SarcGraph: A Python package for analyzing the contractile behavior of pluripotent stem cell-derived cardiomyocytes

Saeed Mohammadzadeh [ID] [1] and **Emma Lejeune** [ID] [2]¶

1 Department of Systems Engineering, Boston University, Massachusetts, the United States of America
2 Department of Mechanical Engineering, Boston University, Massachusetts, the United States of America ¶ Corresponding author

Summary

Heart disease remains the leading cause of death worldwide (World Health Organization, n.d.). In response to this societal burden, scientific funding agencies and researchers have invested significant effort into understanding and controlling the functional behavior of heart cells. In particular, there has been a recent and growing focus on engineered heart cells and tissue to both better understand the complex interactions that drive disease, and to repair the damaged heart. An important component of these endeavors is the study of human induced pluripotent stem cell-derived cardiomyocytes (hiPSC-CMs), cells sampled non-invasively from living humans, transformed into stem cells, and subsequently differentiated into cardiomyocytes, i.e., cardiac muscle cells. These cardiomyocytes are composed of sarcomeres, sub-cellular contractile units, that can be fluorescently labeled and visualized via z-disc proteins. One major challenge in studying hiPSC-CMs in this context is that the immaturity and structural nonlinearities of hiPSC-CMs (i.e., disordered sarcomere chain structure in comparison to the almost crystalline sarcomere structure of mature cardiomyocytes) causes significant complications for performing consistent analysis of their functional contractile characteristics. And, though multiple methods have recently been developed for analyzing images of hiPSC-CMs (Gerbin et al., 2021; Morris et al., 2020; Pasqualin et al., 2016; Pasqualini et al., 2015; Ribeiro et al., 2017; Sutcliffe et al., 2018; Telley et al., 2006), few are suitable for analyzing the asynchronous contractile behavior of beating cells (Toepfer et al., 2019). In our previous publication, we introduced a novel computational framework (`SarcGraph`) to perform this task, directly compared it to other methods, and demonstrated its state of the art efficacy and novel functionalities (Zhao et al., 2021).

Here we introduce an open-source Python package to make the `SarcGraph` approach to performing automated quantitative analysis of information-rich movies of fluorescently labeled beating hiPSC-CMs broadly accessible. In contrast to the original version of the software released in conjunction with our previous publication (Zhao et al., 2021), the updated version is better designed, more efficient, and significantly more user-friendly. In addition, there are multiple methodological and implementation updates to improve overall performance. In brief, our framework includes tools to automatically detect and track z-discs and sarcomeres in movies of beating cells, and to recover sarcomere-scale and cardiomyocyte-scale functional behavior metrics. In addition, SarcGraph includes additional functions to perform post-processing spatio-temporal analysis and data visualization to help extract rich biological information. With this further development of SarcGraph, we aim to make automated quantitative analysis of hiPSC-CMs more accessible to the broader research community. To make our framework more accessible, SarcGraph is capable of running various video and image formats and textures out of the box. And, SarcGraph can be customized and adapted by both users and future

developers. In addition to the ongoing maintenance of SarcGraph by our group, we expect a continuous contribution by other researchers to improve the software.
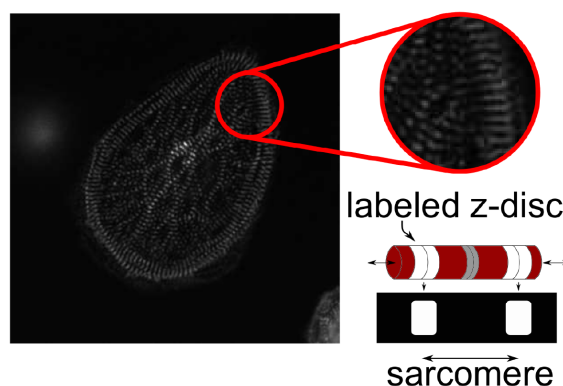

Figure 1: An example frame of a beating hiPSC-CM movie with a schematic illustrations of labeled z-discs and sarcomeres.

