This algorithm can solve optimal control problems transcribed with direct collocation in less than 10 minutes for a 2D model, for example on a computer with an Intel Core i5-3210M CPU at 2.5 GHz clock speed (Koelewijn & Selinger, 2022; Koelewijn & Van den Bogert, 2016) and less than one hour for a 3D model on a single core of a workstation with a 3.2GHz Xeon E5-1660v4 processor (Nitschke et al., 2020). Different solvers could easily be integrated into this solver class.

As most analysis is specific to the problem type, most code for analysis can be found in the respective problem class. For example, metabolic cost requires an integration over time, which is dependent on the problem type that is being used. In addition, general methods, e.g., to calculate a correlation or root mean squared error between two variables, or to write results into an OpenSim file format, can be found in the function `HelperFunctions` folder.

We have implemented different human dynamics models (see Table 1). All implemented models are musculoskeletal models, but they can also be used as skeletal models, such that the input is generated using joint moments instead of muscle excitations. We have implemented a 3D model version (`gait3d`) and two 2D model versions in c, which are compiled as mex functions. The model dynamics can be tested using the test cases coded in the `tests` folder. The model parameters of the 3D model and two 2D model (`gait2d_osim` and `gait10dof18musc`) are loaded from OpenSim, but use our own muscle dynamics model described in (Nitschke et al., 2020), while the other 2D model (`gait2dc`) is used and described in our own previous work (Dorschky, Nitschke, et al., 2019; Dorschky, Krüger, et al., 2019; Koelewijn & Selinger, 2022; Koelewijn & Van den Bogert, 2016). The model parameters are defined in the .osim file for the OpenSim models, and defined in an Excel file for model gait2dc (gait2dc_par.xlsx). The models can be personalized by directly adjusting the parameters in the .osim model or Excel file, such that, for example, OpenSim scaling can be used (Seth et al., 2018). They can also still be adjusted in MATLAB, for example to investigate virtual participants (Dorschky, Nitschke, et al., 2019; Koelewijn & Selinger, 2022). The model dynamics are explained further in Koelewijn & Selinger (2022); Dorschky, Nitschke, et al. (2019); Dorschky, Krüger, et al. (2019) for `gait2dc` and in Nitschke et al. (2020) for `gait3d`. The `gait2d_osim model` has been used in Gambietz et al. (2024). It is based on the `gait10dof18musc.osim` model (Seth et al., 2018), and can be loaded in two ways: the original version, called `gait10dof18musc`, and a version with the lumbar joint locked, called `gait2d_osim`.

Table 1: Overview of dynamics models implemented in the BioMAC-Sim-Toolbox

Model name	Dimension	Degrees of freedom	Number of muscles	Arm torque actuators	Source
gait2dc	2D	9	16	-	(Dorschky, Nitschke, et al., 2019; Dorschky, Krüger, et al., 2019; Koelewijn & Selinger, 2022; Koelewijn & Van den Bogert, 2016)
gait2d_osim	2D	9	18	-	(Seth et al., 2018)
gait10dof18musc	2D	10	18	-	(Seth et al., 2018)
gait3d	3D	33	92	10	(Nitschke et al., 2020, 2023, 2024)

Using the Toolbox

To get started, we recommend that users first look at the tutorial in the folder `Tutorial` and the folder `IntroductionExamples` in `ExampleScripts`. The tutorial is created to help users become familiar with the inputs and outputs of the simulations, as well as create and run simulations. To perform the tutorial, users should start by looking at the file TutorialInTheWild.PDF. In addition, we have added an introductory example for 2D simulations (`script2D.m`) and 3D simulations (`script3D.m`), which is based on Nitschke et al. (2020), in the folder `IntroductionExamples`. The goal of these introductory examples is to show how the toolbox works, and highlight different options in the implementation. Therefore, these functions, and those called inside them, have a lot of comments. We recommend running them line by line to follow along with the example.

Besides the introductory examples, we have included several other examples in the folder `ExampleScripts`. Each example can be run independently through a function in respective folder of which the name starts with `script`. They should produce a relevant graph or movie at the end. These examples are added to highlight different ways that the code can be used. The code in the folder `ExoPaper` can be used to generate predictive simulations of walking with an exoskeleton, based on the paper by Koelewijn & Selinger (2022). The example is added to show how models can be extended by assistive devices, in this case an exoskeleton, and to show how predictive simulations can be generated with the `BioMAC-Sim-Toolbox`. The code in the folder `IMUTracking2D` can be used to generate tracking simulations based on inertial sensor data. This example is added to show how inertial sensor data can be tracked in a 2D musculoskeletal model, and is based on the publications by Dorschky, Nitschke, et al. (2019) and Dorschky et al. (2025). The code in `MarkerTracking3D` can also be used to generate tracking simulations, but from marker and ground reaction force data. This example is added as an example of how the 3D musculoskeletal model has been used and is based on the publication by Nitschke et al. (2024). The code in `Treadmill` can be used to generate simulations of walking on the treadmill. This example is not based on published work, but was added to show how a treadmill can be implemented in the ground contact model and how the `gait2d_osim` model can be used in the `BioMAC-Sim-Toolbox`.

