The Journal of Open Source Software

Kinetics Toolkit: An Open-Source Python Package to Facilitate Research in Biomechanics

Félix Chénier[1, 2]

1 Department of Physical Activity Sciences, Université du Québec à Montréal (UQAM), Montreal, Canada **2** Mobility and Adaptive Sports Research Lab, Centre for Interdisciplinary Research in Rehabilitation of Greater Montreal (CRIR), Montreal, Canada

DOI: 10.21105/joss.03714

Software

- Review
- Repository
- Archive

Editor: Marie E. Rognes

Reviewers:

- @alcantarar
- @melund

Submitted: 08 June 2021
Published: 23 October 2021

Summary

Kinetics Toolkit is a Python package for generic biomechanical analysis of human motion that is easily accessible by new programmers. The only prerequisite for using this toolkit is having minimal to moderate skills in Python and Numpy.

While Kinetics Toolkit provides a dedicated class for containing and manipulating data (`TimeSeries`), it loosely follows a procedural programming paradigm where processes are grouped as interrelated functions in different submodules, which is consistent with how people are generally introduced to programming. Each function has a limited and well-defined scope, making Kinetics Toolkit generic and expandable. Particular care is given to documentation, with extensive tutorials and API references. Special attention is also given to interoperability with other software programs by using Pandas Dataframes (and therefore CSV files, Excel files, etc.), JSON files or C3D files as intermediate data containers.

Kinetics Toolkit is accessible at `https://kineticstoolkit.uqam.ca` and is distributed via conda and pip.

Statement of need

The last decade has been marked by the development of several powerful open-source software programs in biomechanics. Examples include: OpenSim (Seth et al., 2018), SimBody (Sherman et al., 2011), Biordb (Michaud & Begon, 2021a), BiomechZoo (Dixon et al., 2017), Pinocchio (Carpentier et al., 2019), FreeBody (Cleather & Bull, 2015), CusToM (Muller et al., 2019), as well as many others. However, many of these tools are rather specific (e.g., musculoskeletal modelling, neuromuscular optimization, etc.) and not especially well suited for performing generic processing of human motion data such as filtering data, segmenting cycles, changing coordinate systems, etc. Other software programs, while being open source, rely on expensive closed-source software such as Matlab (Mathworks LCC, Naticks, USA).

While Matlab has a long and successful history in biomechanical analysis, it is quickly becoming challenged by the free and open-source Python scientific ecosystem, particularly by powerful packages, including Numpy (Harris et al., 2020), Matplotlib (Hunter, 2007), SciPy (Virtanen et al., 2020) and Pandas (McKinney, 2011). Since Python is regarded as a robust introductory programming language for algorithm development (Fangohr, 2004), it may be an ideal tool for new programmers in biomechanics.

The Pyomeca toolbox (Martinez et al., 2020) is a Python library for biomechanical analysis. It uses an object-oriented programming paradigm where each data class (`Angles`, `Rototrans`, `Analogs`, `Markers`) subclasses xarray (Hoyer & Hamman, 2017), and where the data processing functions are accessible as class methods. While this paradigm may be compelling from a programmer's perspective, it requires users to master xarray and object-oriented concepts such as class inheritance, which are not as straightforward to learn, especially for new programmers who may just be starting out with Python and Numpy.

With this beginner audience in mind, Kinetics Toolkit is a Python package for generic biomechanical analysis of human motion. It is a user-friendly tool for people with little experience in programming, yet elegant, fun to use and still appealing to experienced programmers. Designed with a mainly procedural programming paradigm, its data processing functions can be used directly as examples so that users can build their own scripts, functions, and even modules, and therefore make Kinetics Toolkit fit their own specific needs.

Features

TimeSeries

Most biomechanical data is multidimensional and vary in time. To make it easier for researchers to manipulate such data, Kinetics Toolkit provides the `TimeSeries` data class. Largely inspired by Matlab's `timeseries` and `tscollection`, this data class contains the following attributes:

- `time`: Unidimensional numpy array that contains the time;
- `data`: Dict or numpy arrays, with the arrays' first dimension corresponding to time;
- `time_info` and `data_info`: Metadata corresponding to time and data (e.g., units);
- `events`: Optional list of events.

In addition to storing data, it also provides methods to:

- manage events (e.g., `add_event`, `rename_event`);
- manage metadata (e.g., `add_data_info`, `remove_data_info`);
- split data based on time indexes, times or events (e.g., `get_ts_after_time`, `get_ts_between_events`);
- extract or combine data (e.g., `get_subset`, `merge`);
- convert from and to other formats (e.g., `from_dataframe`, `to_dataframe`)
- etc.