Statement of Need

The goal of the SarcGraph software package is to perform sarcomere segmentation, tracking, and analysis from images and movies of beating human induced pluripotent stem cell derived cardiomyocytes (hiPS-CMs). This is a challenging problem that requires specialized software for two key reasons: (1) sarcomeres are typically not imaged directly, rather their presence and geometry are inferred based on the location of labeled striations (see Figure 1), and (2) unlike mature cardiomyocytes, hiPSC-CMs often have a disordered structure and exhibit asynchronous contraction which necessitate specialized methods to extract meaning from these data. The target audience for this software is biomedical researchers who perform hiPSC-CM experiments that include image analysis. This software package is a complete re-write of our previous version of this software, though we encourage users to reference our previously published paper (Zhao et al., 2021) for additional theoretical and practical context. As a brief note for future SarcGraph users and developers, the code is structured so that the individual steps (e.g., data import, segmentation, tracking, analysis) can be used in isolation (e.g., performing segmentation without tracking on single images) to facilitate their replacement with updated algorithms (e.g., fundamentally change the segmentation algorithm but still perform downstream tracking and analysis).

SarcGraph in Action

SarcGraph provides users with tools to process images and movies of hiPSC-CMs for z-disc and sarcomere segmentation and tracking. Figure 2 demonstrates tracked z-discs and sarcomeres in the first frames of two movies of beating cells. For z-disc and sarcomere segmentation, we build on our previously described work (Zhao et al., 2021) and also implement a more efficient sarcomere detection algorithm, detailed in the Appendix. For tracking, we build on a previously developed Python package for particle tracking TrackPy (Allan et al., 2023).
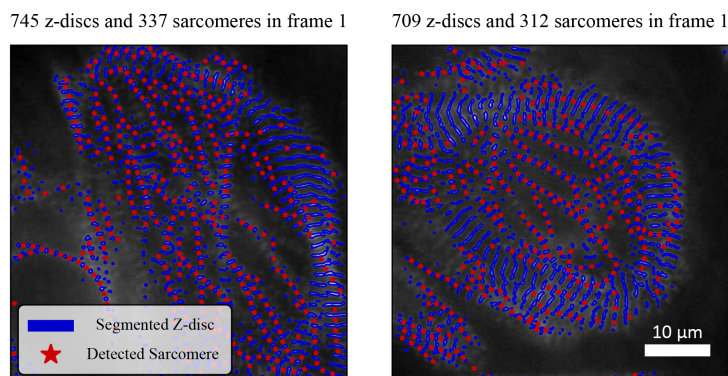
745 z-discs and 337 sarcomeres in frame 1 | 709 z-discs and 312 sarcomeres in frame 1

Figure 2: This figure showcases segmented z-discs and detected sarcomeres in the first frame of two beating hiPSC-CM movies. Detected sarcomeres are marked by red stars, while blue contours indicate z-discs. The code to generate this figure is given in Snippet 1 in the Appendix.

After initial segmentation and tracking, SarcGraph offers several post-processing analysis and visualization functions. Figure 3 showcases some of these features. Notably, there are multiple demos and tutorials that further explain these capabilities in the SarcGraph repository.
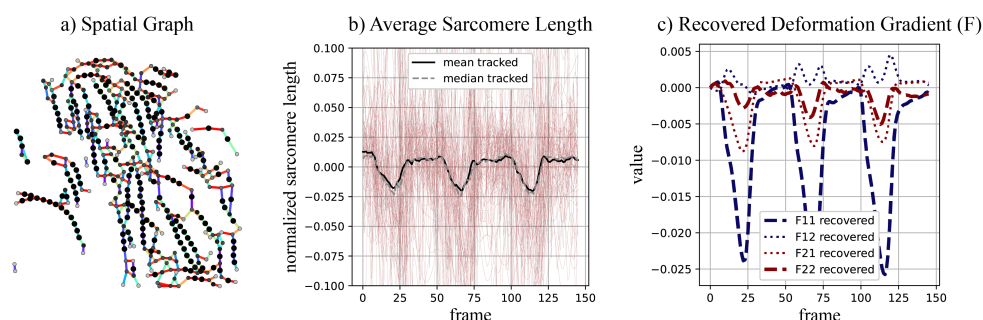

Figure 3: This figure presents three panels illustrating some of the post-processing features of the SarcGraph package on a sample movie of hiPSC-CMs. Panel (a) displays the spatial graph visualization of the segmented z-discs and sarcomeres. Panel (b) shows the average normalized sarcomere length. Panel (c) depicts the recovered deformation gradient matrix. These results demonstrate the ability of SarcGraph to analyze sarcomere dynamics in hiPSC-CMs by showcasing some of the post-processing features of the SarcGraph package. The code to generate this figure is given in Snippet 2 in the Appendix.