Currently, the toolbox can be used to solve trajectory optimization problems and create predictive and reconstructive simulations. We have implemented code to track marker positions, joint angles, translations, ground reaction forces, accelerations, angular velocities, movement duration, and movement speed. We have further implemented objectives to minimize muscular effort, metabolic cost, squared torque, joint accelerations, passive joint moments, ground reaction force impact, and head stability. By combining these different objectives, many different types of simulations can be generated and the movement kinematics and kinetics investigated. To generate new simulations, it is necessary to write scripts in MATLAB. We recommend to use functions, as done in the examples, to set up the different problems. Currently, it is not possible to use the software in a GUI. Due to these limitations, the learning curve is steep for new users. To help new users, we have added slides that were presented in a tutorial at the 2025 Conference of the International Society of Biomechanics to explain the concepts that are used in the toolbox.

Several software suits exist with similar functionality, such as OpenSim Moco (Dembia et al., 2020), Bioptim (Michaud et al., 2022), and PredSim (D'hondt et al., 2024). In OpenSim Moco, more time is required to solve, e.g., a tracking problem required 130 minutes (Dembia et al., 2020), where a similar problem takes 8 minutes with the `BioMAC-Sim-Toolbox` (Koelewijn & Van den Bogert, 2016). Furthermore, Bioptim and PredSim do not include functionality to track experimental measurement data and therefore are limited to predictive simulations, while the `BioMAC-Sim-Toolbox` can be used for both tracking and predictive simulations. A disadvantage of the `BioMAC-Sim-Toolbox` is that we have, so far, only implemented one-step discretization schemes, which are not as accurate as higher-order schemes, which are implemented in the other toolboxes. We are currently working on an implementation of higher order collocation schemes as well, and plan to add methods up to the fifth order, which are two-step and three-step. Another disadvantage is that we have currently only implemented IPOPT as a solver. In the future, we aim to add other open-source solvers, such as, for example WORHP (Büskens & Wassel, 2013).

Future Work

In future, the toolbox could be expanded. We invite users to suggest new features by creating issues and also to help implement them through pull requests. We encourage users to add models, objective and constraint functions to the toolbox to expand the range of optimal control problems that can be solved. Furthermore, new problem classes can be defined, for example for inverse kinematics and inverse dynamics, or to solve muscle activations and excitations statically or dynamically. It would also be possible to implement a single-shooting approach, which would allow for models with discontinuous dynamics or control to be investigated, such as a reflex model (Geyer & Herr, 2010). Through these expansions, the `BioMAC-Sim-Toolbox` would also allow users to implement and compare different movement simulation and analysis approaches and ensure that the outcomes are as good as possible for their research questions.

Acknowledgements

We acknowledge contributions from Marko Ackermann, Dieter Heinrich, Maria Eleni Athanasiadou, Chuyi Wang, Christopher Löffelmann, Linus Hötzel, Utkarsha Shukla, Arne Kuederle, Tobias Luckfiel, Heiko Schlarb. The development of the BioMAC-Sim-Toolbox was supported by adidas AG and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through Project-ID442419336, SFB 1483 - EmpkinS and through Project-ID520189992

References

Ackermann, M., & Van den Bogert, A. J. (2010). Optimality principles for model-based prediction of human gait. *Journal of Biomechanics*, *43*(6), 1055–1060. https://doi.org/10.1016/j.jbiomech.2009.12.012

Betts, J. (2010). *Practical methods for optimal control and estimation using nonlinear programming* (Vol. 19). SIAM. https://doi.org/10.1137/1.9780898718577

Büskens, C., & Wassel, D. (2013). The ESA NLP solver WORHP. In G. Fasano & J. D. Pintér (Eds.), *Modeling and optimization in space engineering* (Vol. 73, pp. 85–110). Springer New York. https://doi.org/10.1007/978-1-4614-4469-5_4

D'hondt, L., Falisse, A., Gupta, D., Van Den Bosch, B., Buurke, T. J., Febrer-Nafría, M., Vandekerckhove, I., Afschrift, M., & De Groote, F. (2024). PredSim: A framework for rapid predictive simulations of locomotion. *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, 1208–1213. https://doi.org/10.1109/biorob60516.2024.10719735

Dembia, C. L., Bianco, N. A., Falisse, A., Hicks, J. L., & Delp, S. L. (2020). Opensim moco: Musculoskeletal optimal control. *PLOS Computational Biology*, *16*(12), e1008493. https://doi.org/10.1371/journal.pcbi.1008493