Processing data

All the data processing functions are included in submodules, for example:

- `filters` to apply frequency or time-domain filters to the TimeSeries data;
- `cycles` to detect and time-normalize cycles;
- `geometry` to express points, vectors, and frames in different global coordinate systems;
- `kinematics` to work with C3D files – thanks to the ezc3d library (Michaud & Begon, 2021b) – and to perform higher-level manipulations on markers and rigid bodies;
- etc.

Visualizing 3D kinematics

Kinetics Toolkit provides the `Player` class, which is a simple interactive 3D visualization tool for markers, bodies and segments. The user can pan and orbit, select and follow markers, animate at different speeds and navigate in time. Since Player is based on Matplotlib, it integrates well with various setups, using either the standard Python interpreter or IPython-based environments such as Spyder or Jupyter. Being integrated with the IPython event loop, multiple Player instances can be used at the same time, without blocking the interpreter.

Saving and loading

Kinetics Toolkit provides `save` and `load` functions to store any standard Python-type data, Numpy arrays, Pandas Series and Dataframes, TimeSeries, or lists and dictionaries that contain such data types . These data are stored into a custom `ktk.zip` (which is an archive of standard JSON files) that is easily opened in other software programs such as Matlab.

Acknowledgements

We want to acknowledge the dedicated people involved in major software programs and packages used by Kinetics Toolkit, such as Python, Numpy, Matplotlib, Pandas, Jupyter, Pytest, Sphinx, and many others. We also wish to thank Benjamin Michaud for creating and maintaining the `ezc3d` package, which is used by Kinetics Toolkit to read and write C3D files.

References

Carpentier, J., Saurel, G., Buondonno, G., Mirabel, J., Lamiraux, F., Stasse, O., & Mansard, N. (2019). The pinocchio c++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. *2019 IEEE/SICE International Symposium on System Integration (SII)*, 614–619. https://doi.org/10.1109/SII.2019.8700380

Cleather, D. J., & Bull, A. M. J. (2015). The development of a segment-based musculoskeletal model of the lower limb: Introducing FreeBody. *Royal Society Open Science*, *2*(6), 140449. https://doi.org/10.1098/rsos.140449

Dixon, P. C., Loh, J. J., Michaud-Paquette, Y., & Pearsall, D. J. (2017). biomechZoo: An open-source toolbox for the processing, analysis, and visualization of biomechanical movement data. *Computer Methods and Programs in Biomedicine*, *140*, 1–10. https://doi.org/10.1016/j.cmpb.2016.11.007

Fangohr, H. (2004). A comparison of c, MATLAB, and python as teaching languages in engineering. In M. Bubak, G. D. van Albada, P. M. A. Sloot, & J. Dongarra (Eds.), *Computational science - ICCS 2004* (Vol. 3039, pp. 1210–1217). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-25944-2_157

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hoyer, S., & Hamman, J. J. (2017). Xarray: N-d labeled arrays and datasets in python. *Journal of Open Research Software*, *5*, 10. https://doi.org/10.5334/jors.148

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Martinez, R., Michaud, B., & Begon, M. (2020). Pyomeca: An open-source framework for biomechanical analysis. *Journal of Open Source Software*, *5*(53), 2431. https://doi.org/10.21105/joss.02431

McKinney, W. (2011). Pandas: A foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, *14*(9), 9.

Michaud, B., & Begon, M. (2021a). 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

Michaud, B., & Begon, M. (2021b). ezc3d: An easy C3D file i/o cross-platform solution for c++, python and MATLAB. *Journal of Open Source Software*, *6*(58), 2911. https://doi.org/10.21105/joss.02911

Muller, A., Pontonnier, C., Puchaud, P., & Dumont, G. (2019). CusToM: A matlab toolbox for musculoskeletal simulation. *Journal of Open Source Software*, *4*(33), 927. https://doi.org/10.21105/joss.00927

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., Hamner, S. R., Arnold, E. M., Yong, J. R., Lakshmikanth, S. K., Sherman, M. A., Ku, J. P., & Delp, S. L. (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

Sherman, M. A., Seth, A., & Delp, S. L. (2011). Simbody: Multibody dynamics for biomedical research. *Procedia IUTAM*, *2*, 241–261. https://doi.org/10.1016/j.piutam.2011.04.023

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S. J. van der, Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Mulbregt, P. van. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2