To validate our methods and ensure correct implementation, we generated challenging synthetic movies with characteristics similar to beating hiPSC-CMs. We used these movies to evaluate the sarcomere detection algorithm by comparing recovered metrics to their known ground truth. Figure 4 shows this process for one of the many tested validation examples.
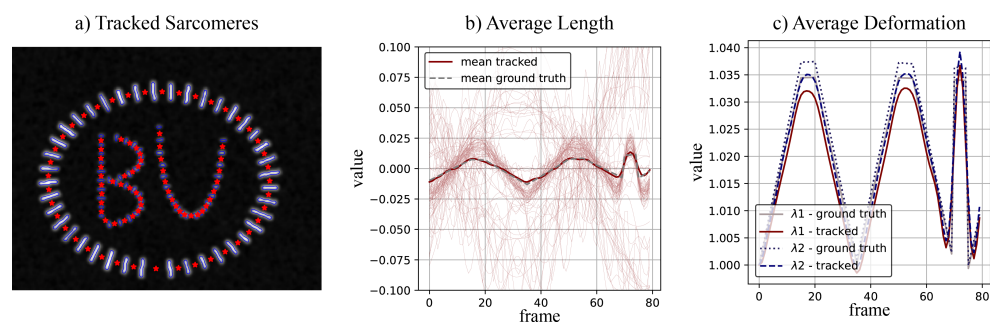
Figure 4: This figure displays the performance of SarcGraph on a synthetically generated movie designed to replicate a beating hiPSC-CM. Panel (a) shows the first frame of the movie with tracked sarcomeres marked by red stars. Panel (b) compares the average normalized sarcomere length obtained from SarcGraph with the ground truth. Panel (c) compares the parameters related to average deformation obtained from SarcGraph with the ground truth. The results demonstrate the validity of the SarcGraph package in accurately analyzing sarcomere dynamics in hiPSC-CMs. As a brief note, the updated version of SarcGraph is better able to recover the ground truth from this synthetic example in comparison to the previous version of the framework shown in Figure 6 in (Zhao et al., 2021). The code to generate this figure is given in Snippet 3 in the Appendix.

SarcGraph vs. Legacy Sarc-Graph

The updated version of the SarcGraph software has the following key improvements: (1) better structure, (2) testing via PyTest, (3) error handling, (4) efficiency improvements, (5) a novel sarcomere detection algorithm, and (6) significantly improved documentation including a documented API and multiple tutorials. This new version represents not only methodological improvements, but also an essential step towards making this a tool with longevity that can reasonably be used by others.

Highlights include a more efficient and user-friendly design, achieved through various enhancements including but not limited to a modular design (by merging eight Python files into two), improved code readability (refactoring the code and removing redundancy - reducing the lines of code from 3,094 to 2,547), and the addition of tests and error handling for a more robust experience. Efficiency improvements include the utilization of built-in NumPy functions and the introduction of a new sarcomere detection algorithm. For example, the new algorithm detects and tracks 86 sarcomeres in the synthetic image (Figure 4) within 8.6 seconds, compared to the previous version which takes 18.3 seconds to detect only 75 sarcomeres. In the experimental sample in Figure 2, left panel, the updated algorithm detects and fully tracks 329 sarcomeres in 37 seconds, while the old algorithm takes 144 seconds to track only 74 sarcomeres [1]. In addition, SarcGraph's enhanced user-friendliness is reflected in its comprehensive documentation, which includes API details and accessible Jupyter Notebook tutorials. Additionally, the package is available on both PyPI and Conda servers for easy cross-platform installation. Furthermore, we provide guidelines for advanced users to contribute to the software by adding new features and functionality, encouraging community-driven improvements. As stated previously, these user-friendly components were not present in the legacy code.