Dorschky, E., Krüger, D., Kurfess, N., Schlarb, H., Wartzack, S., Eskofier, B. M., & Bogert, A. J. van den. (2019). Optimal control simulation predicts effects of midsole materials on energy cost of running. *Com Meth Biomech Biomed Eng*, *22*(8), 869–879. https://doi.org/10.1080/10255842.2019.1601179

Dorschky, E., Nitschke, M., Mayer, M., Weygers, I., Gassner, H., Seel, T., Eskofier, B. M., & Koelewijn, A. D. (2025). Comparing sparse inertial sensor setups for sagittal-plane walking and running reconstructions. *Frontiers in Bioengineering and Biotechnology*, *13*, 1507162. https://doi.org/10.3389/fbioe.2025.1507162

Dorschky, E., Nitschke, M., Seifer, A.-K., Van den Bogert, A. J., & Eskofier, B. M. (2019). Estimation of gait kinematics and kinetics from inertial sensor data using optimal control of musculoskeletal models. *Journal of Biomechanics*, *95*, 109278. https://doi.org/10.1016/j.jbiomech.2019.07.022

Falisse, A., Serrancolí, G., Dembia, C. L., Gillis, J., Jonkers, I., & De Groote, F. (2019). Rapid predictive simulations with complex musculoskeletal models suggest that diverse healthy and pathological human gaits can emerge from similar control strategies. *Journal of The Royal Society Interface*, *16*(157), 20190402. https://doi.org/10.1098/rsif.2019.0402

Gambietz, M., Dröge, A., Schüßler, C., Stahlke, M., Wirth, V., Miehling, J., Vossiek, M., & Koelewijn, A. D. (2024). Unobtrusive gait reconstructions using radar-based optimal control simulations. *2024 58th Asilomar Conference on Signals, Systems, and Computers*, 1505–1509. https://doi.org/10.1109/ieeeconf60004.2024.10942908

Geyer, H., & Herr, H. (2010). A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *18*(3), 263–273. https://doi.org/10.1109/tnsre.2010.2047592

Koelewijn, A. D., Dorschky, E., & Van den Bogert, A. J. (2018). A metabolic energy expenditure model with a continuous first derivative and its application to predictive simulations of gait. *Computer Methods in Biomechanics and Biomedical Engineering*, *21*(8), 521–531. https://doi.org/10.1080/10255842.2018.1490954

Koelewijn, A. D., & Selinger, J. C. (2022). Predictive simulations to replicate human gait adaptations and energetics with exoskeletons. *IEEE Trans Neur Sys Rehab Eng*, *30*, 1931–1940. https://doi.org/10.1109/TNSRE.2022.3189038

Koelewijn, A. D., & Van den Bogert, A. J. (2016). Joint contact forces can be reduced by improving joint moment symmetry in below-knee amputee gait simulations. *Gait & Posture*, *49*, 219–225. https://doi.org/10.1016/j.gaitpost.2016.07.007

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

Nitschke, M., Dorschky, E., Heinrich, D., Schlarb, H., Eskofier, B. M., Koelewijn, A. D., & Bogert, A. J. van den. (2020). Efficient trajectory optimization for curved running using a 3D musculoskeletal model with implicit dynamics. *Scientific Reports*, *10*(1), 17655. https://doi.org/10.1038/s41598-020-73856-w

Nitschke, M., Dorschky, E., Leyendecker, S., Eskofier, B. M., & Koelewijn, A. D. (2024). Estimating 3D kinematics and kinetics from virtual inertial sensor data through musculoskeletal movement simulations. *Frontiers in Bioengineering and Biotechnology*, *12*, 1285845. https://doi.org/10.3389/fbioe.2024.1285845

Nitschke, M., Marzilger, R., Leyendecker, S., Eskofier, B. M., & Koelewijn, A. D. (2023). Change the direction: 3D optimal control simulation by directly tracking marker and ground reaction force data. *PeerJ*, *11*, e14852. https://doi.org/10.7717/peerj.14852

Seth, A., Hicks, J. L., Uchida, T. K., Habib, A., Dembia, C. L., Dunne, J. J., Ong, C. F., DeMers, M. S., Rajagopal, A., Millard, M., & others. (2018). OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. *PLoS Computational Biology*, *14*(7), e1006223. https://doi.org/10.1371/journal.pcbi.1006223

Wächter, A., & Biegler, L. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25–57. https://doi.org/10.1007/s10107-004-0559-y

Zarrugh, M., Todd, F., & Ralston, H. (1974). Optimization of energy expenditure during level walking. *European Journal of Applied Physiology and Occupational Physiology*, *33*(4), 293–306. https://doi.org/10.1007/BF00430237