Current Limitations and Plans for Future Development

At present, the major limitation of SarcGraph is that it only gives reliable results for relatively high quality images. Very noisy and/or low resolution experimental images will lead to poor segmentation performance where inconsistent segmentation across frames leads to downstream

---

[1]We performed the analysis of all movie samples using a Dell XPS desktop computer running Windows 11, with a 2.9 GHz 9-Core Intel Core i7 processor and 16 GB of RAM.

failures in the tracking and analysis steps. Future development of SarcGraph will focus on both identifying sarcomeres in the presence of severe noise, and on adding functionality that can individually segment overlapping and partially occluded z-discs. Furthermore, the performance of the algorithm is sensitive to parameter settings, and while we have fine-tuned the parameters for the samples we worked with, users might need to adjust them for different samples. Future development of SarcGraph will draw from more diverse datasets to create automatic parameter tuning routines to avoid this issue. Finally, in the current version of SarcGraph, detected sarcomeres are not guaranteed to be correctly identified. Future development will be required to more robustly reject spurious sarcomeres.

Acknowledgements

Appendix

Sarcomere Detection Algorithm

In this implementation of SarcGraph, we introduce a novel and further customizable algorithm for sarcomere detection, which replaces the ghost points-based approach used in our previous work (Zhao et al., 2021). Our new method works by first constructing a spatial graph from the segmented z-discs in a given frame of the movie. In this initial spatial graph, each node represents a z-disc and each edge represents a potential sarcomere location where each node is initially connected to its three nearest neighbors. To score each edge, we define a function $\mathcal{S}_i$ that takes into account the length of the edge l_i, the angle between the edge and its neighboring edges $\theta_{i,j}$, the maximum allowable length for a sarcomere l_{max}, an initial guess for the average length of sarcomeres l_{avg}, and three user-defined functions f_k.

The scoring function is defined as:

$$\mathcal{S}_i = \mathbb{1}_{l_i < l_{max}} \left(\max_{j \in \{1, \ldots, n_i\}} \left(c_1 \times f_1(\theta_{i,j}) + c_2 \times f_2(l_i, l_j) \right) + c_3 \times f_3(l_i, l_{avg}) \right)$$

where n_i is the number of edges connected to edge i and,

$$f_1(\theta_{i,j}) = \mathbb{1}_{\theta_{i,j} \leq \pi/2} (1 - \theta_{i,j}/(\pi/2))^2$$
$$f_2(l_i, l_j) = (1 + |l_j - l_i|/l_i)^{-1}$$
$$f_3(l_i) = e^{-\pi(1 - l_i/l_{avg})^2}$$

and c_1, c_2, and c_3 are user-defined constants that weigh the importance of each link feature. The default values of c_k and functions for f_k in SarcGraph were selected to produce accurate results on all of the samples that we tested (both real and synthetic data) without any example-specific parameter tuning. However, it is possible to customize these values and functions to suit different scenarios where the defaults may not perform optimally. In future development of SarcGraph, automated parameter tuning will be implemented as needed.

To prune this spatial graph to retain the most probable candidates for actual sarcomeres, we adhere to four rules: (1) each node can maintain a maximum of two edges (reflecting the biological reality that a z-disc is typically part of two sarcomeres), (2) all remaining edges must exceed a threshold score, s_{max} (to discard low-probability potential sarcomeres), (3)

for nodes with two remaining edges, the angle between them must surpass a threshold, θ_{max} (which prevents predicted myofibrils from having unrealistic kinked structure), and (4) the retained edges at a node should be those with the highest scores that comply with the first three rules (preserving only the potential sarcomeres that are the most likely to be valid). The new algorithm offers improved effectiveness and flexibility over the previous method and is a key component of the new SarcGraph package's sarcomere detection capabilities.
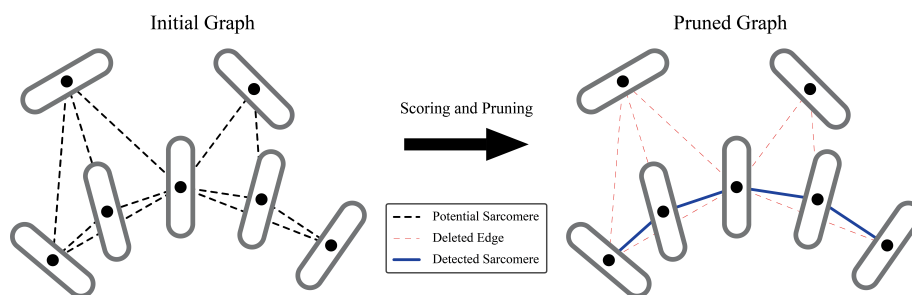

Figure 5: In the initial graph (left) the edges show the potential sarcomeres. After scoring and pruning, the remaining edges that represent the detected sarcomeres are shown in blue lines in the pruned graph (right). The edges that were deleted during pruning are shown in red dashed lines.

Instructions to Reproduce the Results Shown in this Paper

In this Appendix, we present the code snippets used to generate the figures in the main text. It should be noted that some minor aesthetic adjustments have been made to the figures after their initial generation using these snippets in order to enhance their visual clarity and presentation quality within the paper (e.g., moving legends or changing titles).

```python
from sarcgraph.sg import SarcGraph
from sarcgraph.sg_tools import SarcGraphTools

sg = SarcGraph("results", "video")
sg_tools = SarcGraphTools("results")

_, _ = sg.sarcomere_detection("sample_file") # "samples/sample_2.avi" and "samples/sample_3.avi"
_ = sg_tools.time_series.sarcomeres_gpr()

sg_tools.visualization.zdiscs_and_sarcs(frame_num=0)
```

Snippet 1: Python code snippet to generate panels in Figure 2

```python
from sarcgraph.sg import SarcGraph
from sarcgraph.sg_tools import SarcGraphTools

sg = SarcGraph("results", "video")
sg_tools = SarcGraphTools("results")

_, _ = sg.sarcomere_detection("sample_file") # "samples/sample_2.avi"
_ = sg_tools.time_series.sarcomeres_gpr()
sg_tools._run_all("sample_file")

sg_tools.visualization.spatial_graph()
sg_tools.visualization.normalized_sarcs_length()
sg_tools.visualization.F()
sg_tools.visualization.J()
```

Snippet 2: Python code snippet to generate panels in Figure 3

```python
import numpy as np
import matplotlib.pyplot as plt

from sarcgraph.sg import SarcGraph
from sarcgraph.sg_tools import SarcGraphTools


sg = SarcGraph("results", "video")
sg_tools = SarcGraphTools("results")
sg_tools_gt = SarcGraphTools("validation-data")

_, _ = sg.sarcomere_detection("samples/sample_1.avi")
sarcs = sg_tools.time_series.sarcomeres_gpr()
_ = sg_tools.analysis.compute_F_J()
lambdas = sg_tools.visualization.F_eigenval_animation(no_anim=True)

sarcs_gt = sg_tools_gt._load_sarcomeres_gpr()
_ = sg_tools_gt.analysis.compute_F_J()
lambdas_gt = sg_tools_gt.visualization.F_eigenval_animation(no_anim=True)

sg_tools.visualization.zdiscs_and_sarcs(frame_num=0)

fig, axs = plt.subplots(1, 2, figsize=(10, 5))
fig.subplots_adjust(wspace=0.3)
axs[0].grid('on')
axs[0].set_ylim(-0.1, 0.1)
axs[0].set_xlabel('frame')
axs[0].set_ylabel('value')
for _, data in sarcs.groupby('sarc_id')["length_norm"]:
    axs[0].plot(data.to_numpy(), linewidth=0.25, color="#942d28", alpha=0.25,)
axs[0].plot(sarcs.groupby('frame').length_norm.mean(), color="#942d28", label="mean tracked")
axs[0].plot(sarcs_gt.groupby('frame').length_norm.mean(), "--", color="#999a9d", label="mean ground truth")
axs[0].legend()

axs[1].grid('on')
axs[1].set_xlabel('frame')
axs[1].set_ylabel('value')
axs[1].plot(lambdas_gt[0, :], color="#a39d9e", label=r"$\lambda_1$ - ground truth")
axs[1].plot(lambdas[0, :], color="#831915", label=r"$\lambda_1$ - tracked")
axs[1].plot(lambdas_gt[1, :], ":", color="#182160", label=r"$\lambda_2$ - ground truth")
axs[1].plot(lambdas[1, :], "--", color="#182160", label=r"$\lambda_2$ - tracked")
axs[1].legend()
```

Snippet 3: Python code snippet to generate panels in Figure 4

References

Allan, D. B., Caswell, T., Keim, N. C., Wel, C. M. van der, & Verweij, R. W. (2023). *Soft-matter/trackpy: v0.6.1* (Version v0.6.1). Zenodo. https://doi.org/10.5281/zenodo.7670439

Gerbin, K. A., Grancharova, T., Donovan-Maiye, R. M., Hendershott, M. C., Anderson, H. G., Brown, J. M., Chen, J., Dinh, S. Q., Gehring, J. L., Johnson, G. R., & others. (2021). Cell states beyond transcriptomics: Integrating structural organization and gene expression in hiPSC-derived cardiomyocytes. *Cell Systems*, *12*(6), 670–687. https://doi.org/10.1016/j.cels.2021.05.001

Morris, T. A., Naik, J., Fibben, K. S., Kong, X., Kiyono, T., Yokomori, K., & Grosberg, A. (2020). Striated myocyte structural integrity: Automated analysis of sarcomeric z-discs. *PLoS Computational Biology*, *16*(3), e1007676. https://doi.org/10.1371/journal.pcbi.1007676

Pasqualin, C., Gannier, F., Yu, A., Malécot, C. O., Bredeloux, P., & Maupoil, V. (2016). SarcOptiM for ImageJ: High-frequency online sarcomere length computing on stimulated cardiomyocytes. *American Journal of Physiology-Cell Physiology*, *311*(2), C277–C283. https://doi.org/10.1152/ajpcell.00094.2016

Pasqualini, F. S., Sheehy, S. P., Agarwal, A., Aratyn-Schaus, Y., & Parker, K. K. (2015). Structural phenotyping of stem cell-derived cardiomyocytes. *Stem Cell Reports*, *4*(3), 340–347. https://doi.org/10.1016/j.stemcr.2015.01.020

Ribeiro, A. J. S., Schwab, O., Mandegar, M. A., Ang, Y.-S., Conklin, B. R., Srivastava, D., & Pruitt, B. L. (2017). Multi-imaging method to assay the contractile mechanical output of micropatterned human iPSC-derived cardiac myocytes. *Circulation Research*, *120*(10), 1572–1583. https://doi.org/10.1161/CIRCRESAHA.116.310363

Sutcliffe, M. D., Tan, P. M., Fernandez-Perez, A., Nam, Y.-J., Munshi, N. V., & Saucerman, J. J. (2018). High content analysis identifies unique morphological features of reprogrammed cardiomyocytes. *Scientific Reports*, *8*(1), 1258. https://doi.org/10.1038/s41598-018-19539-z

Telley, I. A., Denoth, J., Stüssi, E., Pfitzer, G., & Stehle, R. (2006). Half-sarcomere dynamics in myofibrils during activation and relaxation studied by tracking fluorescent markers. *Biophysical Journal*, *90*(2), 514–530. https://doi.org/10.1152/ajpcell.00094.2016

Toepfer, C. N., Sharma, A., Cicconet, M., Garfinkel, A. C., Mücke, M., Neyazi, M., Willcox, J. A., Agarwal, R., Schmid, M., Rao, J., & others. (2019). SarcTrack: An adaptable software tool for efficient large-scale analysis of sarcomere function in hiPSC-cardiomyocytes. *Circulation Research*, *124*(8), 1172–1183. https://doi.org/10.1161/CIRCRESAHA.118.314505

World Health Organization. (n.d.). *Cardiovascular diseases (CVDs) fact sheet*. https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds).

Zhao, B., Zhang, K., Chen, C. S., & Lejeune, E. (2021). Sarc-graph: Automated segmentation, tracking, and analysis of sarcomeres in hiPSC-derived cardiomyocytes. *PLoS Computational Biology*, *17*(10), e1009443. https://doi.org/10.1371/journal.pcbi.1009443